Abstract

Biometrics is refers to the automatic identification of a living person based on physiological or behavioral characteristics. Hand identification involves an analysis and measures of the features of the hand.

In this research work, we have two steps. The first step is enrollment step and the second is identification step.

In the enrollment step, the stages image capture, image binarization, edge detection and feature extraction were implemented. In the image capture, the user has to put his hand in the scanner with fingers spread freely without using any pegs. In the image binarization the color image is converted to black and white image. And in the edge detection the Laplace operator was used to find the hand boundary. In the feature extraction two types of features (geometrical and nongeometrical features) were extracted. The geometrical features are fingers length, finger width, hand span and distance between joints. The central moment to each finger after finding the fingers direction were extracted as nongeometrical features.

In the identification step, the feature vector to the unknown person is extracted from its hand image. Two methods for identify the feature vector of the unknown person with those listed in the database for 13 persons; where for each person 5 images are taken as training samples. The first adopted method is fuzzy method with difference membership function (i.e., a triangular, trapezoidal and bell shape function) and the second method is a fuzzy-neural method (fuzzy self organization map). By using any one of the above methods we can identify the feature vector of the unknown person. By the test it is shown that the trapezoidal membership function shows better performance in comparison with the others.

Acknowledgment

I would like to express my sincere appreciation to my research supervisor, Dr. Loay A. George, for giving me the major steps to go on to explore the subject, shearing with me the ideas in my research “Hand Identification using Fuzzy-Neural” And perform the points that I felt were important.

Also I wish to thank, Dr. Ban N. Al-kallak, my supervisor for her available advice and encouragement. Grateful thanks for the Head of Department of Computer Science Dr. Taha S. Bashaga.

I wish to thank the staff of Computer Science Department at the AL-Nahrain University for their help.

I would like to say "thank you" to my faithful friends for supporting and giving me advises, and to my family and my fiancé.

جمهورية العراق
وزارة التعليم العالي و البحث العلمي
جامعة النهرين

تعريف اليد
بأستخدام
المضبب-العصبي

رسالة
مقدمة الى قسم علوم الحاسبات في جامعة النهرين
كجزء من متطلبات نيل درجة الماجستير في علوم
الحاسبات

من قبل
علي محسن محمد
(بكالوريوس جامعة النهرين ٢٠٠٢)

أشراف

د. لؤي أدور جورج   د. بان نديم الكلاك

شباط ٢٠٠٥   محرم ١٤٢٦

الخلاصة

البيولوجيا الاحصائية تشير الى المطابقة الآلية للشخص الحي بالاعتماد على الخواص الفسلجية و السلوكية. مطابقة اليد تتضمن تحليل و قياس خواص اليد.

النظام المقترح يتضمن مرحلتين، الاولى هي مرحلة التحضيرات والمرحلة الثانية هي مرحلة المطابقة.

في مرحلة التحضيرات تم تنفيذ أستحصال الصورة و تحويل الصورة الى النظام الثنائي بالاظافة الى بحث الحدود و أستخراج الخواص. في مرحلة أستحصال الصورة سيضع المستعمل يده على جهاز أستحصال الصورة (Scanner) واضع أصابعه بشكل حر و بدون أستخدام مساند. في مرحلة تحويل الصورة الى النظام الثنائي فأن الصورة الملونة يتم تحويلها الى صورة تحتوي على الاسود و الابيض فقط. و بأستخدام ال(Laplace Operator) تقوم بأيجاد حدود اليد. هنالك نوعان من الخواص (هندسية و غير هندسية) يتم أستخراجها في هذة المرحلة. الخواص الهندسية هي طول الأصابع، عرض الأصابع، نصف قطر الدائرة التي ترسم في راحة اليد و ايضا المسافات بين نقاط ألتقاء الأصابع. في الخواص الغير هندسية تم حساب العزم المركزي لكل أصبع بعد أيجاد أتجاة الأصابع.

في مرحلة المطابقة، يتم أستخدام مصفوفة الخواص من يد الشخص الغير معرف. في النظام المقترح هنالك أربع طرق تم تنفيذها من أجل أجراء عملية المطابقة، حيث أن النظام يعتمد قاعدة بيانات للأشخاص المعروفين بحيث كل شخص يملك خمس مصفوفات للخواص معتمدة من خمس عينات من الصور لنفس يد الشخص المعرف. أول ثلاث طرق معتمدة هي الطرق الضبابية(Fuzzy Methods) و الاختلاف بينهم هو بأستخدام دالة العضوية(Membership Function) وهذه الطرق هي (Triangular, Trapezoidal, Bell Shape Function) أما الطريقة الرابعة فهي طريقة هجينة تعتمد الضبابية و الشبكات العصبونية (Fuzzy Self Organization Map). ومن خلال الفحوصات المعتمدة أتضح أن طريقة تربزويدل (Trapezoidal Method) هي افضل الطرق المعتمدة في المطابقة.

Chapter Five
Conclusions and Future Work

5.1 Introduction

In the previous chapters, we design the hand recognition system by name "**H**and **I**dentification **U**sing **F**uzzy-**N**eural". In this chapter we show number of conclusions remarks and the future works.

5.2 Conclusions

There are many conclusions derived from the test results of the suggested system, among these conclusion are the following

1. Combining the geometrical features (fingers length, fingers width, hand span and distance between joints) with the nongeometrical features (central moment) had improved recognition accuracy.
2. The experimental results indicate that the fuzzy logic approach has better performance than the fuzzy-neural. Because the fuzzy methods (trapezoidal, triangular and bell membership function) showed higher rates of success than the fuzzy-neural.
3. The training time in fuzzy logic method is larger than the training time in the fuzzy-neural.
4. The best order of the central moment is the sixth order that give more discriminating information on the features of the finger's shape.
5. The methods based on finding the finger's direction shows better recognition performance than their corresponding which ignore the fingers direction.

5.3 Suggestions

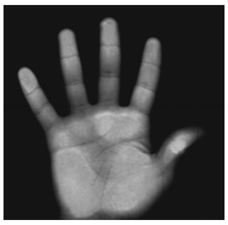
In the following a number of the future work suggestions are laid out:

1. In the propose work we used the 2D geometrical feature, and can use the 3D geometrical features to the fingers.
2. In the future work may be used the color of the hand skin as a feature.
3. Used the invariant or complex moment as a nongeometrical feature.
4. Using one of the supervise neural network methods as a finger classification method.
5. Using other methods of membership as a triangular or bell membership function in the fuzzy neural.

Chapter Four
Experimental and Result

4.1 Introduction

In chapter three, we can have discussed the two modules (enrollment module and classification module). The result of the enrollment module is creating the features vector of the unknown person. In the classification module the extracted features vector of the unknown person is matched with the features vectors listed in the database belong to 13 persons, where each one has 5 samples of its hand image. Figure (4.1) show some samples of their hands. The hand images are a BMP 24 bit/pixel and the size of each training image is 200 x 200 pixels. The matching processes will be done by using both fuzzy logic and fuzzy-neural methods. While both types of the features (geometrical and nongeometrical) will be utilizes as discriminating features.


(A) Adel

(B) Ahmed_drweish

(C) Arshed

(D) Baha

(E) Basam

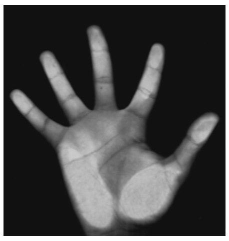
(F) Husam

(G) Mohanad

(H) Muntaser

(I) Nktal

(K) Saif


(J) Omer

(L) Wesam

(M) Yousif

Figure (4.1) Images of Hands

4.2 The Extracted Features

In the following the geometrical and nongeometrical features for two samples images belong to Baha and Nktal.

Table (4.1) shows the extracted geometrical features from the images of Baha's hand and table (4.2) shows the extracted nongeometrical features of the Baha's first hand image, and table (4.3) presents the nongeometrical feature of the Baha's second hand image. Table (4.4) shows the geometrical features extracted from the two samples images of Nktal hand, and table (4.5) lists the extracted nongeometrical features of the Nktal's first hand image and table (4.6) show the nongeometrical features of the Nktal's second hand image.

Table (4.1) The extracted geometrical features for Baha's hand

	Hand span	**Finger width**				**Finger length**	**Distance between joints**
First Image	36.3	16	17	18	20	57	27
		19	20	20	22	73	28
		19	21	22	23	78	30
		18	20	21	23	69	25
		17	19	21	22	36	22
Second Image	36.5	15	16	16	18	50	21
		17	18	18	20	67	26
		17	20	20	22	71	26
		17	18	20	22	62	24
		14	17	19	20	41	21

Table (4.2) The extracted nongeometrical features to the first test hand image for Baha's hand

Feature no.	1st. finger	2nd. finger	3rd. finger	4th.finger	5th.finger
Feature 1	1	1	1	1	1
Feature 2	2.188	2.172	2.489	2.551	2.152
Feature 3	5.322	5.410	6.549	6.784	5.388
Feature 4	13.298	13.703	17.8251	18.589	13.459
Feature 5	34.536	6.825	49.745	52.107	36.355
Feature 6	90.661	7.278	141.399	148.657	94.238
Feature 7	3.242	3.465	3.472	3.312	2.909
Feature 8	7.175	7.531	8.688	8.475	6.258
Feature 9	17.385	18.695	22.850	22.527	15.636
Feature 10	43.494	47.304	62.130	61.618	38.801
Feature 11	112.441	126.542	173.12	172.308	104.542
Feature 12	295.144	34.312	491.282	490.262	268.456
Feature 13	10.986	2.484	12.573	11.46	8.825
Feature 14	24.521	7.186	31.559	29.3698	19.009
Feature 15	59.3	7.284	82.999	78.019	47.390
Feature 16	148.429	70.198	225.477	213.058	117.045
Feature 17	382.645	453.393	627.396	594.509	314.464
Feature 18	1003.676	1198.928	1777.724	1687.462	801.745
Feature 19	38.412	6.253	46.894	40.905	27.637
Feature 20	86.332	100.985	117.986	104.941	59.688
Feature 21	208.617	49.280	310.277	278.589	148.491
Feature 22	522.204	30.648	842.173	759.638	365.605
Feature 23	1343.825	1673.9	2340.362	2115.554	979.316
Feature 24	3521.702	431.696	6621.958	5991.876	2484.216
Feature 25	137.437	74.913	178.746	149.412	88.674
Feature 26	310.625	83.031	450.515	383.537	192.202
Feature 27	750.489	43.401	1184.621	1017.467	477.237
Feature 28	1878.464	2387.431	3212.632	2770.518	1172.911
Feature 29	4828.173	317.434	8917.031	7702.273	3132.043
Feature 30	2641.959	6745.29	25196.522	21772.913	7919.093
Feature 31	500.295	71.857	692.632	555.312	289.865
Feature 32	1136.013	475.816	1748.024	1425.92	631.028
Feature 33	2745.143	628.146	4595.564	3780.086	1564.079
Feature 34	6870.288	184.948	12452.576	10279.923	3841.010
Feature 35	7643.364	4242.544	34524.319	28533.846	10224.819
Feature 36	6160.638	4323.322	97430.580	80518.924	25804.971

Table (4.3) The extracted nongeometrical features to the second tested hand image for Baha's hand

Feature no.	1st. finger	2nd. finger	3rd. finger	4th.finger	5th.finger
Feature 1	1	1	1	1	1
Feature 2	2.141	2.032	2.249	2.351	2.113
Feature 3	5.012	4.895	5.634	5.920	5.07
Feature 4	12.224	11.803	14.302	15.462	12.581
Feature 5	30.671	30.798	38.832	41.493	32.236
Feature 6	78.522	77.247	100.777	113.557	84.177
Feature 7	3.151	3.414	3.441	3.213	2.876
Feature 8	6.790	6.941	7.796	7.6	6.096
Feature 9	15.884	16.676	19.463	19.1	14.546
Feature 10	38.681	40.127	49.509	49.759	35.937
Feature 11	96.836	104.409	133.639	133.112	91.533
Feature 12	247.312	261.278	349.07	363.126	237.626
Feature 13	10.359	12.133	12.311	10.794	8.639
Feature 14	22.463	24.712	28.065	25.619	18.379
Feature 15	52.552	59.198	69.865	64.262	43.682
Feature 16	127.849	142.353	178.028	166.992	107.548
Feature 17	319.583	369.255	477.908	445.382	272.638
Feature 18	814.765	923.746	1255.657	1211.202	704.415
Feature 19	35.136	44.343	45.316	37.445	26.813
Feature 20	76.642	90.573	103.845	89.051	57.28
Feature 21	179.4	216.338	257.883	222.982	135.781
Feature 22	436.231	520.334	657.932	578.047	333.452
Feature 23	1089.34	1345.418	1757.562	1537.392	842.213
Feature 24	2773.826	3368.603	4640.072	4168.728	2167.658
Feature 25	121.995	165.431	170.392	133.021	85.316
Feature 26	267.54	338.986	392.109	316.683	183.059
Feature 27	626.734	807.46	971.786	791.701	433.235
Feature 28	1523.69	1943.361	2481.104	2047.805	1061.932
Feature 29	3802.352	5009.233	6600.171	5432.484	2674.561
Feature 30	9673.608	12560.239	17488.594	14691.152	6862.471
Feature 31	431.162	626.839	651.245	481.101	276.712
Feature 32	950.161	1288.745	1503.638	1145.955	596.414
Feature 33	2227.842	3061.949	3720.326	2860.683	1410.206
Feature 34	5416.217	7375.753	9501.491	7384.347	3451.938
Feature 35	13510.045	18955.353	25186.742	19543.848	8674.901
Feature 36	34349.081	47611.401	66911.273	52723.733	22203.232

Table (4.4) The extracted geometrical features for Nktal's hand

	Hand span	Finger width				Finger length	Distance between joints
First Image	44.5	17	18	18	20	57	25
		18	20	19	19	75	27
		19	21	20	21	80	30
		17	19	21	23	68	22
		17	21	22	21	40	22
Second Image	42.18	17	18	18	18	54	26
		17	20	20	19	73	25
		19	20	20	20	79	29
		17	19	20	22	66	22
		15	19	21	21	42	22

Table (4.5) The extracted nongeometrical features from the first tested hand for Nktal's hand

Feature no.	1st. finger	2nd. finger	3rd. finger	4th.finger	5th.finger
Feature 1	1	1	1	1	1
Feature 2	2.059	2.163	2.47	2.435	2.298
Feature 3	5.053	5.296	6.407	6.285	5.776
Feature 4	12.196	13.392	17.196	16.81	14.918
Feature 5	32.344	35.266	47.329	46.129	39.88
Feature 6	81.013	93.689	132.749	129.021	107.786
Feature 7	3.225	3.473	3.516	3.326	3.101
Feature 8	6.69	7.5	8.703	8.140	7.118
Feature 9	16.365	18.336	22.574	21.003	17.891
Feature 10	39.544	46.228	60.571	56.102	46.180
Feature 11	104.456	121.495	166.626	153.671	123.374
Feature 12	261.875	321.729	467.077	428.98	333.081
Feature 13	10.818	12.563	12.846	11.532	9.910
Feature 14	22.591	27.127	31.82	28.305	22.744
Feature 15	55.099	66.244	82.527	72.988	57.137
Feature 16	133.349	166.704	221.311	194.651	147.318
Feature 17	350.787	437.494	608.348	532.114	393.002
Feature 18	881.016	1156.005	1703.899	1482.306	1059.124
Feature 19	37.337	46.754	48.224	41.197	32.425
Feature 20	78.476	101.068	119.519	101.305	74.487
Feature 21	190.862	246.589	309.867	260.996	186.994
Feature 22	462.753	619.765	830.351	694.843	481.541
Feature 23	1212.475	1624.588	2280.454	1895.595	1282.184
Feature 24	3051.956	4286.236	6381.155	5269.149	3448.35
Feature 25	131.627	177.645	184.738	150.489	108.126
Feature 26	278.359	384.684	457.958	370.439	248.824
Feature 27	675.309	937.82	1186.709	953.375	624.221
Feature 28	1640.334	2355.013	3177.35	2533.689	1605.607
Feature 29	4282.11	6166.578	8717.599	6898.004	4266.458
Feature 30	10804.058	16251.822	24368.306	19132.996	11450.580
Feature 31	471.599	685.585	718.685	559.117	366.155
Feature 32	1003.110	1487.707	1781.612	1376.984	844.676
Feature 33	2428.326	3624.084	4613.866	3539.833	2117.788
Feature 34	5908.94	9094.811	12342.195	9391.069	5441.940
Feature 35	15374.006	23790.049	33827.902	25516.117	14431.252
Feature 36	38879.586	62645.731	94456.923	70624.887	38655.928

Table (4.6) The extracted nongeometrical features from the second tested hand for Nktal's hand

Feature no.	1st. finger	2nd. finger	3rd. finger	4th.finger	5th.finger
Feature 1	1	1	1	1	1
Feature 2	4.534	4.152	3.917	4.032	4.79
Feature 3	20.565	17.253	15.359	16.278	22.953
Feature 4	93.286	71.718	60.286	65.754	109.988
Feature 5	423.222	298.265	236.858	265.818	527.109
Feature 6	1920.397	1241.030	931.498	1075.444	2526.439
Feature 7	2.95	3.954	4.39	4.771	5.111
Feature 8	13.379	16.419	17.198	19.244	24.486
Feature 9	60.694	68.214	67.433	77.673	117.32
Feature 10	275.369	283.531	264.671	313.749	562.176
Feature 11	1249.578	1179.083	1039.844	1268.354	2694.177
Feature 12	5671.350	4905.701	4089.378	5131.450	12913.156
Feature 13	9.802	15.816	19.359	22.8	26.131
Feature 14	44.454	65.672	75.827	91.947	125.187
Feature 15	201.638	272.808	297.305	371.106	599.798
Feature 16	914.766	1133.848	1166.87	1499.01	2874.107
Feature 17	4150.736	4714.88	4584.355	6059.807	13773.813
Feature 18	18837.260	19615.701	18028.732	24516.44	66017.453
Feature 19	32.889	63.922	85.702	109.076	133.631
Feature 20	149.152	265.385	335.67	439.871	640.173
Feature 21	676.521	1102.357	1316.076	1775.316	3067.184
Feature 22	3069.1	4581.317	5165.266	7170.982	14697.233
Feature 23	13925.802	19049.335	20292.894	28988.875	70434.403
Feature 24	63198.794	79248.231	79805.016	117281.864	337588.414
Feature 25	115.519	260.726	380.86	522.468	683.521
Feature 26	523.779	1082.362	1491.671	2106.923	3274.444
Feature 27	2375.33	4495.597	5848.328	8503.44	15688.346
Feature 28	10774.065	18682.169	22952.898	34347.537	75174.491
Feature 29	48878.221	77676.792	90175.002	138850.912	360261.24
Feature 30	221784.476	323130.619	354627.896	561760.229	1726705.242
Feature 31	401.179	1072.167	1698.733	2505.63	3497.007
Feature 32	1818.972	4450.609	6653.078	10104.175	16752.491
Feature 33	8248.906	18484.386	26083.992	40779.748	80263.166
Feature 34	37415.275	76810.071	102370.708	164719.67	384598.078
Feature 35	169739.79	319343.23	402182.528	665887.519	1843112.319
Feature 36	770194.205	1328380.192	1581654.807	2694062.54	8833855.380

4.3 Matching Results

The test procedure was conducted both on the suggested fuzzy logic methods and the fuzzy-neural method.

4.3.1 Fuzzy Logic Result

After extracted the feature vector to the unknown person we have applied the fuzzy methods (Trapezoidal, Triangular or Bell membership function).

A. Trapezoidal Membership Function Results

Table (4.7) shows the results of using the trapezoidal method. In this method we take the value of ($\alpha = 2$) and the value of ($\beta = 3$). The number of training images to each person was 5 images, and the number of test images was 4. And the required training time was 734.954 second. The recognition success of this method was 100% for the training images, and 86.538% for the tested images.

Table (4.7) Trapezoidal results

No.	Person name	Training image	Success training images	Testing images	Success Test image	Success training ratio	Success tested ratio
1	Adel	5	5	4	4	100%	100%
2	Ahmed_Drweish	5	5	4	3	100%	75%
3	Arshed	5	5	4	3	100%	75%
4	Baha	5	5	4	4	100%	100%
5	Basam	5	5	4	4	100%	100%
6	Husam	5	5	4	4	100%	100%
7	Mohaned	5	5	4	4	100%	100%
8	Muntaser	5	5	4	4	100%	100%
9	Nktal	5	5	4	2	100%	50%
10	Omer	5	5	4	4	100%	100%
11	Saif	5	5	4	4	100%	100%
12	Wesam	5	5	4	2	100%	50%
13	Yousif	5	5	4	3	100%	75%

B. Triangular Membership Function Result

Table (4.8) shows the recognition results of the triangular method. In this method we take the value of $\alpha = 3.5$. The number of training images for each person was 5 images, and the test images were 4. And the required training time was 348.546 second. The ratio of success for the training images was 96.9%, while for the test images it was 84.61%.

Table (4.8) Triangular results

No.	Person name	Training image	Success training images	Testing images	Success Test image	Success training ratio	Success tested ratio
1	Adel	5	5	4	4	100%	100%
2	Ahmed_Drweish	5	5	4	4	100%	100%
3	Arshed	5	4	4	1	80%	25%
4	Baha	5	5	4	4	100%	100%
5	Basam	5	5	4	4	100%	100%
6	Husam	5	5	4	4	100%	100%
7	Mohaned	5	5	4	4	100%	100%
8	Muntaser	5	5	4	3	100%	75%
9	Nktal	5	4	4	2	80%	50%
10	Omer	5	5	4	4	100%	100%
11	Saif	5	5	4	4	100%	100%
12	Wesam	5	5	4	2	100%	50%
13	Yousif	5	5	4	4	100%	100%

C. Bell-Shape Membership Function Result

Table (4.9) shows the recognition results of bell method. In this method, we take the value of $\alpha = 3.5$. The number of training images to each person was 5 images and the test images were 4. The required training time was 253.579 second. The success ratio was 93.8% for training images, and 86.538% for tested images.

Table (4.9) Bell-shape result

No.	Person name	Training image	Success training images	Testing images	Success Test image	Success training ratio	Success tested ratio
1	Adel	5	5	4	4	100%	100%
2	Ahmed_Drweish	5	5	4	4	100%	100%
3	Arshed	5	5	4	2	100%	50%
4	Baha	5	5	4	4	100%	100%
5	Basam	5	5	4	4	100%	100%
6	Husam	5	5	4	4	100%	100%
7	Mohaned	5	5	4	4	100%	100%
8	Muntaser	5	3	4	2	60%	50%
9	Nktal	5	4	4	2	80%	50%
10	Omer	5	5	4	4	100%	100%
11	Saif	5	5	4	4	100%	100%
12	Wesam	5	5	4	3	100%	75%
13	Yousif	5	4	4	4	80%	100%

4.3.2 Fuzzy-Neural Result

In fuzzy-neural we have applied the Fuzzy Self Organization Map as a method to identify the unknown person.

Table (4.10) shows the result of applying fuzzy-neural method. In this method, we applied the trapezoidal membership function on the extracted features, after that we applied the neural network as a classification method. In the neural network, we toke the learning rate value $\alpha = 0.6$. The number of training images to each person was 5 images, and the test images were 4. In addition, the consumed training time was 120.7 second. The ratio of success was 100% for training images, and 82.69% for the tested images.

Table (4.5) Fuzzy-Neural result

No.	Person name	Training image	Success training images	Testing images	Success Test image	Success training ratio	Success tested ratio
1	Adel	5	5	4	3	100%	75%
2	Ahmed_Drweish	5	5	4	3	100%	75%
3	Arshed	5	5	4	3	100%	75%
4	Baha	5	5	4	4	100%	100%
5	Basam	5	5	4	4	100%	100%
6	Husam	5	5	4	4	100%	100%
7	Mohaned	5	5	4	4	100%	100%
8	Muntaser	5	5	4	4	100%	100%
9	Nktal	5	5	4	2	100%	50%
10	Omer	5	5	4	4	100%	100%
11	Saif	5	5	4	4	100%	100%
12	Wesam	5	5	4	2	100%	50%
13	Yousif	5	5	4	2	100%	50%

Chapter One
Introduction

1.1 Introduction

Person identification and verification using biometric methods are getting more and more important in today's information society. Hand recognition is a promising biometric because of its simplicity and fairly good performance of identification. The personal and institutional security requirements increase. The person has to remember lots of passwords, pin number and other security codes. In the future, the biometric systems will take the place of this concept since it is more convenient and reliable [Cenk02].

Hand geometry involves analyzing and measuring the shape of the hand. This biometric offer a good balances of performance characteristics and is relatively easy to use. It might be suitable where there are more users or where users access the system infrequently and are perhaps less disciplined in their approach to the system. Accuracy can be very high if desired and flexible performance tuning and configuration can accommodate a wide range of applications. Some organizations use hand geometry readers in various scenarios, including time and attendance recording, where they have proved extremely popular. Ease of integration into other systems and processes, coupled with ease of use, and makes hand geometry an obvious first step for many biometric projects [Simo01].

1.2 What is Biometric

Biometric refers to the automatic identification of a living person based on physiological or behavioral characteristics. There are many types of biometric technologies in the market: face recognition, finger print, finger geometry, hand geometry, iris recognition, vein recognition, voice and signature. The method of biometric identification is preferred over traditional methods involving password and PIN numbers for various reasons: The person to be identified is required to be physically present at the point of identification or the identification based on biometric techniques obviates the need to remember or carry a token or a smartcard. With the rapid increase in use of PIN and passwords occurring as a result of the information technology revolution, it is necessary to restrict access to sensitive/personal data. By replace PINs and passwords, biometric techniques are more convenient in relation to the user and can potentially prevent unauthorized access to or fraudulent use of ATMs, Time and Attendance System, Cellular Phones, Smart Cards, desktop PCs, Workstations, and computer networks. PINs and password may be forgotten and token based methods of identification like passports, driver's licenses and insurance cards may be forgotten, stolen or lost [Geri03].

1.3 The Difference between Identification and Verification

In day-to-day life most people with whom you do business *verify* your identity. You claim to be someone (your *claimed identity*) and then you should provide proof to back up your claim. For encounters with friends and family, there is no need to claim an identity. Instead, those familiar to you identify you, *determining* your identity upon seeing your face or hearing your voice. These two examples illustrate the difference between the two primary uses of biometrics: identification and verification.

Identification (*1:N, one-to-many, recognition*) it is the process of determining a person's identity by performing matches against multiple biometric templates. Identification systems are designed to determine identity based solely on biometric information [Zden00].

There are two types of identification systems: positive identification and negative identification.

Positive identification systems are designed to find a match for a user's biometric information in a database of biometric information. Positive identification answers the question "Who am I?", although the response is not necessarily a name – it could be an employee ID or another unique identifier.

Negative identification systems search databases in the same fashion, comparing one template against many, but are designed to ensure that a person is *not* present in a database. This prevents people from enrolling twice in a system, and is often used in large-scale public benefits programs in which users enroll multiple times to gain benefits under different names.

Verification (*1:1, matching, and authentication*) is the process of establishing the validity of a claimed identity by comparing a verification template to an enrollment template. Verification requires that an identity be claimed, after which the individual's enrollment template is located and compared with the verification template. Verification answers the question, "Am I who I claim to be?" Some verification systems perform very limited searches against multiple enrollee records. For example, a user with three enrolled finger-scan templates may be able to place any of the three fingers to verify, and the system performs 1:1 matches against the user's enrolled templates until a match is found.

One-to-few: there is a middle ground between identification and verification referred to as one-to-few *(1: few)*. This type of application involves identification of a user from a very small database of enrollees. While there is no exact number that differentiates a 1:N from a 1:few system, any system involving a search of more than 500 records is likely to be classified as 1:N. A typical use of a 1: few system would be access control to sensitive rooms at a 50-employee company, where users place their finger on a device and are located from a small database [Inte01].

1.4 Biometrics Methods

There are many types of biometric methods, the most familiar ones are those listed in the following [Hiba03]:

(I) *Eye Scanning:* it divides into two different fields. First is the iris scanning: which measures the colored band of tissue that surrounds the pupil of the eye. Second is the retina scanning: retinal scans direct a low intensity beam of light through the pupil and the back of the eye to measure the pattern of blood vessels at the back of the eye.

(II) *Face Recognition:* which analyzes the unique shape, pattern and positioning of the facial features.

(III) *Fingerprint Scanning:* it scans the series of ridges and furrows on the surface of the finger as well as the minutiae.

(IV) *Hand Geometry:* measures the shape and length of the fingers and knuckles.

(V) *Finger Geometry:* measures the finger geometry to determine the identity.

(VI) *Signature Recognition:* it measures the manner in which a user signs his/her name, stroke order, speed, pressure, and other factors which relate to the actual behavior of signing a document.

(VII) *Palm Print:* analyzes ridges, valleys and minutiae data.

1.5 Aim of Project

The aim of the project is to design a system uses the geometrical and nongeometrical features to identify the unknown person. This system name "**H**and **I**dentification **U**sing **F**uzzy-**N**eural". This project is implementing by using computer of type P4 (processor speed 1700) and captures the hand image by using genex scanner and use the language visual basic version 6.0 to implement the project.

1.6 Literature Survey

Several researches in the field of hand recognition were developed and published in the literature. The present survey includes significant previous work related to the thesis objective.

- Hiba Zuhair [Hiba03], her work was attempted to draw attention to important biometric method by designing a prototype personal authentication system using hand geometry methods. The research involves several proposed methods based on extracting the hand geometry features (finger length, finger width, etc) from the captured hand images, and compare the extracted features with the template of the claimed user, which should be previously enrolled to the system and stored in database.
- Anil K. Jain [Anil99], he work was attempted to draw attention to important biometric method by designing a prototype hand geometry

based identity authentication system and also present his preliminary verification results based on hand measurements of 50 individual captures over period time. The research uses a number of features as width of the fingers at different locations, width of the palm, thickness of the palm, length of the fingers. And used another type of feature as wrinkles of the skin.

- Milan Markovic [Mila99], his research was dedicated to the problem of two-dimensional (2D) shape recognition from the point of view of possible optimization, regarding feature extraction and classification methods. Moment invariants and stochastic AR models are considered as feature extraction methods. In this research, two types of moments were used; they are the central moment and moment invariant.
- Slobodan Ribaric [Slob98], he describes the design and development of a prototype system for the automatic identification of an individual based on the fusion of palm and hand geometry features. The hand geometry features measures the length of the fingers and the width of the four fingers at different heights. The palm print features are based on the principal lines (the heart line, the head line and the life line).

1.7 Thesis Layout

Chapter Two: "***Theoretical Overview",*** presents the basic image processing methods require to extract the geometrical and nongeometrical features and an introduction to the fuzzy concepts and explain the fuzzy-neuron concepts.

Chapter Three: "***Design and Implementation",*** in this chapter the design and implementation steps of the system are presented.

Chapter Four: "***Experimental and Result",*** in this chapter the discussing and evaluate the implementation of the system**.**

Chapter Five: "***Conclusion and Future work",*** in this chapter we implies the conclusion of the thesis and recommendations for the future work.

Chapter Three
Design and Implementation

3.1 Introduction

This chapter presents the design considerations and implementation steps of the proposed Hand Identification Using Fuzzy-Neural (HIFN). This system is designed to identify the hand's person by capturing his/her hand image using color scanner.

3.2 Description of HIFN

Figure (3.1) shows the layout of the proposed system. This system consists of two modules:

Module 1: the input to this module is the hand's image. The first stage in this module performs the preprocessing operation (it includes Image Banarization, Edge Detection, and Chain Code). The second stage is for feature extraction, in this research work two types of features were used, they are the geometrical features (length of fingers, width of fingers, hand span, distance between joints) and the non-geometrical features (find central moment of the shape of each finger). The result of this module is creating features vector for each input image.

Module 2: the input to this module is the features vector. The first stage in this module is to build a membership function and applied the fuzzification to feature vector. After this step the system will applied the feature analysis to decide which of these features have a good recognition. After finding the good discriminating features, then the matching step is performed between the unknown person features with the representative feature vectors listed in the database which contains the required template features for number of hand image samples taken for each person. The result of this module is person ID (index).

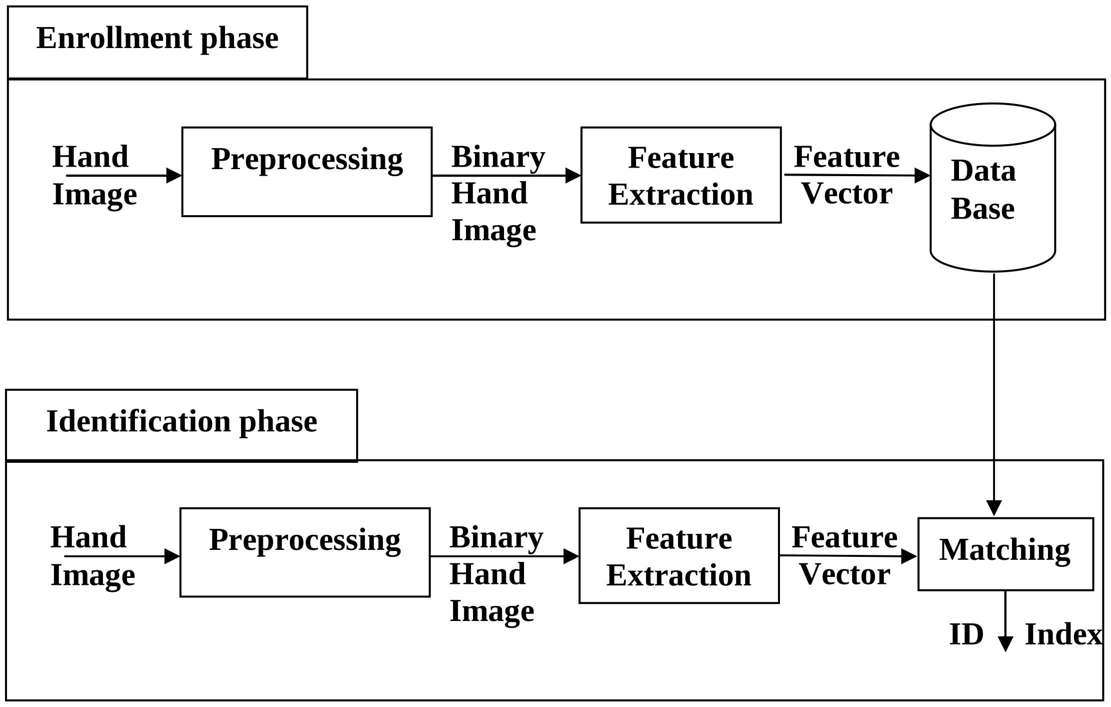


Figure (3.1) The main system

3.3 Input Stage

In this stage, a colored hand image BMP is input to the system, figure (3.2) shows a typical hand image. This step includes reading the header of the BMP image file format. This header contains the require information about the image attributes such as height and width of the image, data type, etc. This information is very important to load the image data.

Among the types of BMP file formats are the following two commonly used types [Umba98]:

(a) 8-bits/pixel: sometimes called palette driven type, in which the RGB palette table is used. The RGB palette table consists of the color contents of the image, all the existing colors in the image are listed in the RGB table, and their index table is used for encoding each pixel in the image. When a pixel is displayed, its color index is used as a palette index to load the color's (RGB) component from the palette table (one byte for red, one byte for green, one byte for blue). In 8 bits/pixel images, usually the image data begins from byte number 1078 (offset value) and ends at the location (1078+ ((Width*8+31) / 4)*4*Height).

(b) 24-bits/pixel images, unlike the 8-bits/pixel images, have on RGB palette, but it uses 8 bits to represent each of the three color bands for each pixel. The image information begins at byte number 54 (offset value).

The BMP images header has a size of 54 bytes; these bytes represent all the information required to load and present the image. The BMP header has the following structure:

```
BitMapFileHeader: record
        Bftype:  Integer
        Size: long
        Bfreserved1, Bfreserved2: integer
        Bfoffset, Bisize, Biwidth, Biheight: long
        Biplan, Bibitcount: integer
        Bicompression, Bisizeimage, Bixmeter1, Bixmeter2,
        Biclrused, Biclrimportant: long
```

The most important BMP header parameters are Biheight, Biwidth and Biplan. The Biheight represent the height of the image, and the Biwidth represent the width of the image, and the Biplan represent the number of bits required to represent each color of the pixel. Figure (3.2) show the input color BMP image. The following algorithm was used to read the BMP image

Figure (3.2) Color image.

```
Algorithm 3.1: Read Image Header.
Input: Color Image.
Output: Header record

   Open BMP color image file for read
    Read BitMapFileHeader record
   Close color image file
End.
```

3.4 Design of Enrolment Phase

This phase implies all the processes or steps required to determine the features vectors for an input image. The extracted features vectors in this phase are stored in a database, to be utilized latter in the identification phase. Figure (3.3) shows the implementation steps of the enrolment stage.

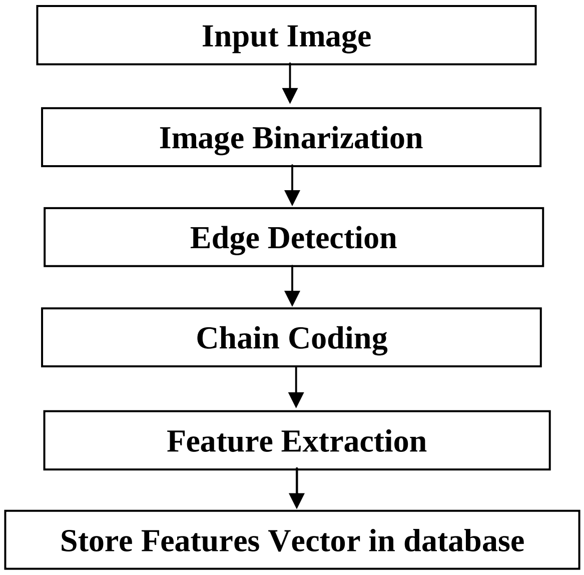


Figure (3.3) Implementation steps of the "Enrollment Phase"

3.4.1 Preprocessing Stage

Some preprocessing processes are used to make the data of the hand image more suitable for the primary data reduction and to make the analysis task easier. Preprocessing is a necessary stage when the requirements are typically obvious and simple, such as eliminate the information that is not required for the application [Umba98].

The first step of preprocessing in our suggested system is converting the color image to binary image (black and white), the second step is finding the edge of the outer boundary of the hand's person, and the third step is determining the chain code of the boundary.

3.4.1.1 Image Binarization

The first step of the image binarization process is converting the color image to gray image (its gray scale=256). The second step is converting the produced gray image to binary image (it consists of 2 colors: black and white). These two steps are implemented as follows:

(a) Convert the color image to gray image: basically each pixel in color image have three components of color (i.e.: red, green, blue), the value of each color component is represented by one byte. The gray value to each pixel is computed by using the following equation:

$$gray\_value = \frac{Red + Grean + Blue}{3} \quad ,......................(3.1)$$

The gray_value for all pixels is computed by using the above equation (3.1). Applying this equation on all image pixels will lead to convert the color image to gray image.

(b) Convert the gray image to binary image: in this step we compute the histogram of the gray image and then determine the minimum value from histogram, this minimum value is used as a threshold value to binarize the gray image. Figure (3.4) the histogram value at (0) and (255) is zero (minimum value), if you choose one of them as threshold, the gray image will converted to white or black color. But by test we finding that the minimum value between the value (55) and (80) in the histogram is the best threshold value to make the background of the image is black and set the hand's pixels white. After finding the threshold value, now we are ready to convert the gray image to black and white image. This is done by scanning all the pixels of the gray image and check the value of the pixel if it greater than the threshold value then set this pixel to white color (255) otherwise set this pixel to black color (0).

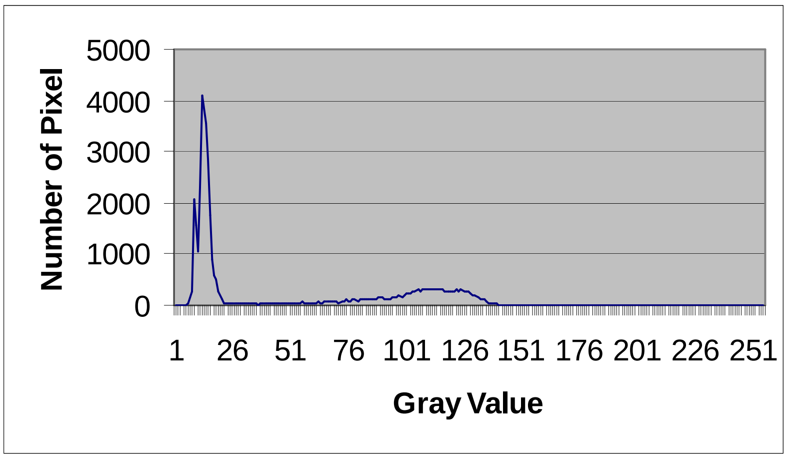


Figure (3.4) Image Histogram

Algorithm (3.2) illustrates the implemented steps to binarize the gray image.

Algorithm 3.2 // Image Banarization
Input: BMP 24 bit/pixel color image
Output: Binary_image

```
        Open BMP header file
        Loop for each pixel in color image do
              Compute gray_value using eq. 3.1
              Set Temp (pixel) =gray_value
              Set histogram (gray_value) = histogram(gray_value)+1
        End loop
        Find threshold    //the gray value (within the range 55 to 80)
                            whose corresponding histogram value is
                            minimum
        Loop for each pixel in Temp do
                If Temp (pixel)>= threshold then
                    Set binary_image (pixel) =255
                Else
                    Set binary_image (pixel) =0
                End if
        End loop
  End.
```

3.4.1.2 Edge Detection

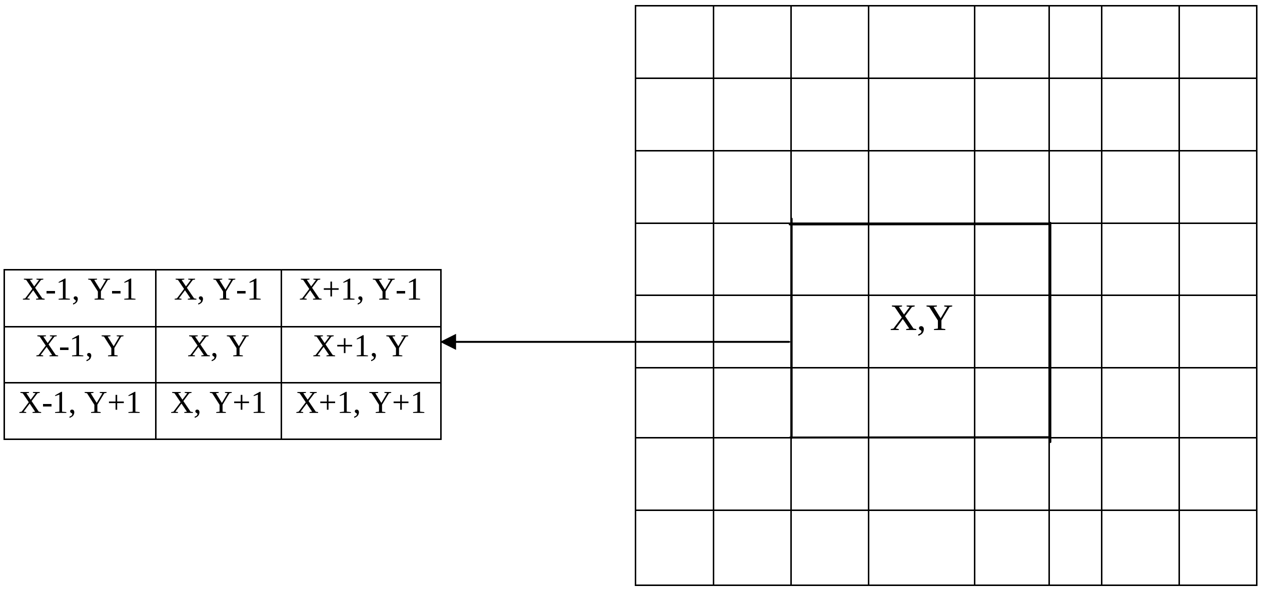
In the edge detection step the boundary is found, this will make the determination of the finger's features become easy. In this work we have used the Laplace edge detection operator.

Figure (3.5) shows the Laplace mask that passed through all the points of the images as a moving window. At each point (x,y) of the image the center of the Laplace widow is placed to cover all the neighbor pixels as illustrated in figure (3.6.a). Then the sum of multiplying the mask values by corresponding image values will give us the Laplace output at the point (x, y). It is obvious from the elements values of Laplace mask that only the four neighbors (left, right, up, down) will participate in the determination of Laplace values at each point, where the Laplace output at the point (x, y) is:

$$lap(x,y) = p(x,y-1) + p(x,y+1) + p(x-1,y) + p(x+1,y) - 4 * p(x,y) \quad ,\ldots.(3.2)$$

0	-1	0
-1	4	-1
0	-1	0

Figure (3.5) Laplace mask



a. The neighbors to pixel (x,y) **b. Binary Image**

Figure (3.6) Binary Image and Sub Image

If the value of the edge is greater than zero then the center pixel is considered as edge pixel otherwise is not edge. The following algorithm (3.3) was established to find the boundary of the hand. The input to this algorithm is binary image and the output is array of point type records, each record contains x-coordinate and y-coordinate of all boundary points of the hand.

```
Algorithm 3.3  // Laplace edge detection
Input: Binary_image
Output: boundary_array

    Loop y=0 to binary_image height
        Loop x=0 to binary_image width
            If binary_image(x, y) = 255 then
                Current_pixel = binary_image(x, y)
                Top_pixel = binary_image(x, y-1)
                bottom_pixel = binary_image(x, y+1)
                left_pixel = binary_image(x-1, y)
                right_pixel = binary_image(x+1, y)
                Compute edge   //using eq. 3.2
                If edge>0 then
                   Store current_pixel in boundary_array
                End if
            End if
        End Loop
    End Loop
End.
```

3.4.1.3 Chain Code

Chain code is used to represent the boundary as a connected sequence of straight line segments of specific length and direction [Gonz87].

After applying Laplace edge detector, the produced edge image contains only the boundary points of the hand as white points. Now we need to represent those boundary points as a sequence of adjacent points, in order to use this sequence to extract the hand features.

The main idea of chain code is constructing a one dimensional array of point type records, each contain the X and Y of the boundary pixel. The chain coding process begin by finding the start point of the hand edge (as shown in figure 3.7) and check the 8-neighbors of the start point to detect the next connected (chain) point. The coordinates of the start point is registered in the boundary array, after that we start checking its neighbors in the clockwise manner, if one of them is an edge pixel (pixel value 255) then store this pixel in the boundary array and make it as a next test point and delete the old tested (start) point, then repeat this operation to find the new connect neighbor to the new start point.

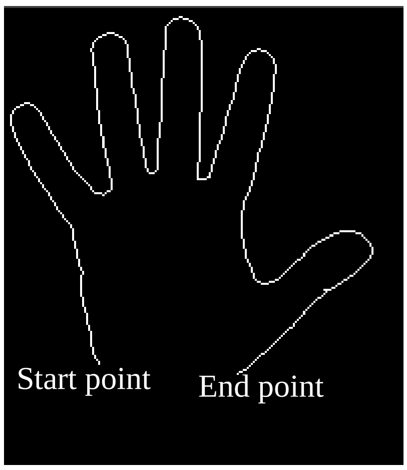
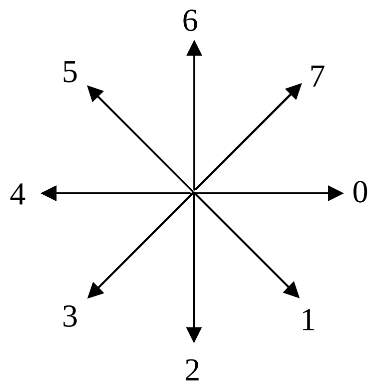


Figure (3.7) Boundary Hand Image

In the proposed work, after finding the start point and start searching its 8-neighbores to find out if any of them could be considered as a new edge point, in the case of non existence of the new neighbor connected point (i.e. whose value=255) then the detected start point should be ignored and not listed in the boundary array, in such a case the step of searching for the start point should be continued until finding out another start point with connected neighbors. In the case of finding out the correct start point then register the coordinates of the start point and all the connected adjacent points. The applied search method to check the 8-neighbors is circular, as shown in the figure (3.8).

X-1,Y-1	X,Y-1	X+1,Y-1
X-1,Y	X,Y	X+1,Y
X-1,Y+1	X,Y+1	X+1,Y+1

A. The pixel (x,y) and its 8-neighbors



B. The order of chain code

Figure (3.8) Chain code directions

From the above figure search could be started with the following direction sequence 1,2,3,4,5,6,7,0. Or it could be started from any direction and transfer points (clockwise or counter clockwise).

Algorithm (3.4) illustrates the implemented steps to trace the boundary points of the hand's person.

Algorithm 3.3 //Chain Coding Method
Input: Myimage
Output: Boundary_array

```
          Loop For i=0 to hm
              Loop for j=0 to wm
                   If Myimage(i,j)=255 then
                       Center_pixel.x=i
                       Center_pixel.y=j
                       Exit loop j and i
                    End if
             End loop
       End loop
       Set counter=0
       Set Flag=true
         Loop while flag =true
                Set boundary_array=center_pixel
                Set counter=counter+1
                Set Temp   //coordinates of the 8-neighbors of center pixel
                Set m=0
                Loop for n=8 down to 1
                         If Myimage(Temp(n).x, Temp(n).y)=255 then
                             Set Index (m) =n
                             Set m=m+1
                          End if
                End loop
                Set flag1=true
                Loop for n=1 to m
                        Set New_Temp
                        Loop for j=8 to 1
                          if Myimage(New_Temp(j).x,New_Temp(j).y)=255 then
                                Number=number+1
                          End if
                       End loop
                If Number<>0 then
                         Set center_pixel.x=Temp(index(n)).x
                         Set center_pixel.y=Temp(index(n)).y
                         Set flag1=false
                     End if
                  End loop
                  If flag1=true then
                      Set flag=false
                  End if
        End while
End.
```

Note

- The Temp and New_temp are constructed as two dimensional arrays of records contain the x and y coordinates, as shown in figure 3.8(a).
- The index( ) is an array of integers contain the index number of all edge pixels in the Temp.

3.4.2 Features Extraction

In the proposed work, we used two types of features (Geometrical features and non geometrical feature). The geometrical features include the length of the fingers, width of the fingers at different vertical positions, hand span, and distance between the joints of the fingers.

The non geometrical features include the central moment of each finger after determining the direction of that finger.

3.4.2.1 Geometrical Feature

The first step in the measurement of the geometrical features is to find the top points and joint points of the fingers, as shown in figure (3.9).

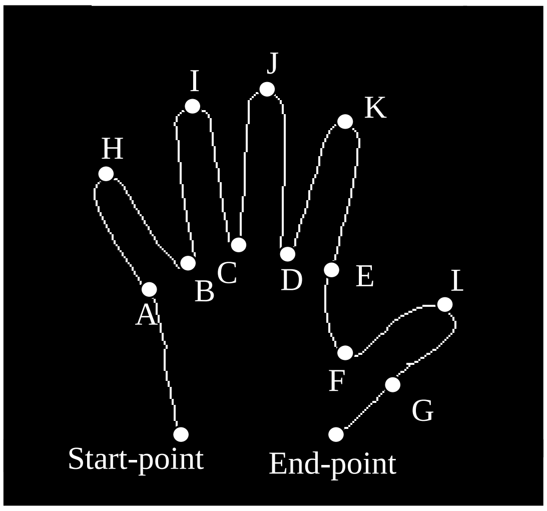


Figure (3.9) Top and Joint points

After finding all top and joint points, we can compute all geometrical features.

A. Find top and joint points

By check the y-coordinates of all the boundary points all top and some of the joint points (B, C, D, and F) could be found. An additional method was used to find the remaining joint points (A, E, G).

1. By using a method based on checking the y-coordinates the top points and the joint point (B, C, D, and F) could be found. This method utilize the criteria that the top point has y-coordinate is less than the y-coordinates of the previous and next pixels. On the other hand the joint points have y-coordinate greater than those for the previous and the next pixels. Algorithm (3.5) presents the implementation steps of find the top and joint points (B, C, D, and F) as shown in figure (3.9).

```
Algorithm 3.5// Find the Top and Joint Points
Input: boundary_array
Output: Top_list and Joint_list

        Set no_of_Top=0, Set no_of_Joint=0, Set i=2
        Loop while (no_of_Top<5)
            If (boundary_array[i+1].y>=boundary_array[i].y) and
              (boundary_array[i-1].y<boundary_array[i].y) then
                Set Top_list[no_of_Top].y= boundary_array[i].y
                Set Top_list[no_of_Top].x= boundary_array[i].x
            End if
            Set flag=true, Set m=i+1
            Loop while (flag=true)and(no_of_Top<5)
              If (boundary_array[m].y>=boundary_array[m+1].y)
                  And(boundary_array[m-1].y<boundary_array[m].y)
               then
                  Set Joint_list[no_of_joint].y=boundary_array[m].y
                  Set Joint_list[no_of_joint].x=boundary_array[m].x
                  Set no_of_joint=no_of_joint+1
                  Set flag=false
              End if
              Set m=m+1. Set i=i+1
            End while
            i=i+1
        End while
End.
```

2. To find the joint point **A**, a scanning to all the pixels that lay between the start_pixel and the top point **H** (in the sequence of boundary points) is performed to find out point **A** (which is a pixel that has minimum distance with the joint point **B).**

The same above method could be used to find the joint points **E** and **G.** To find the joint point **E** we scan the boundary pixels between the top point **K** and the joint point **F,** and we take the pixel of minimum distance with pixel **D**. The joint **G** could be defined by scanning the pixel between the top point **L** and the end_pixel, where **G** is the pixel of minimum distance with the pixel **F**.

Algorithm (3.6) was implemented to find the joint points **A, E** and **G**.

```
Algorithm 3.6// Compute some of joint points
Input: boundary_array
Output: Joint_list

Step1        // Find the joint point A
        Loop for all pixels between Start_point and top point H
              Find index    // Pixel index in the boundary_array
                               that have minimum distance with the joint
                               point B
        End Loop
        Set joint point A= boundary_array [index]
Step 2        // Find the joint point E
        Loop for all pixels between point K and point L
              Find index    // Pixel index in the boundary_array
                               that have minimum distance with the joint
                               point C
        End Loop
        Set joint point E= boundary_array [index]
Step 3       // Find the joint point G
        Loop for all pixels between point L and end_point
              Find index    // Pixel index in the boundary_array
                               that have minimum distance with the joint
                               point F
        End Loop
        Set joint point G= boundary_array [index]
End.
```

B. Extract the geometrical features

After extracting all top and joint points, the system now ready to measure the geometrical features. The geometrical features represent the length of the fingers, width of the fingers, hand span and distance between joints.

I. Fingers Length

The length of each finger represents the distance between the top of the finger and the middle point between the joint points of that finger, as shown in the figure (3.10).

The middle point could be computed from the following equation:

$$Midlle\,point = \left(\frac{joint\ A + joint\ B}{2}\right), \ldots\ldots\ldots\ldots\ldots\ldots (3.3)$$

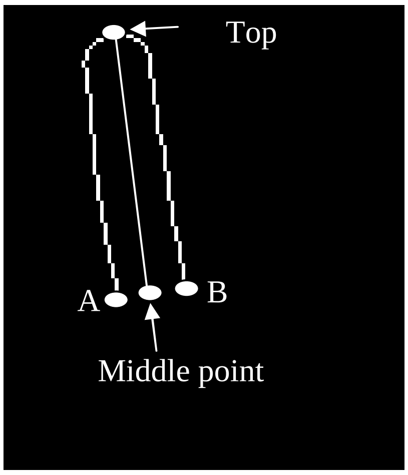


Figure (3.10) Person Finger

Algorithm (3.7) illustrates the implemented steps to compute the length of the five fingers.

```
Algorithm 3.7 // Measure the fingers length
Input: Top_list, Joint_list
Output: Length_fingers_list

        Loop for each finger
             Compute middle point   // using eq. 3.3
             Compute distance (Finger's top point, and determined
                                   Middle point)
             Store distance in Length_fingers_list.
        End loop
```

II. Finger's Width

It is represents the width of each finger at different vertical positions. The main idea of the width determination is select two corresponding pixels (left_pixel and right_pixel), such that the line passing through those should be perpendicular on the finger's longitudinal axis (i.e., the line passing through the finger's top point and it's middle point0 and intercept with it at certain mid point (i.e., the point cut the longitudinal line into $\frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}$ of its length), as shown in figure (3.11). The slope value of the width line could be computed according to the following equation:

$$\text{Slope}=\frac{-\Delta y}{\Delta x} \qquad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.4)$$

Where Δy is the vertical difference between the position of the finger's top point and its middle bottom point, and Δx is the horizontal difference between the top point and middle bottom point.

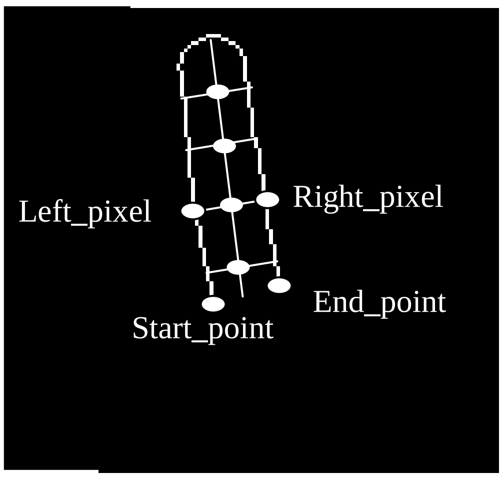


Figure (3.11) Finger width

Algorithm (3.8) illustrate the implementation of measurement the width of all fingers

```
Algorithm 3.8// Measure Finger Width
Input: finger_length_list, boundary_array, Top_list, Joint_list
Output: fingers_width_list
        Loop for each finger
          Set seg_length=finger_length/5
          Set mid1= finger_length - seg_length
          Set mid2= finger_length - 2*seg_length
          Set mid3= finger_length - 3*seg_length
          Set mid4= finger_length - 4*seg_length
          Set finger_center= (start_point + end_point)/2
          Loop for each mid point
               Compute slope    //using eq. 3.4 between mid and
                                  Finger_center
               Loop for each boundary_array pixels between start and top
                         point
                  compute f=(pixel.y-mid.y)- slope*(pixel.x-mid.x)
                 End Loop
                 Select Left_point   // pixel of minimum value of f
                Loop for each boundary_array pixel between top and end
                         point
                  compute f=(pixel.y-mid.y)- slope*(pixel.x-mid.x)
                 End Loop
                  Select Right_point   // pixel of minimum value of f
                  Compute distance  // distance between Left_point and
                                       Right_point
          End loop
     End loop
End.
```

III. Hand Span

The hand span represents the radius of the largest circle that can be drawn within the palm area of the hand.

The idea of determining the radius of the hand span that shown in figure (3.12), is based on allocating the Top and Bottom points. The radius of hand span represent the half of the distance between the Top and Bottom points.

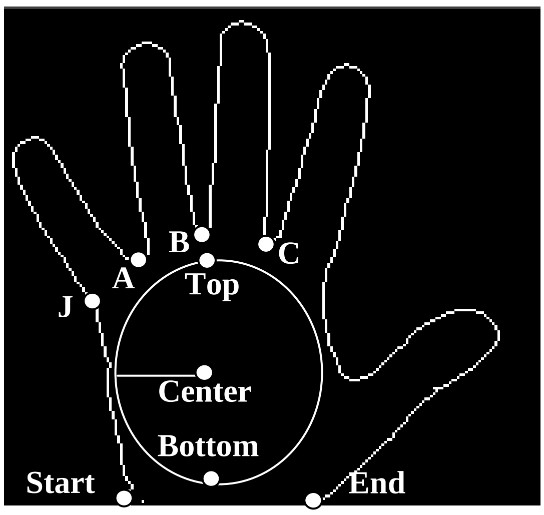


Figure (3.12) Hand Span

The Bottom point represents the middle the distance between the Start and End points. The Top point could be computed as the average of the joint points **A**, **B** and **C** as follows:

$$Top = \left(\frac{Joint\,A + Joint\,B + Joint\,C}{3} \right) \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.5)$$

The y-coordinate of the central point represents the middle distance between the Top and Bottom points. The radius of the hand span is represents the distance between the Top and the Center points.

Sometime the determined circle may draw out the hand boundary, this mean that the computed radius is not correct. The solution to this problem is done by scanning all the boundary pixels lay between the Start point and point **J**. For each scanned pixel compute the distance between that pixel and the hand center. If the distance is less than the computed radius we must decrease the radius by the half of the difference between the distance and radius. On the other hand, the hand center must be modified by the same value of the modifying the radius.

Algorithm (3.9) illustrates the steps of determining the hand span:

```
Algorithm 3.9 // Measure the Hand Span
Input: Joint-list, boundary-array
Output: Hand-span

          Find bottom_point
          Find top_point
          Compute $hand\_center = \left(\frac{top\_point + bottom\_point}{2}\right)$
          Compute radius  // distance between hand_center
                                 And top-point
          Set i=0
          Loop while boundary_array (i)$\neq$ joint point J do
                Compute new_radius  // distance boundary_array(i) and
                                           Hand_center
               If new_radius<radius then
                    $\Delta r = new\_radius - radius$
                    $hand\_center.y = hand\_center.y - \frac{\Delta r}{2}$
                    $radius = radius - \frac{\Delta r}{2}$
                End if
          End loop
          Set hand_span=radius
  End.
```

IV. Distance between joints

One of the geometrical features set is the set of distances between the successive joints of all fingers. These features were computed by scanning all fingers, and compute the distance between the previous and next joints points. Algorithm list (3.10) illustrates the steps of computing the distance between adjacent joints:

```
Algorithm 3.10 // Measure the Distance between Joints
Input: joint_list
Output: distance_joint_list

        Loop for each finger
             Compute distance  // distance between the previous
                                  joint and next joint
             Store distance in distance_joint_list
        End loop
End.
```

3.4.2.2 Nongeometrical Features

After constructing all the geometrical features (fingers length, fingers width, hand span, distance between joints), the next step is constructing the nongeometrical features which are the central moments for each finger. This is done by:

A. Partition the image into subimages to separate the fingers from the hand and create five binary subimages each one content only one finger.

B. Find the direction of the principal axis of each finger.

C. Measure the central moment for each finger along the finger's principal axis.

A. Image Partitioning

In this step, the subimage for each finger is created to simplify extracting its feature. This image is drawn by using the finger's boundary from the boundary array, as shown in the figure (3.13). Joint A and B are represents the start and end of the finger boundary.

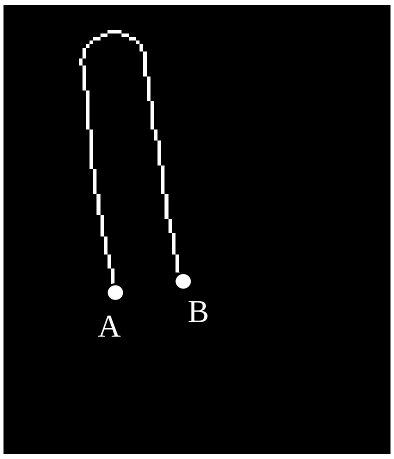


Figure (3.13) Person Finger

The produced subimages contain black background and white finger. By using the Bresenham algorithm a line between the points A and B can be drawn to split the finger from the hand. The next step is filling the inside area bounded by the finger's boundary points to white color. Algorithm list (3.11) illustrates Bresenham algorithm, and the algorithm list (3.12) was implemented to split the five fingers and create the five subimages of the fingers.

Algorithm 3.11 // Bresenham Algorithm
Input: image, point A and B
Output: draw line between A and B

```
    $\Delta x = |B.x - A.x|$
    $\Delta y = |B.y - A.y|$
    If $B.x > A.x$ then Incx=1 else If $B.x < A.x$ then Incx=-1
    Otherwise Incx=0

    If $B.y > A.y$ then Incy=1 else If $B.y < A.y$ then Incy=-1
    Otherwise Incy=0
    If $\Delta x > \Delta y$ then
        E= $-\left(\frac{\Delta y}{2}\right)$
        X= $A.x$
        Y= $A.y$
        Loop for i=0 to $\Delta x$
            Put pixel(x, y) on image
```

```
                If E>=0 then
                    Y=Y + Incy
                    E=E- $\Delta x$
                End if
                X=X + Incx
                E=E+ $\Delta y$
            End loop
        Else
        E= $-\left(\frac{\Delta x}{2}\right)$
        X= $A.x$
        Y= $A.y$
        loop for i=0 to $\Delta y$
              Put pixel(X, Y) on image
              If E>=0 then
                  X=X + Incx
                  E=E- $\Delta y$
              End if
              Y=Y + Incy
              E=E+ $\Delta x$
        End loop
    End if
End.
```

Algorithm 3.12 // Image Partitioning

```
Input: boundary_array
Output: finger_images

      Loop for i=1 to 5 do
          Create finger_image(i)
          Set prev_pixel=start joint of the $finger_i$

          Set next_pixel=next list joint of the $finger_i$

          Loop for each boundary_array pixel between prev_pixel
                     And next_pixel
                Put pixel in finger_image(i)
          End loop
          Call Bresenham algorithm(finger_image(i), prev_pixel,
          next_pixel)
          For each row in finger_image(i)
              Compute the transition position  //number of changes in the
                                                     color from black to white
              If change=2 then
                 Loop for j=first change to second change
                       finger_Image(i)(j)=white
                 End loop
              End if
       End loop
   End.
```

B. Find Finger Direction

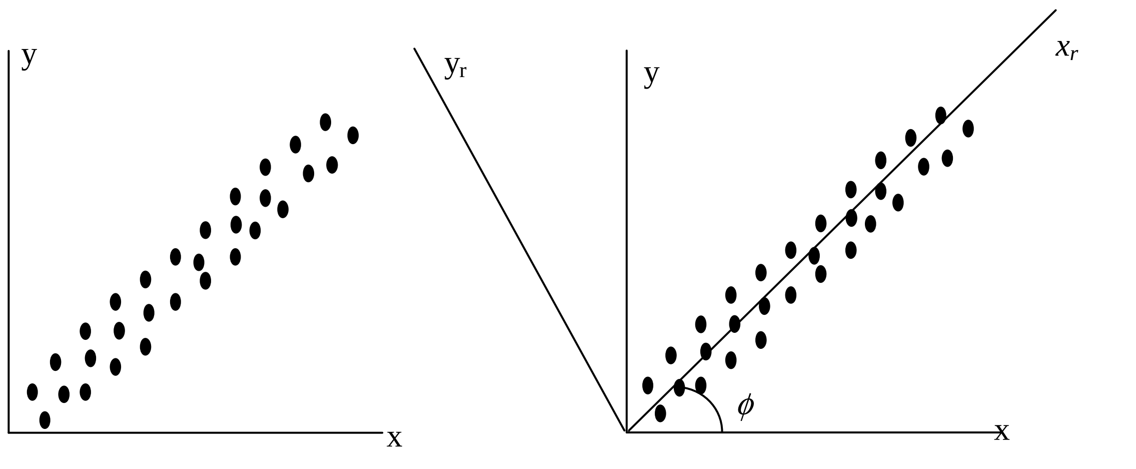
The direction of principal axis of the finger represents the finger orientation in the xy-plane (i.e. the angle between the finger and x-axis). Then the central moment, along the principal axis, for each finger could be computed.

The main idea is that each shape has a maximum variance along its longitudinal axis (i.e. assume x-axis) and minimum variance along the perpendicular axis (i.e. its y-axis). The variance is one of the measures of the dispersion of the data set. The formula of the variance is

$$\sigma^2 = \frac{\sum_{i=1}^{n}\left(x_i - \overline{x}\right)^2 M(x)}{(n-1)} \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.5)$$

Where $\overline{x}$ represents the mean of the data set and n is the number of data element. *M(x)* is the distribution function of the variable (x).

In the proposed work all white pixels in the finger images is use as a data set. The mean of the data set represent the center of the finger. Figure (3.14) shows the data set and its direction.



a. original data **b. max variance along the angle** (ϕ)

Figure (3.14) Data Direction

Where x_r and y_r represent the rotated coordinates as shown in figure (3.14.b), ϕ represents the angle between the x-axis and its rotated direction. This angle represents the direction of the maximum variance of the finger data set. In the propose work the central moment to the finger along the angle ϕ was determined. The finger variance is computed as follows:

$$\sigma_{xr}^2 = \sum_{y=0}^{hm}\sum_{x=0}^{wm} M(x,y)\left(x_r - x_c\right)^2 \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.6)$$

Where σ_{xr}^2 represents the variance of the shape along the x_r-axis, and

$$M(x,y) = \begin{Bmatrix} 1 & finger \\ 0 & outside \end{Bmatrix} \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.7)$$

x_c and y_c represent the x-coordinate and y-coordinate, respectively, of the finger center point, they computed according to the following equations:

$$x_c = \frac{\sum_{y=0}^{hm}\sum_{x=0}^{wm} M(x,y)x}{\sum_{y=0}^{hm}\sum_{x=0}^{wm} M(x,y)} \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.8)$$

$$y_c = \frac{\sum_{y=0}^{hm}\sum_{x=0}^{wm} M(x,y)y}{\sum_{y=0}^{hm}\sum_{x=0}^{wm} M(x,y)} \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.9)$$

The x_r and y_r represent the coordinates (of the rotated axis) of the finger's pixels, they computed by using the following equations:

$$x_r = (x - x_c)\cos\phi - (y - y_c)\sin\phi + x_c \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.10)$$

$$y_r = (x - x_c)\sin\phi + (y - y_c)\cos\phi + y_c \quad ,\ldots\ldots\ldots\ldots\ldots\ldots....(3.11)$$

By substituting the value of x_r in equation (3.6) the finger variance can be written as follows:

$$\sigma_{xr}^2 = \sum_{y=0}^{hm}\sum_{x=0}^{wm} M(x,y)\left((x - x_c)\cos\phi - (y - y_c)\sin\phi\right)^2 ,\ldots\ldots\ldots(3.12)$$

The finger's longitudinal axis direction (i.e. the angle ϕ) can be computed by applied the following maximization criteria:

$$\frac{\partial\sigma_{xr}^2}{\partial\phi} = 0 ,\ldots\ldots\ldots\ldots\ldots\ldots....(3.13)$$

By combine equations (3.12) and (3.13) and perform the differentiation with respect to ϕ will lead to:

$$\sum_{y=0}^{hm}\sum_{x=0}^{wm} M(x,y)\sin\phi\cos\phi\left[(y-yc)^2 - (x-xc)^2\right] - \sum_{y=0}^{hm}\sum_{x=0}^{wm} M(x,y)(x-xc)(y-yc)(\cos^2\phi - \sin^2\phi) = 0$$

After some straightforward steps we can get:

$$\phi = \frac{1}{2}\tan^{-1}\left[\frac{2 * \sum_{y=0}^{hm}\sum_{x=0}^{wm} M(x,y)(x - x_c)(y - y_c)}{\sum_{y=0}^{hm}\sum_{x=0}^{wm} M(x,y)\left[(y - y_c)^2 - (x - x_c)^2\right]}\right] ,\ldots\ldots\ldots\ldots\ldots\ldots(3.14)$$

So, the direction (ϕ) of the finger's longitudinal axis can computed by equation (3.14). Algorithm list (3.13) illustrates the implemented steps to find the directions of the fingers.

```
Algorithm 3.13  // Find the direction of the finger
Input: finger_images
Output: angles_array

        Set i=0
        For each finger_images(i)
                Compute xc  //using eq. 3.8
                Compute yc  //using eq. 3.9
                Set a=0, b=0
                For each white pixel in finger_images(i) do
                     a=a+(x-xc)*(y-yc)
                     b=b+$(y-yc)^2$-$(x-xc)^2$
                End loop
                $\phi = \frac{1}{2}\tan^{-1}(\frac{2*a}{b})$
                set angles_array(i)=$\phi$
                set i++
        End loop

End.
```

C. Compute Central Moment

The next step is to compute the central moment to the finger. For black and white image of size H.W the nth order moment is defined as follows:

$$m_{pq} = \sum_{i=1}^{H.W} x^p y^q M(x_i, y_i) \quad ,.............................(3.15)$$

Where p, q=0, 1, 2… and $M(x_i, y_i) = 1$if the pixel i at position (x_i, y_i) is set (white pixel) and 0 otherwise (black pixel).

The central moments around the finger center is given as:

$$C_{pq} = \sum_{i=1}^{H.W} (x_i - x_c)^p \ (y_i - y_c)^q M(x_i, y_i), \ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.16)$$

Where x_c, y_c represent the central coordinates of the finger that compute by using equations (3.8) and (3.9).

As shown in the figure (3.15) the central moments along the finger's longitudinal axis (x_r) and cross axis (y_x) could be determined by replacing the coordinates (x_i, y_i) by their corresponding rotated coordinated (x_r,y_r) in equation (3.16). So we can rewrite the equation (3.16) as follows:

$$C_{pq} = \sum_{i=1}^{H.W} (x_r - x_c)^p \ (y_r - y_c)^q M(x_i, y_i) \ , \ldots\ldots\ldots\ldots\ldots\ldots(3.17)$$

In the proposed work the central moments up to sixth order were computed such that they produce for each finger 36 moments features. That means that for each hand person we can construct 180 features. And the determined moments can be normalized by divided the moments by their corresponding zeroth order moment.

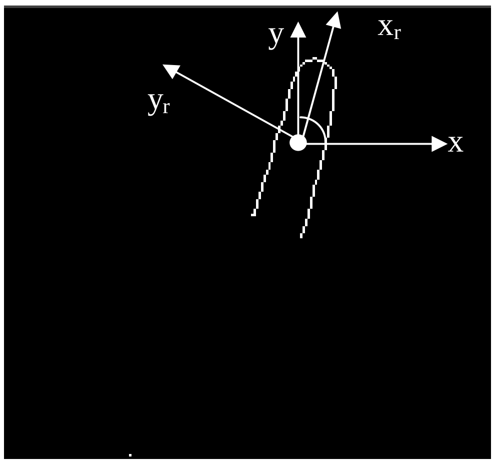


Figure (3.15) Finger direction

Algorithm (3.14) illustrates the implemented steps applied the central moments.

```
Algorithm 3.14 // Compute Central Moment
Input: finger_image
Output: moment_list

    Loop for each finger_image
        Compute $x_c$   //using eq. 3.8
        Compute $y_c$   //using eq. 3.9
        Loop for each white pixel
            Compute $x_r$   //using eq 3.10
            Compute $y_r$   //using eq. 3 11
            Loop p=0 to 5
              Loop q=0 to 5
                  $C(p,q) = C(p,q) + (x_r - x_c)^p * (y_r - y_c)^q$
              End loop
            End loop
            Normal=$C_{0,0}$
            Loop for i=0 to 5
                Loop for j=0 to 5
                    $C_{i,j} = \frac{C_{i,j}}{Normal}$
                End loop
            End loop
            Store $C$ in moment_list
   End loop
 End.
```

3.4.3 Create Features Vector

After finding all geometrical and nongeometrical features, the features vector is constructed which contains all features (fingers length, fingers width, hand span, distance between joints and central moments). The number of features is 211 feature. The data structure of the feature vector is a list of real numbers. In the propose work a database of 13 person was established, for each person have 5 hand images were used as templete samples to extract the hand features, the extracted features vectors are stored in the database.

3.5 Fuzzification Features Vector

After construct the features vector, a fuzzy membership function was applied on the features to find the membership degree for each feature. Various membership functions (as triangular, trapezoidal and bell membership function) were used in this research.

3.5.1 Triangular Membership Function

It is one of the simplest methods that used to find the membership of the features. Figure (3.16) shows the triangular membership function:

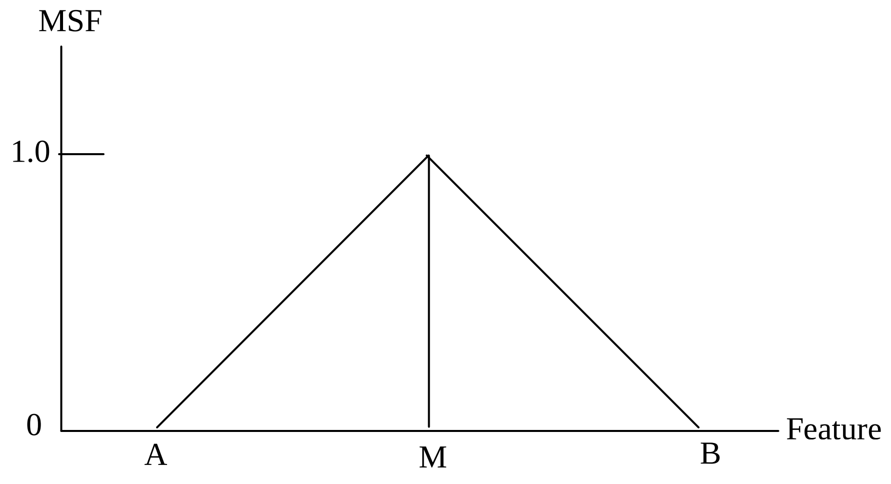


Figure (3.16) Triangular Membership Function

In the proposed work each person have 5 images this means that the $Feature_i$ have 5 values. The point M in the above figure represents the mean of the five features that computed according to the following equation:

$$M = \frac{\sum_{i=1}^{5} feature_i}{5} \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.18)$$

While, the point A and B are computed by using the following equations:

$$A = M - n*\sigma \ ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.19)$$

$$B = M + n*\sigma \ ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.20)$$

Where σ represents the standard deviation of the feature (computed by using equation (3.21), and n is a real number represent distance from the mean in terms σ.

$$\sigma = \sqrt{\frac{1}{4}\sum_{i=1}^{5}(feature_i - M)^2} \ ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.21)$$

To compute the degree of the membership (MSF) to the feature, that when the value of the feature is equal to the mean (M) of the features$_i$ the membership degree is equal to one. And when the value of the feature is greater than point B or less than point A the membership degree equal to zero. But when the feature value is between the point A and point M, the degree of MSF is given as the value of the line passing between the points T and A.

The general line equation is

$$MSF = a*feature + b \ ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.22)$$

So, to determine the line equation for the *MSF* for the interval lays between the features values from A to M, we have to apply following conditions:

1. when feature=M then *MSF*=1
2. when feature$\leq$A then MSF=0

On equation (3.22), we will get the result

$$a = \frac{1}{n\sigma} \qquad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.23)$$
$$b = 1 - \frac{M}{n\sigma}$$

Substituting the values of the a and b in equation (3.22) we will get the following *MSF* equation:

$$MSF = \frac{1}{n\sigma}(feature - M) + 1 \ ,\ldots\ldots\ldots\ldots\ldots(3.24)$$

In similar way to determine the line equation for *MSF* within the interval [M, B] we applied the following conditions:

1. when feature=M then *MSF*=1
2. when feature=B then *MSF*=0

applied these two conditions on equation (3.22) will lead to:

$$a = -\frac{1}{n\sigma} \qquad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.25)$$
$$b = 1 + \frac{M}{n\sigma}$$

And the linear relationship become:

$$MSF = \frac{1}{n\sigma}(M - feature) + 1) \ ,\ldots\ldots\ldots\ldots\ldots(3.26)$$

Algorithm (3.15) illustrates the implemented steps applied to compute the triangular membership function to extracted features vectors listed.

```
Algorithm 3.15//Triangular Membership Function
Input: person database
Output: compute membership degree (MSF)
    Loop i for all features
        Loop j for all persons
            $M(i,j)=0, STD(i,j)=0$
            Loop k for each person's test images
                $M(i,j)=M(i,j)+feature(i,j,k)$
                $STD(i,j)=STD(i,j)+(feature(i,j,k))^2$
            End loop k
            $M(i,j)=\frac{M(i,j)}{5}$
            $STD(i,j)=\sqrt{\frac{STD(i,j)-(M(i,j))^2}{5}}$
            Loop k for all tested person's images
                Set $f = feature(i,j,k)$
                If (f=M) then MSF=1
                Else If (f<M) and(f>A) then
                    $MSF(i,j,k)=\frac{1}{n*STD}*f+(1-\frac{M}{n*STD})$
                Else If (f>M) and(f<B) then
                    $MSF(i,j,k)=\frac{-1}{n*STD}*f+(1+\frac{M}{n*STD})$
            End loop k
        End loop j
    End loop i
End.
```

3.5.2 Trapezoidal Membership Function

It is one of the functions used to define the membership degree (MSF) for the feature. Figure (3.17) presents the shape of the trapezoidal function:

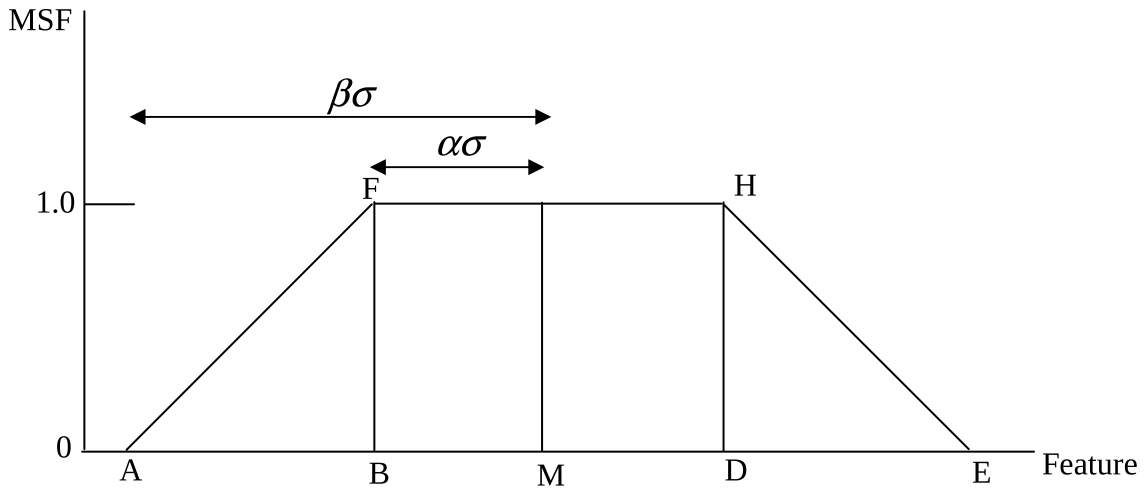


Figure (3.17) Trapezoidal Membership Function

From the figure (3.17) the coordinates of point B is $(M-\alpha\sigma,0)$, point D is $(M+\alpha\sigma,0)$, point A is $(M-\beta\sigma,0)$, point E is $(M+\beta\sigma,0)$, point H is $(M+\alpha\sigma,1)$ and point F is $(M-\alpha\sigma,1)$.

The point M represents the mean of the features. When the feature value is between the point B and D the membership degree (MSF) is equal to 1. The membership function for the feature value between the points A and B is computed by using the linear relationship represented by the line passing between the points F and A. The linear relationship could be determined by using the following conditions:

1. when $feature = M-\alpha\sigma$ then MSF =1
2. when $feature = M-\beta\sigma$ then MSF=0

Substituting these two conditions in the linear equation (3.22) we will get the following:

$$a = \frac{1}{\sigma(\beta - \alpha)} \qquad b = \frac{\sigma\beta - M}{\sigma(\beta - \alpha)} \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.27)$$

By using the values of a and b in linear equation so the *MSF* could be express as follows:

$$MSF = \frac{feature + \sigma\beta - M}{\sigma(\beta - \alpha)} \quad ,\ldots\ldots\ldots\ldots\ldots\ldots(3.28)$$

In the same way we can define the membership function for the feature interval between the points D and E, the linear relationship could be determined by using the following conditions:

1. when $feature = M + \alpha\sigma$ then *MSF* =1
2. when $feature = M + \beta\sigma$ then *MSF*=0

Substituting these two conditions in the linear equation we get the followings:

$$a = \frac{1}{\sigma(\alpha - \beta)} \qquad b = -\frac{M + \sigma\beta}{\sigma(\alpha - \beta)} \quad ,\ldots\ldots\ldots\ldots\ldots(3.29)$$

And the linear relationship for the feature interval $[M + \alpha\sigma, M + \beta\sigma]$ is:

$$MSF = \frac{feature - M - \sigma\beta}{\sigma(\alpha - \beta)} \quad ,\ldots\ldots\ldots\ldots(3.30)$$

The values of α, β are a real number. Algorithm list (3.16) illustrates the implemented steps applied to compute the trapezoidal membership function for all features extracted from the training samples.

```
Algorithm 3.16  // Trapezoidal Membership Function
Input: person database
Output: compute membership degree (MSF)

        Set $\alpha = 2, \beta = 3$
        Loop i for all features
                Loop j for all persons
                        $M(i,j) = 0, STD(i,j) = 0$
                        Loop k for each person's test images
                                $M(i,j) = M(i,j) + feature(i,j,k)$
                                $STD(i,j) = STD(i,j) + (feature(i,j,k))^2$
                        End loop k
                        $M(i,j) = \frac{M(i,j)}{5}$
                        $STD(i,j) = \sqrt{\frac{STD(i,j) - (M(i,j))^2}{5}}$
                        Loop k for all tested person's images
                                Set $f = feature(i,j,k)$
                                If (f=B) and(f(<D) then MSF=1
                                Else If (f>A) and(f<B) then
                                $MSF(i,j,k) = \frac{f + STD(i,j) * \sigma - M(i,j)}{STD(i,j) * (\beta - \alpha)}$
                                Else If (f>D) and (f<E) then
                                $MSF(i,j,k) = \frac{f - M(i,j) - STD(i,j) * \beta}{STD(i,j) * (\alpha - \beta)}$
                        End loop k
                End loop j
        End loop i
End.
```

3.5.3 Bell Shape Membership Function

It is one of the functional forms used to compute the membership degree to the feature. Figure (3.18) illustrates the bell shape function.

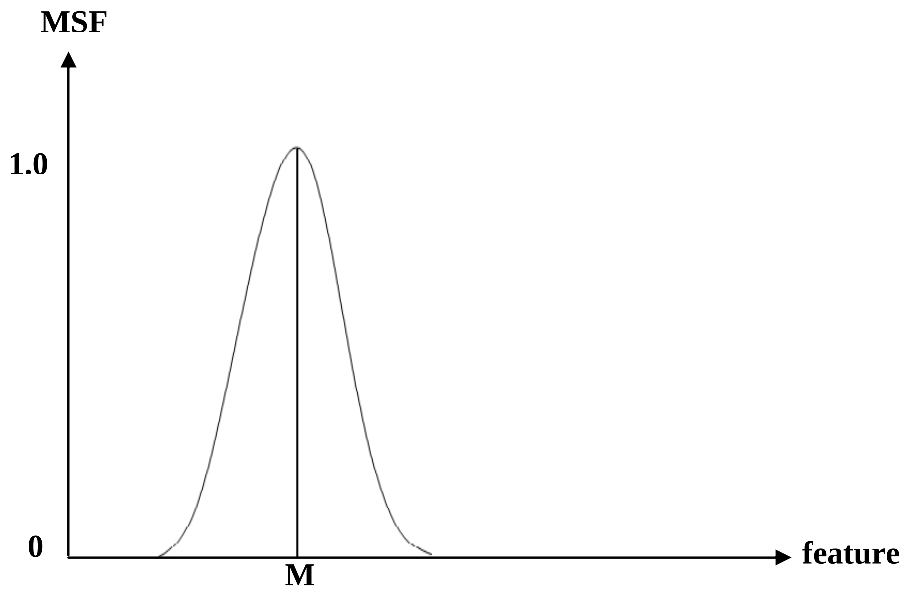


Figure (3.18) Bell membership function

We see from above figure is that the membership degree is computed from the following equation:

$$MSF = \exp\left(\left(\frac{-(feature - M)^2}{\alpha\sigma}\right)\right) \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots.. (3.31)$$

Algorithm (3.17) presents the implementation steps applied to compute the bell membership function.

```
Algorithm 3.17  // Bell Shape Membership Function
Input: person database
Output: compute membership degree (MSF)

        Set $\alpha = 3$
        Loop i for all features
                Loop j for all persons
                        $M(i,j) = 0, STD(i,j) = 0$
                        Loop k for each person's test images
                                $M(i,j) = M(i,j) + feature(i,j,k)$
                                $STD(i,j) = STD(i,j) + (feature(i,j,k))^2$
                        End loop k
                        $M(i,j) = \frac{M(i,j)}{5}$
                        $STD(i,j) = \sqrt{\frac{STD(i,j) - (M(i,j))^2}{5}}$
                        Loop k for all tested person's images
                                Set $f = feature(i,j,k)$
                                $MSF(i,j,k) = e^{\frac{-(f - M(i,j))^2}{\alpha * STD(i,j)}}$
                        End loop k
                End loop j
        End loop i
End.
```

3.6 Feature Analysis

After fuzzification of feature vector, some feature doesn't have recognition ability because they don't show specific ranges of values for each person, their ranges or wide and overlapped. For this reason we must select only the features that show distinct and non overlapped ranges of values from person to another. Algorithm list (3.18) presents the steps of evaluating the discriminating capabilities of the extracted feature.

```
Algorithm 3.18  // Select Recognition Features
Input: Feature database
Output: recognition_feature

    Loop i for all features
        Loop j for all persons
            Loop k for all tested person's images
                F= $feature(i, j, k)$
                Compute $M(i, j)$ //using equation (3.18)
                Compute $STD(i, j)$ //using equation (3.21)
                Compute $MSF1$ //using Trapezoidal function
                                and F, $M(i, j)$, $STD(i, j)$
                Index=j
                Loop s for all remain person
                    Compute $M(i, s)$ using equation (3.18)
                    Compute $STD(i, s)$ using equation (3.21)
                    Compute $MSF2$ //using Trapezoidal function
                                    and F, $M(i, s)$, $STD(i, s)$
                    If MSF2>MSF1 then
                        Index=s
                    End if
                End loop s
                If Index=j then accept++
            End loop k
        End loop j
        If accept >=threshold then
            Store F in recognition_feature
    End loop i
End.
```

The above mentioned analysis was extended such that instead of using each feature individually to determine the membership degree but a combination of two features (feature$_i$ and feature$_{i+1}$) was used in the analysis by computing the degree of the membership for both feature$_i$ and feature$_{i+1}$ and adding the determined two degrees and use the result to discrimination power of the combine features. Algorithm (3.19) was implemented to perform the selection of the combined features.

Algorithm 3.19 // Selection Combination Features
Input: person database
Output: recognition_feature_list

```
  Loop f1 for all features-1
      Loop f2 for all features
            Loop j for all persons
                  Loop k for all tested person's images
                      Fe1=feature $feature(f1, j, k)$
                      Compute $M(f1, j)$ // using equation (3.18)
                      Compute $STD(f1, j)$ // using equation (3.21)
                      Compute MSF1 //using trapezoidal function and
                                Fe1, $M(f1, j)$, $STD(f1, j)$
                      Fe2=feature $feature(f2, j, k)$
                      Compute $M(f2, j)$ // using equation (3.18)
                      Compute $STD(f2, j)$ // using equation (3.21)
                      Compute MSF2 // using trapezoidal function and
                                Fe2, $M(f2, j)$, $STD(f2, j)$
                      MSF_add1=MSF1+MSF2
                      Index=j
                      Loop s for all remain person
                          Compute $M(f1, s)$ // using equation (3.18)
                          Compute $STD(f1, s)$STD //using equation (3.21)
                          Compute MSF1 //using trapezoidal function and
                                Fe1, $M(f1, s)$, $STD(f1, s)$

                          Compute $M(f2, s)$ // using equation (3.18)
                          Compute $STD(f2, s)$ // using equation (3.21)
                          Compute MSF2 // using trapezoidal function and
                                Fe2, $M(f2, s)$, $STD(f2, s)$
                          MSF_add2=MSF1+MSF2
                          If MSF_add1>MSF_add2 then
                               Index=s
                          End if
                      End loop s
                  If Index=j then accept++
                End loop (k)
           End loop (j)
           If accept >=threshold then
           Store f1 and f2 in recognition_feature_list
      End loop (f2)
    End loop (f1)
End.
```

Where Threshold may take the value (No_of_Person * No_of_Image). And the recognition_feature_list it is array of records each record contain the index if the two discriminating features.

3.7 Identification Using Fuzzy Logic

The aim of identification stage is to identify the unknown person image and matching it with the enrolled database. In this research work we have used the Fuzzy Logic and Fuzzy Neural Network to identification the feature vector extracted from the hand image of unknown person with the template features vectors (for known persons) listed in the database and specify the ID of the person whose template feature vector shows less difference with the extracted feature vector of the unknown person. In fuzzy logic we have three methods to compute the membership function to each feature. Algorithm (3.20) was implemented to perform identification between extracted feature vector with enrolled vector. The input to this algorithm is the extracted feature vector from the hand image of the unknown person and the recognition_feature_list. And the output is the index (ID) of the person in the database whose template feature vector in the nearest to the unknown feature vector.

```
Algorithm 3.20 // Identification unknown person
Input: person database, recognition_feature_list, feature_vector
Output: person_index
    Loop x for all persons
        Loop while not end (recognition_feature_list)
            Set index1=recognitiuon_feature_list.f1
            Set index2=recognitiuon_feature_list.f2
            Set f1=feature_vector(index1)
            Set f2=feature_vector(index2)
            Compute MSF1 // using membership function and
                    f1, $M(index1, x)$, $STD(index1, x)$
            Compute MSF2 // using membership function and
                    f2, $M(index2, x)$, $STD(index2, x)$
            Sum (x) =sum (x) + (MSF1+MSF2)
        End while
    End loop
    Find person_index // index of maximum value in array sum
End.
```

3.8 Identification Using Fuzzy-Neural

Self Organization Map (SOM) is one of the methods that used to perform classification using the neural network. In this method we have two main layers the input layer and the output layer. The input layer is represents the all feature vectors that constructed from the training images. In the propose work, a database have 13 persons and each person have five training images. And the output layer is represented by the number of persons (13 nodes). In the traditional work all extracted features listed in the feature vectors were used as input, but in this research work some modifications in the implementing SOM was performed to improve its performance, these modifications are:

- Not all the features listed in the feature vector, but we applied only features passed through the analysis test perform by using fuzzy logic (trapezoidal membership function). The advantage of this step is to select only the features that have good recognition ability and make these features as the input to the input layer in the SOM. This will reduce the computational complexity of the network beside to improve its performance.
- The initialization to the weight matrix is done by using the membership value of each selected feature.

Algorithm (3.21) illustrates the implementation steps to perform Fuzzy Self Organization Map (FSOM).

Algorithm 3.21 // Fuzzy Self Organization Map (FSOM)
Input: person database, recognition_feature_list, feature_vector
Output: person_index

```
Step1: Training
      Initialization weight matrix W
      Set flag=true
    Loop while (flag=true)
      Set $\alpha$ =0.6

      Loop while ($\alpha$ >0.1)
         For k=0 to N      // N number of patterns
            For i=0 to No_Of_Person
                 Dj(i)=0
                 For j=0 to M  // M is number of feature in
                              recognition_feature_list
                 Compute MSF to $feature_{kj}$
                  Dj(i)=Dj(i) $+(w_{ji} - MSF)^2$
                 End loop
             End loop
             Find minimum Index in Dj
             For j=0 to M  // M is number of feature in
                          recognition_feature_list
                 Set f= $feature_{kj}$
                 Compute MSF to f
                 $W_{j\,index} = W_{j\,index} + \alpha(MSF - W_{j\,index})$
             End loop
             Update $\alpha = \alpha - 0.01$
         End while
         Check $\left|W^t - W^{t+1}\right| < 0.0001 then\ flag = false$
    End while
Step 2: Testing
      For j=0 to No_of_Person
         Dj(i)=0
         For i=0 to M  // M is number of feature in
                      recognition_feature_list
            Let F= $feature\_vector_i$
            Compute MSF to F
             $Dj(i) = Dj(j) + (W_{ji} - MSF)^2$
         End loop
      End loop

      Find Index of minimum value in $Dj$
End.
```

Chapter Two
Theoretical Overview

2.1 Introduction

All hand identification methods that have been proposed in the last decade use various geometric features of hand (width, height, length of the finger, hand span, etc). In addition to the above features the moments of the finger's shape were used as additional features to identify the hand.

In this chapter, a description to the identification system that used the geometry and non-geometry features of the hand to identify his/her hand.

2.2 The Pattern Recognition System

Recognition is regarded as a basic attribute of human beings, as well as other living organisms. A pattern is the description of an object. Human begins recognize the objects around them, and they move and act in relation to them. Peoples recognize the voice of a known individual; and can recognize handwritings and analyze fingerprints. A human being is a very sophisticated information system, partly because he/she possesses a superior pattern recognition capability [Gonz74].

In simple language, pattern recognition can be defined as the classification (or categorization) of input data into identifiable classes, via the extraction of significant features (or attributes) of the data from a background data of irrelevant detail. Pattern recognition involves the following major concepts [Hiba03]:

- ***A pattern*** is the description of an object. According to the nature of the patterns to be recognized, we may divide our acts of recognition into two types: the recognition of concrete items and the recognition of abstract items. The first type involves the identification and classification of spatial patterns (for example, characters; fingerprint; and hand shape; etc) and temporal patterns (for example, speech; signatures; etc). While the second type of recognition process involves the recognition of abstract items such as an old argument or a solution to a problem.

- ***A pattern class:*** the set of patterns with some given common features will determine a category. A pattern is a description of any member of a category representing a pattern class. The pattern is be recognized and classified by automatic pattern recognition system must posses a set of measurable features. When these features are similar to those belong to a group of patterns (a pattern class), the tested pattern is considered to be a member of the same pattern class.

The objective of pattern recognition system is to determine, on the basis of the observed data, the pattern class responsible for generating a set of measurements similar to the observed data [Hiba03].

The typical pattern recognition process is shown in figure (2.1), the process of input data is called preprocessing when it is used in the design of pattern recognition systems. This preprocessing step includes a number of image processing operations that transform the input image into another form. After proper preprocessing, several pattern features are extracted and the pattern vector is plotted in the pattern space, this is done in the feature extraction step. Then the proper decision theory draws the optimum boundaries in the space of the recognition during the classification step [Gonz74].

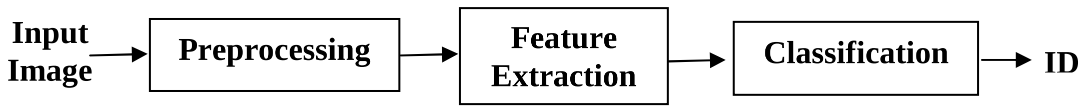


Figure (2.1) The functional Block diagram of typical pattern recognition system

2.3 Image Representation

We have seen that the human visual system receives an input image as a collection of spatially distributed light energy; this form is called an optical image. Optical images are the types we deal with everyday (cameras capture them, monitors display them, and we see them). We know that these optical images are represented as video information in the form of analog electrical signals and have seen how these are sampled to generate the digital image *I(r, c)* [Umba98].

The term monochrome image or simple image, refer to a two dimensional light intensity function *I(r, c)*, where *r* and *c* denote spatial coordinates and the value of *I* at any *point (r, c)* is proportional to the brightness (or gray level) of the image at that point [Gonz87].

The images types are: Binary, Gray-scale, Color, and Multispectral [Umbo98].

2.3.1 Binary Images

Binary images are the simplest type of images and can take two discrete values, typically black and white, or ‘0’ and ‘1’. A binary image is referred to as a 1 bit / pixel image because it takes only 1 binary digit to represent each pixel. These types of images are most frequently used in computer vision applications where the only information required for the task is general shape, or outline, information.

Binary images are often created from gray-scale images via a threshold operation, where every pixel above the threshold value is turned white (‘1’), and those below it are turned black (‘0’).

2.3.2 Gray-Scale Images

Gray-scale images are referred to as monochrome, or one- color, images. They contain brightness information only, no color information. The number of bits used for each pixel determines the number of different brightness levels available. The typical image contains 8 bits/pixel data, which allows us to have 256 (0…255) different brightness (gray) levels. Additionally, the 8 bits representation is typical due to the fact that the byte, which corresponds to 8 bits of data, is the standard small unit in the world of digital computers.

2.3.3 Color Images

Color image can be modeled as three-band monochrome image data, where each band of data corresponds to a different color. The actual information stored in the digital image data is the brightness information in each spectral band. When the image is *displayed*, the corresponding brightness information is displayed on the screen by picture elements that emit light energy corresponding to the particular color. Typical color images are represented as red, green and blue, or RGB images. Using the 8-bit monochrome standard as a model, the corresponding color image would have 24 bits/pixel, 8 bits for each of the three-color bands (red, green, and blue).

2.3.4 Multispectral Images

Multispectral images typically contain information outside the normal human perceptual range. This may include infrared, ultraviolet, X-ray, acoustic, or radar data. These are not images in the usual sense because the represented information is not directly visible by the human system. However, the information is often represented in visual form by mapping the different spectral bands to RGB components. If more than three bands of information are in the multispectral image, the dimensionality is reduced by applying a principal component transform.

2.4 Hand Image Capture

The hand geometry and the palm features are extracted from the image of the right hand, which is placed on the flat glass surface of a scanner. The user has to put his hand on the scanner with the fingers spread naturally. There are no pegs, which usually serve as control points for the appropriate placement of a hand. The spatial resolution of the images is 200 dots per inch (dpi). Figure (2.2) shows a typical image obtained by the scanner [Slob98].

A commercial flatbed scanner was used to acquire the hand images. Randomly placed hands of the participants were scanned with dark background [Alex98].

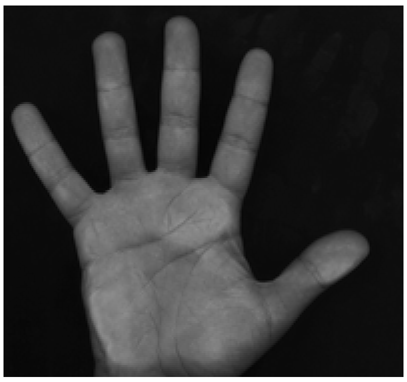


Figure (2.2) Typical image obtained by a scanner

2.5 The Preprocessing Stage

After the hand's image captured, preprocessing may be required. The first step in preprocessing is converting the color image of the hand's person to binary image (black and white), the hand is white and the background is black, the next step is applying the Laplace edge detection to find the boundary of the hand, after that the chain code of the boundary of the hand should be traced.

2.5.1 Thresholding

Thresholding is one of the most important approaches to the image segmentation. Segmentation is accomplished by scanning the image (pixel by pixel), and labeling each pixel as object or background, depending on the gray level of the that pixel whether it is greater or less than the value of chosen threshold ***T*** [Gonz87].

One of the most important and commonly used image processing techniques is Thresholding. This is the conversion of a color (typically 24-bit) or grayscale (typically 8-bit) image to a binary (1-bit) image. It is the reduction of images to black and white representations of features and background [Vasa98]. Pixels with a grey level above the threshold are set to one or white (255) and all other pixels set to zero or black as shown in figure (2.3). This produces a binary image of white objects on a black background (or otherwise, depending on the original distribution) [Vasa98].

```
Set a given point (r,c)
If   I (r, c)>=$T$   then    I(r, c) =object (or set 1)
Else                         I(r, c) =background (or set 0)
```

Figure (2.3) Binary Image

2.5.2 Edge Detection

An edge is the boundary between two regions with relatively distinct gray level properties. Basically the idea underlying most edge detection techniques is the computation of a local derivative operator [Gonz87].

Edge detection operators are based on the idea that edge information in an image is found by looking at the relationship a pixel has with its neighbors. If a pixel's gray level value is similar to those around it, there is probably not an edge at that point. However, if a pixel has neighbors with widely varying gray levels, it may represent an edge point. In other words, an edge is defined by discontinuity in gray level value [Umba98].

Edge detection generally results in a considerable reduction of image data. Some of well known edge detectors are Robert's cross, Laplace, Sobel, Kirsch, Prewitt [Toma00].

In our proposed work, we used Laplace operator to find the edges. The three Laplace masks, figure (2.4), represent different approximations of Laplace operator. Unlike compass masks, Laplace masks are rotationally symmetric, which implies "edges at all orientations contribute to the result". They are applied by selecting one mask and convolving it with the image. The sign of the result (positive or negative) from two adjacent pixel locations provides directional information, and tells us which side of the edge is brighter [Umba98].

0	-1	0
-1	4	-1
0	-1	0

1	-2	1
-2	4	-2
1	-2	1

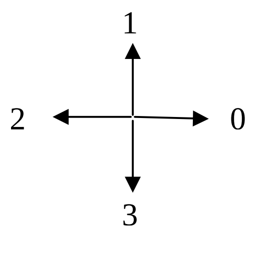
-1	-1	-1
-1	8	-1
-1	-1	-1

Figure (2.4) Laplace masks

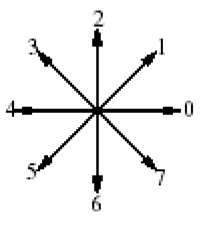
2.5.3 Chain Code

Chain codes are used to represent a boundary by a connected sequence of straight line segments of specific length and direction. Typically, this representation is based on 4 or 8 directions, each segment is coded by using a numbering scheme such as the ones shown in figure (2.5) [Gonz87].

The basic information that store about a region is where it is. One could store the location of each pixel in the region, but one can be more efficient by simply storing the border. By traversing the border in a pre-defined direction around the region, one can build a ***chain*** [Brya00].



(a)



(b)

Figure (2.5) Directions indexing for (a) 4-direction chain code.
(b) 8-direction chain code.

The connectivity neighborhood of the starting pixel are treated as a circular sequence of direction numbers as shown in figures 2.6(a) and 2.6(b), the starting pixel will be redefined so that the resulting sequence of the numbers must form an integer of minimum magnitude.

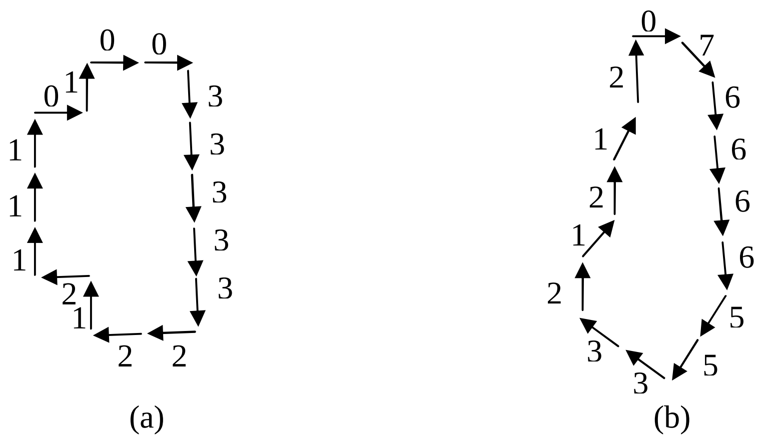


Figure (2.6) Chain code (a) 4-direction chain code; (b) 8-direction chain code

The boundary representation in figure 2.6(a) is the chain code (0033333221211101), and in figure 2.6(b) is the chain code (076666553321212) [Gonz87].

2.6 Features Extraction

Feature extractions refer to the process of finding a mapping that reduces the dimensionality of the patterns by extracting some numerical measurements form raw input pattern [Venu99].

The goal of the image analysis is to extract the useful information for solving application based problems. This is done by intelligently reducing the amount of image data with the tools we have explored (including the image segmentation and transforms). Now, we can consider extraction of features that can be useful for solving computer image problems. Image segmentation allows us to look at object features, and the image transforms provide us with features based on spatial frequency information-spectral feature [Umba98].

Exactly what we have to do with features will by application dependent. If we are working on a computer vision problem, the end goal may be the generation of a classification rule in order to identify objects. If we are working to develop a new image compression algorithm, we may want to determine what image data are important; the insignificant information can be compressed or eliminated completely [Umba98].

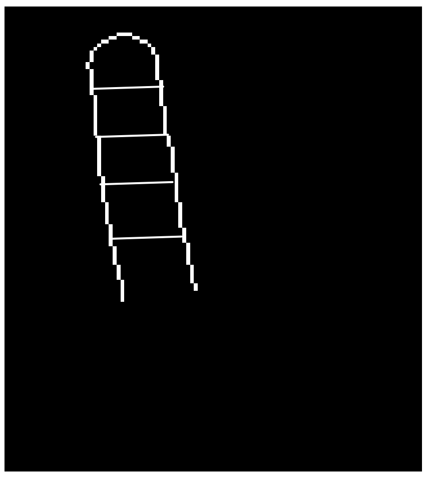
In the propose work, we have used two types of geometrical features and some non geometrical features.

2.6.1 Geometry Features

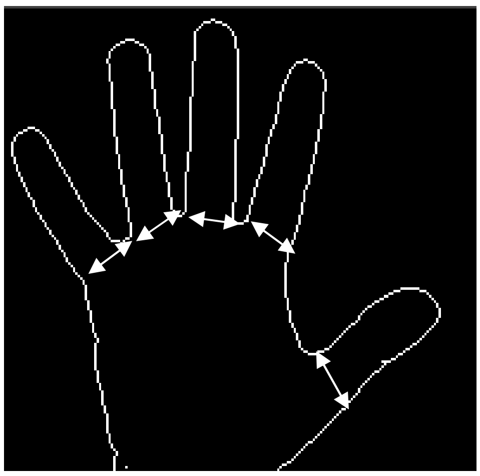
Hand geometry based identification system relies on the geometrical features of a human hand. Typical geometrical features include (length and width of the finger; width of the hand as well as the distance between joint points; etc) [Anil99].

The main geometrical features can be divided into the following categories [Hiba03]:

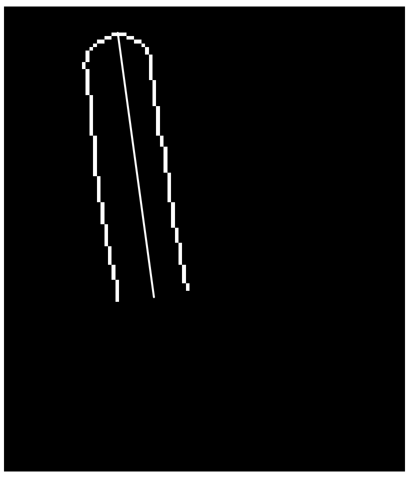
- Widths: the widths of each of the five fingers are measured at different finger's locations, as shown in figure (2.7) (a).
- Distances: the distances among the joint points in vertical and horizontal coordinates is measured, as shown in figure (2.7) (b).
- Lengths: the distance between the top point of the finger and the middle point of the distance between the two joints corresponding to that finger, this distance is named as the length of the finger, as shown in finger (2.7) (c).
- Hand span: the span of the hand is extracted by measuring the radius of the greatest circle drawn inside the palm, as shown in figure (2.7) (d).



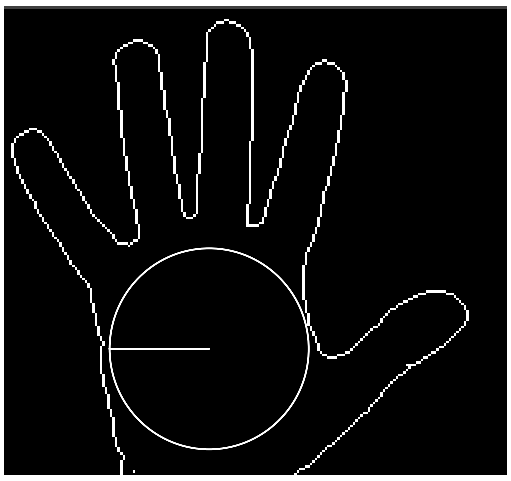
(a) Finger width



(b) Distance between joints



(c) Finger length



(d) Hand span

Figure (2.7) Hand geometrical features

2.6.2 Non-Geometry Features

In general, moments are set of parameters which describe the distribution of material (in image processing it is equivalent to brightness) from a reference point or an axis. The idea of using moments to construct the image feature vectors is one of the common utilized methods used today. Each moment order reflects different information for the same image. The determination of moments for image analysis is straightforward if we consider a binary or gray level image segment as a two dimensional density distribution function [Fadi04].

The shape of boundary segments (and of signature) can be described quantitatively by using moments. For 2D continuous function *f(x,y),* the moment of order (p + q) is defined as

$$m_{pq=}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{p} y^{q} f(x,y) dx\, dy \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(2.1)$$

For p, q = 0, 1, 2…

A "uniqueness" theorem states that if *f(x,y)* is piecewise continuous and has nonzero values only in a finite part of xy-plane, moments of all order exist, and the moment sequence (m_{pq}) is uniquely determined by *f(x,y).*

Conversely, (m_{pq}) uniquely determine *f(x,y).* The central moments can be expressed as [Gonz87]

$$\mu_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left(x - \bar{x}\right)^{p} \left(y - \bar{y}\right)^{q} f(x,y)\, dx\, dy \quad ,\ldots\ldots (2.2)$$

Where

$$\bar{x} = \frac{m_{10}}{m_{00}} \quad and \quad \bar{y} = \frac{m_{01}}{m_{00}} \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.3)$$

The above moments are called a central moments. There is another type of moments called a moment invariants, that type of moments is derived from the central moment.

Moment invariants can be used to establish feature vectors for shape representation from the moments of the shape. Moment invariants have been shown to be invariant to 2D shape transformations including rotation, scale and translation. Moment invariants can be calculated for the shape boundary or the shape silhouette. Silhouette based moments contain more information of the low frequency components or the global features of the shapes reflected by the boundary based moments which contain more information of the high frequency components or the fine details in shapes. Silhouette based moments are more popular for shape representation and have been used in the experiments performed by us [Atul98].

A set of seven moment invariants can be derived from the second and third moments [Gonz87]:

$$\phi_1 = \mu_{02} + \mu_{20} \quad ,\ldots\ldots(2.4)$$

$$\phi_2 = (\mu_{02} - \mu_{20})^2 + 4\mu_{11}^2 \quad ,\ldots\ldots(2.5)$$

$$\phi_3 = (\mu_{30} - 3\mu_{12})^2 + (3\mu_{11} - \mu_{03})^2 \quad ,\ldots\ldots(2.6)$$

$$\phi_4 = (\mu_{30} + \mu_{12})^2 + (\mu_{21} + \mu_{03})^2 \quad ,\ldots\ldots(2.7)$$

$$\phi_5 = (\mu_{30} + 3\mu_{12})(\mu_{30} + \mu_{12})\ [(\mu_{30} + \mu_{12})^2 - 3(\mu_{21} + \mu_{03})^2] + (3\mu_{21} - \mu_{03})(\mu_{21} + \mu_{03})[3(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2] \quad ,\ldots\ldots(2.8)$$

$$\phi_6 = (\mu_{02} + \mu_{20})[(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2] + 4\mu_{11}(\mu_{30} + \mu_{12})(\mu_{21} + \mu_{03}) \quad ,\ldots\ldots(2.9)$$

$$\phi_7 = (3\mu_{21} + \mu_{03})(\mu_{30} + \mu_{12})[(\mu_{30} + \mu_{12})^2 - 3(\mu_{21} + \mu_{03})^2] + (3\mu_{12} - \mu_{30})(\mu_{21} + \mu_{03})[3(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2] \quad ,\ldots(2.10)$$

2.7 Fuzzy Logic

The term "Fuzzy logic" has been used in the literature in two different senses. In a broad sense, fuzzy logic viewed as a system of concepts, principles, and methods for dealing with modes of reasoning that are approximate rather than exact [Geor97].

Fuzzy logic provides an inference morphology that enables approximate human reasoning capabilities to be applied to knowledge based systems. The theory of fuzzy logic provides a mathematical strength to capture the uncertainties associated with human cognitive processes, such as thinking and reasoning [Robe00].

Some of the essential characteristics of fuzzy logic relate to the following [Robe00]:

- In fuzzy logic, exact reasoning is viewed as limiting case of approximate reasoning.
- In fuzzy logic, everything is a matter of degree of decision maker.
- In fuzzy logic, knowledge is interpreted a collection of elastic or, equivalently, fuzzy constraint on a collection of variables.
- Inference is view as a process of propagation of elastic constraints.
- Any logic system can be fuzzified.

There are two main characteristics of fuzzy systems that give them better performance of specific applications [Robe00]:

- Fuzzy systems are suitable for uncertain or approximate reasoning, especially for the system with mathematical model that is difficult to drive.
- Fuzzy logic allows decision making with estimated values under incomplete or uncertain information.

2.7.1 Introduction to Fuzzy Set

The concept of the set is one of the most basic concepts in both logic and mathematics. By saying that it is basic, mean there are no more basic concepts in term of which a "Set" may be defined [Geor97].

In a classical set theory, a set is normally defined as a collection of objects, which can be finite, countable, or uncountable. An element x can be either belong to or not belong to a set A. if set A represent the concepts, for example, "male" and "female", a person is either a "male" or "female" because the concepts "male" and "female" are well defined without any ambiguity. This can be totally different if we consider the concepts" tall "and" short". A person have a height of 1.95 meter is commonly regarded as tall. But a person with height of 1.7 meter can be said to be tall for Asian with very high certainty, while for Europeans and American, this person is not considered to be tall. To handle these ambiguous concepts, we should extend the conventional set theory to fuzzy set theory [Rana99].

2.7.2 Membership Function

As already mentioned, one of the principal motivation for introducing fuzzy sets is to represent imprecise concepts. Because an individual's membership in a fuzzy set may admit some uncertainty, we say that its membership is a matter of degree. Accordingly, a person is a member of the set "tall people" to the degree to which he or she meets the operating concept of "tall". Alternatively, we can say that the degree of membership of individual in a fuzzy set expresses the degree of compatibilities of the concept represented by the fuzzy set. Each fuzzy set, $\mathbf{A}$, is defined in terms of a relevant universal set, $\mathbf{X}$, by a function analogous to the characteristic function. This function, called a *membership function,* assigns to each element x of X a number, $A(x)$, in the closed unit interval [a, b] that characterizes the degree of membership of x in A. Membership functions are thus functions of the form [Geor97]:

$$A: X \rightarrow [a, b] \quad ,\ldots\ldots\ldots\ldots\ldots(2.11)$$

In conventional fuzzy set classifiers, the range of each input variable is divided into several intervals. Then each interval is considered to be a fuzzy set and an associated membership function is defined. Thus, the input space is divided into several subregions that are parallel to input axis. For each subregion a fuzzy rule is defined; if the input is in the subregion, the input belongs to the class associated with the subregion. Then the degrees of membership of an unknown input for all the fuzzy sets are calculated and the input is classified into the class with the maximum degree of membership. Therefore, the membership functions directly influence the performance of the fuzzy classifier [Shig01].

There are many types of membership functions as (a) Triangular, (b) Trapezoidal, (c) Bell-shaped:

A. Triangular Membership Functions

The triangular membership function for the input x can be considered as that shown in the figure (2.8). At x=c (the center of the membership function) the degree of membership is 1, and it decreases as x moved away from c, and reaches 0 when x≤v or x≥V. The center c is not necessarily the middle point between v and V. The membership function m(x) for the input variable x can be defined as follows [Shig01]:

$$m(x) = \begin{cases} 0 & for\; x > V, \\ 1 - \dfrac{x-c}{V-c} & for\; c \le x \le V, \\ 1 - \dfrac{c-x}{c-v} & for\; c \ge x \ge v, \\ 0 & for\; x < v. \end{cases} \quad ,\ldots\ldots\ldots\ldots\ldots(2.12)$$

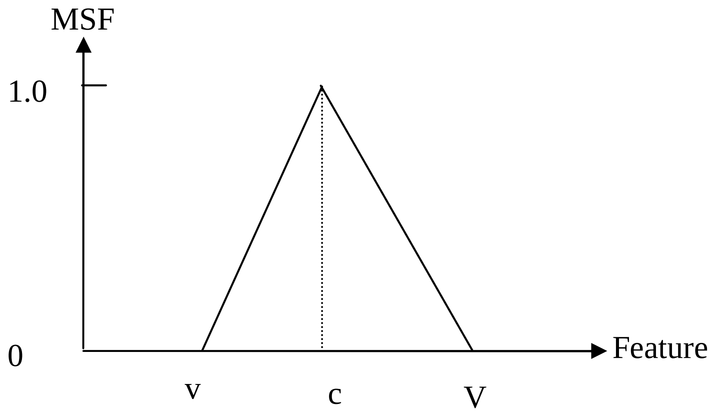


Figure (2.8) Triangular membership function

B. Trapezoidal Membership Functions

Another important class of membership functions has trapezoidal shape, which is illustrated by the generic graphical representation given in figure (2.9). Each function in this class is fully characterized by the four parameters, a, b, c, and d, via the generic form [Geor97]

$$m(x) = \begin{cases} \dfrac{(a-x)}{(a-b)} & when\, a \le x \le b \\ 1 & when\, b \le x \le c \\ \dfrac{(d-x)}{(d-c)} & when\, c \le x \le d \\ 0 & otherwise \end{cases} \quad ,\ldots\ldots\ldots\ldots\ldots(2.13)$$

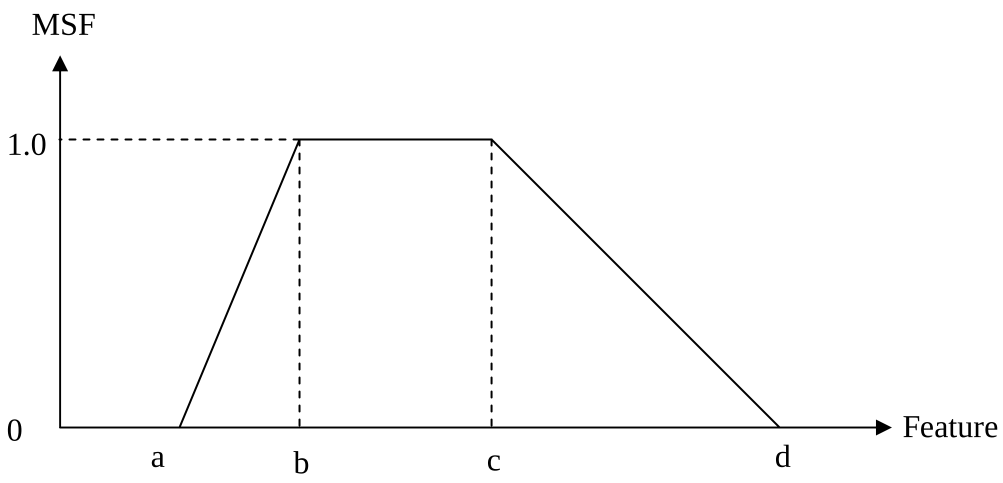


Figure (2.9) Trapezoidal membership function

C. Bell-Shape Membership Functions

Figure (2.10) shows the bell-shaped membership functions for input variable x. At the center **c``** the degree of membership is 1 and it decreases as x moves away from the center. The membership function m(x) is given by

$$m(x) = \exp\left(\frac{-(x-c)}{\alpha\sigma}^{2}\right) \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots..(2.14)$$

Where α a positive tuning parameter to tune the membership and function and σ is a variance of x around c [Shig01].

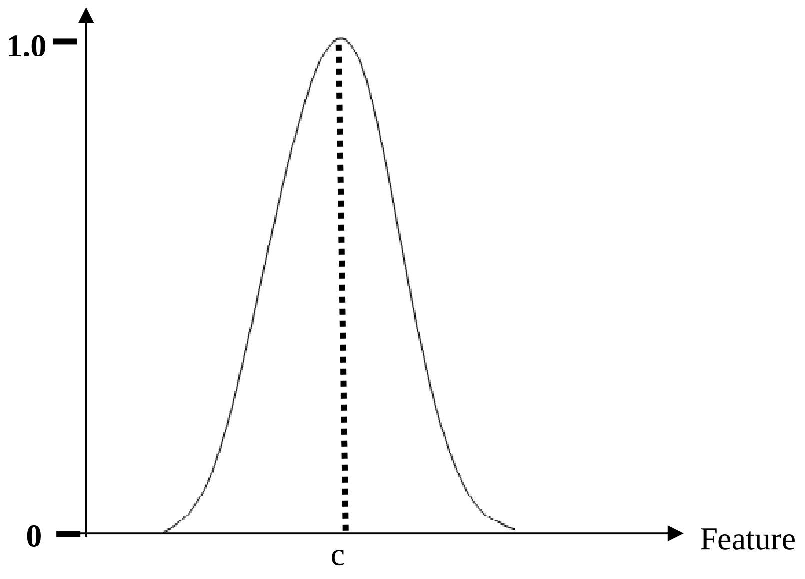


Figure (2.10) Bell- shape membership function

2.8 Artificial Neural Network (ANN)

NNs are statistical models of real world systems, which are built by tuning a set of parameters. These parameters, known as weights, form a mapping function between a set of a given values, known as inputs, to an associated set of values, known as outputs. Passing a set of examples of input-output pairs through the network and adjusting the weights carry out the process of tuning the weights the correct values training in order to minimize the error between the answer the network given and the desired output. Once the weights have been set, the model is able to produce an answer to test values that were not included in the training data. The models do not refer to the training data after they have been trained; in this sense they are functional summary of the training data [Ban01].

Create research work was devoted in the field of artificial neural network over the last years. The interesting thing about artificial neural networks is that they appear to be enormously useful in solving problems that have proved to be very difficult with conventional algorithms [Rana99].

In conventional computers, which are programmed to perform specific task, most neural networks must be taught, or trained. They can learn new associations, new functional dependencies and new patterns. Although computers outperform both biological and artificial neural systems for tasks based on precise and fast arithmetic operations, artificial neural network systems represent the promising new generation of information processing networks [Robe00].

2.9 The Basic Artificial Model

A neural network consists of a large number of simple processing elements called neurons, units, cells or nodes. Each neuron is connected to other neurons by means of directed communication links, each with an associated weight. These weights represent information being used by the net to solve problem. A node is generally an extremely simple device that has a number of input signals and a single output signal, as shows in figure (2.11).

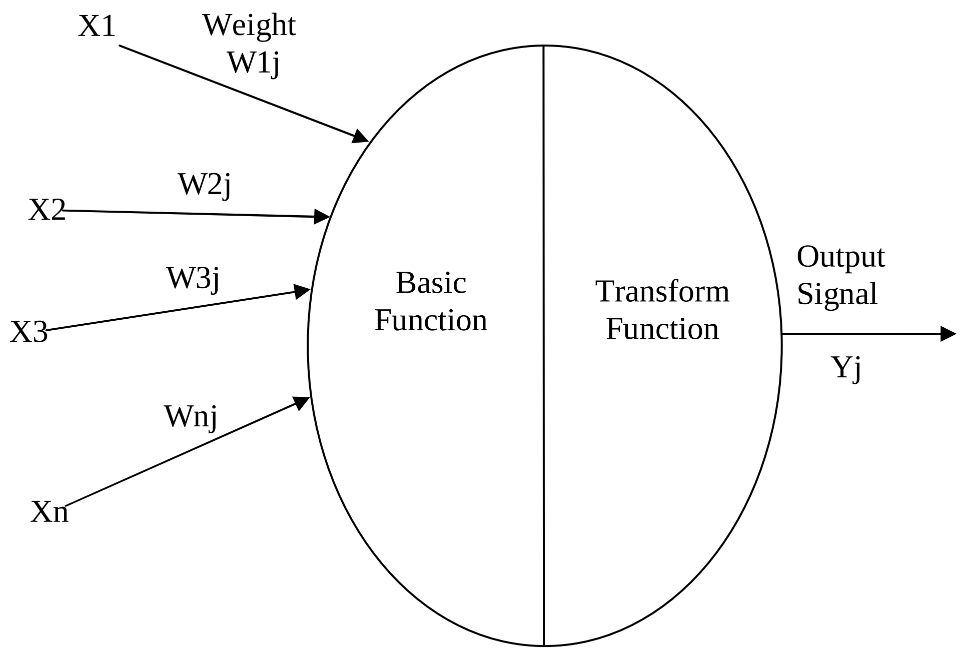


Figure (2.11) Neural network nods

For a node **J,** each input signal x_i (i=1, 2... n) is assigned a relative weight W_{ij}. From the weighted total input the node computes a single output signal Y_j. The following three steps will take place when each node is activated or processed [Rana99]:

1. A number of inputs (either from original data, or from the output of the other neurons in NN) are received.
2. The mapping of the weighted inputs is calculated by using the basic function.
3. The result of the basis function is transformed by an activation function. The transformed result (output signal of the node Y_j) may be sent to other nodes**.**

2.10 Types of Neural Network Classifiers

There are two types of classifiers, supervised and unsupervised, as show in the following:

1. **Supervised neural network classifiers**: supervised learning algorithms include a special case of reinforcement learning, where a given pattern is identified as a member of already known classes. Multilayer feed forward have emerged as a popular mode for pattern classification tasks. The most important attribute of multilayer feed forward is that it can learn a mapping of any complexity [Patt96].
2. **Unsupervised neural network classifier**s: in the unsupervised methods, the network must discover for itself the patterns, features, correlation or categories in the input data and code for them in the output. The connection weights are modified through different iterations of network operation, depending on the winner neurons; this process is called *self-organization.* The proper clusters are formed by discovering the similarities and dissimilarities among the objects. Kohonen's SOM is an example of such net. The final step in the unsupervised classification is to decide which cluster represents each physical class [Ban01].

2.11 Fuzzy Neuron

Hybrid systems combining fuzzy logic, neural network, genetic algorithms, and expert systems are proving their effectiveness in a wide variety of real world problems. Every intelligent technique has particular computation properties (e.g. abilities to learning, explanation of decisions) that make them suited for particular problems and not for others. While neural networks are good at recognizing patterns, they are not good at explaining how they reach their decisions. Fuzzy logic systems, which can reason with imprecise information, are good at explaining their decision but they can't automatically acquire the rules they use to make those decisions. These limitations have been a central driving force behind the creation of intelligent hybrid systems where two or more techniques are combined in a manner that overcomes the limitations of individual techniques [Robe00].

A fuzzy neuron is designed to function in much the same way as a non-fuzzy neuron, expect that it reflects the fuzzy nature of a neuron, and has the ability to cope with fuzzy information. The basic structure of a fuzzy neuron is described in figure (2.12). Input to the fuzzy neuron are fuzzy set $(X_1, X_2, ..., X_n)$ in the universe of discourse $(U_1, U_2, ..., U_n)$ respectively. These fuzzy sets may be labeled by such linguistic terms as *high, large, warm, etc.* The fuzzy inputs are then *weighted* in synapses in a much different way from that in non-fuzzy case. The weighted fuzzy inputs are then aggregated not by the summation but by the fuzzy aggregation operations (fuzzy union, weighted mean or intersection).

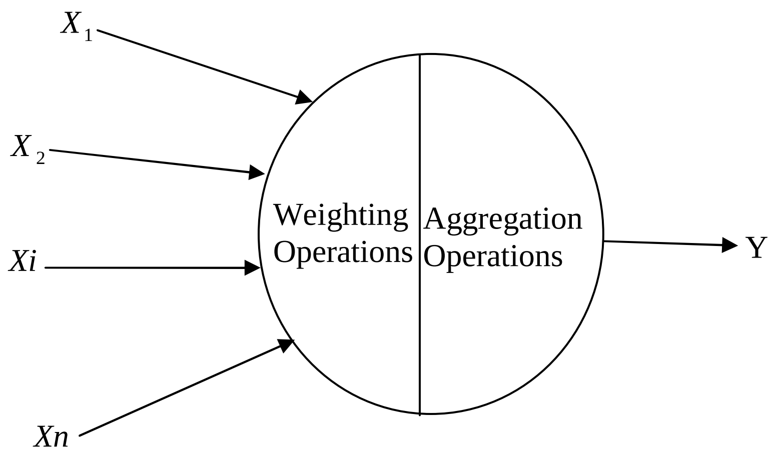


Figure (2.12) A fuzzy neuron model

An important difference between the computational aspects of a non-fuzzy neuron and that of a fuzzy neuron is the definition of mathematical operations. The mathematical operation in a non-fuzzy neuron may be defined in terms of confluence operation (usually, an inner product) between adjustable synaptic weights and neural inputs. Any learning and adaptation occurring within the neuron involves modifying these synaptic weights. In a fuzzy neuron, a two-dimensional fuzzy relation between the synaptic weights and neural inputs represents the synaptic connection. In this situation, learning involves changing the two-dimensional relation surface at each synapse [Ban01].

2.12 Self Organization Map (Kohonen Neural Network)

Kohonen's self organization map (SOM) first proposed by Kohonen is an unsupervised learning method that automatically models the features in the input data reflects these features on topological maps. The resulting maps from local neighborhoods that act as feature classifiers on the set of input patterns are mapped onto close neighborhoods in the maps [Ban01].

A Kohonen layer is composed of neurons that compete with other. Like in adaptive resource theory, the Kohonen SOM is another case of using a winner-take-all strategy. The inputs are fed into each of the neurons in the Kohonen layer (from the output layer). Each neuron determines its output according to a weighted formula:

$$Output_j = \sum w_{ij} x_i \quad ,\ldots\ldots\ldots\ldots\ldots..(2.15)$$

Where i=1 to number of input nodes

The neuron with largest output is the winner. The Kohonen network has two layers, an input layer and output layer as shown in figure (2.13). The input layer is a size determined by the user and must match the size of each row (pattern) in the input data [vall97].

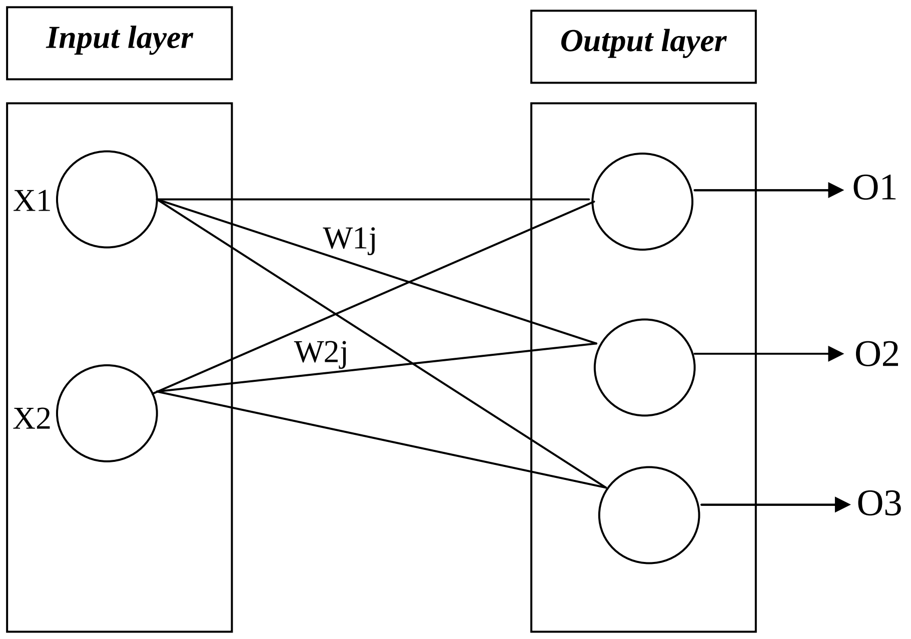


Figure (2.13) Kohonen Network

The winning neuron is trained according to the following equation:

$$w_{ij}^{t+1} = w_{ij}^{t} + \alpha(x_i^t - w_{ij}^t) \text{,………….. (2.16)}$$

Where α is the learning parameter or gain $(0 < \alpha \leq 1)$ which should be constant. Two factors affect the clustering operation in Kohonen neural net, these factors are:

1. The weight initialization (different weight initialization may lead to different result).
2. The learning rate value (α)

Weights could be initializing using one of two ways:

1. Random initialization: in this case, α starts relatively high ($0.1 \leq \alpha \leq 0.8$) and decreases gradually to produce finer weight adjustments (e.g. initially α =0.8, then reduced gradually after a specified number of iterations until α =0.1).

2. Convex combination: it is another method used to select initial weights suggested by Hecht-Nielsen. In this method, all weight vectors are initialized at the same value [Venu99]:

$$w_i^0 = \frac{1}{\sqrt{n}}[1,1,1,........,1]^T \quad for\ i = 1,...,n \quad ,……(2.17)$$

الأهداء

الى من أنحدروا كألسيل من بحر النبوة...

الى من علموا الاحرار حريتهم...

الى من هم حقيقة الارادة و العقيدة...

الى من أحيوا قيم السماء و بذلوا أنفسهم و أولأدهم في كربلاء...

قمر بني هاشم أبي الفضل العباس (ع).

و

سيد شباب أهل الجنة أبي عبدالله الحسين (ع).

Republic of Iraq
Al-Nahrain University
College of Science

Hand Identification Using Fuzzy-Neural

A THESIS
SUBMITTED TO THE
COLLEGE OF SCIENCE, Al-NAHRAIN UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE IN
COMPUTER SCIENCE

By

Ali Muhsen Mohammed

(B.Sc. 2002)

SUPERVISORS

Dr. Loay A. George **Dr. Ban N. Al-kallak**

February 2005 **Muharam 1426**

بِسۡمِ ٱللَّهِ ٱلرَّحۡمَٰنِ ٱلرَّحِيمِ

وَمَا تَوۡفِيقِيٓ إِلَّا بِٱللَّهِۚ عَلَيۡهِ تَوَكَّلۡتُ وَإِلَيۡهِ أُنِيبُ ﴿٨٨﴾

صدق الله العلي العظيم

هود-٨٨

List of Abbreviations

PIN	Person Identification Number
ATM	Asynchronous Transfer Mode
BMP	Bite Map
PC	Personal Computer
ID	Identity
FL	Fuzzy Logic
NN	Neural Network
ANN	Artificial Neural Network
SOM	Self Organization Map
MSF	Member Ship Function
FSOM	Fuzzy Self Organization Map

References

- [Alex98] Alexandra L.N wong, Pengcheng shi, "peg-free Hand Geometry Recognition using Hierarchical Geometry and Shape Matching", Hong Kong university,1998.
- [Anil99] Anil k. Join, Arun Ross, Sharath Pankanti, "A prototype Hand Geometry-based verification system", in second international conference on Audio and Video-based Biometric person Authentication, Washington D.C., 1999.
- [Atul98] Atul Sajjanhar, Guojun Lu, " A comparison of Techniqued for Shape Retrieval", Gippsland school of computing anf information technology, Monash university,1998 E-mail:guojun.Lu@fcit.monash.edu.qu
- [Ban01] Ban Nadeem, "Fuzzy rule extraction", PH.D thesis, university of Technology, 2001.
- [Brya00] Bryan S. Morse. "Data structure for image Analysis", Brigham Young University, 2000. www.homepages.inf.ed.ac.uk/rbf/cvonline/local_copies/morse/data_staructure.pdf
- [Cenk02] Cenker Oden, Aytul Ercil, Burak Buke, "Combining Implicit polynomials and Geometric features for hand recognition", Bogazici University, 2002, E-mail: {Oden,bukeb}@boun.edu.tr.
- [Fadi04] Fadil Salman," Adaptive Fractal image compression", phd thesis, Al-Rashed collage, 2004
- [Geor97]George J. Klir, Uet st. Clain, Bo Yuan. " Fuzzy set Theory Foundation and Application", printice hall, 1997.
- [Geri03] Gerik Alexander," Biometric in Access Control", Bergdata Biometric GmbH, Bonn, Germany, 2003.

Email:graevenitz@bergdata.com.

- [Gonz87] Rafael C. Gonzalez, Richard E. woods," Digital Image Processing", Addison-Wesley, 1987.
- [Gonz74] Tou, T.J and Gonzalez, R.C "Pattern recognition Principles", Reading Mass., Addison-Wesley publishing company 1974.
- [Hiba03] Hiba Zuhair," Hand Geometry-based Identity Authentication Method", Msc thesis, collage of science, Al-Nhrain university, 2003.
- [Inte01] International Biometric Group 2001, "Biometrics Explained", New York, www.biometricgroup.com.
- [Mila99] Milan Markovic, Predrag Prjnovic, Srdjah Stankovic, " On Two- Dimensional shape recognition using moment inveriants and Ar models", 1999, Facta universitaties, Electronic and energetics.
- [Patt96] D.W. Patterson, "Artificial Neural Network Theory and application", Printice Hell, 1996.
- [Rana99] Rana Saadi," Image segmentation Using Fuzzy and Neural networks", Msc thesis, collage of science, Al-Nhrain university, 1999.
- [Robe00] Robert Fuller' "Introduction to Neuro-fuzzy system", Physica-herlag Heidelberg, 2000.
- [Shig01] Shigeo Abe," Pattern Classification Neuro-Fuzzy Method and Their comparison", Springer-Verlag London Limited 2001.
- [Simo01] Simon Liu, Mark Silverman "A practical Guide to Biometric Security Technology", 2001, E-mail: Simo_liu@nlm.nih.gov.

- [Slob98] Slobodan Ribaric, Damir Ribaric, Nikola Pavesic "A Biometric Identification system Based on the Fusion of hand and palm Features", E-mail:Slobodan.ribaric@fer.hr
- [Toma00]Tomas Braunl ,S. feyre, W. Refpf, M. Reinhardt," Parallel image processing", Springer-verlag Berlin Heidelbery New York 2000.
- [Umba98] Scott E. Umbaugh, "Computer Vision and Image Processing", Printic-Hell, 1998.
- [Vall97] Valluru Rao & Hayagriva Rao, "C++ Neural Networks and Fuzzy logic", Second edition ,BpB publications, 1997.
- [Vasa98] Vasant Monohar," Digital image processing-CAP5400 Basic Threshold and Smoothing ",computer science & Engineering Department, university of South Florieda, 1998 www.csee.usf.edu/~vmanohar/assignment1.html
- [Venu99] Venus W. Samawi," An Investigation into the use of neural networks in texture classification ", phd thesis, collage of science, Al-Nhrain university, 1999.
- [Zden00] Zdenek Riha, "biometric Authentication systems", Faculty of information Masaryk University, 2000.

Supervisor Certification

We certify that this thesis was prepared under our supervision at the Department of Computer Science/ Collage of Science / Al-Nhrain University, By **Ali Muhsen Mohammed** as partial fulfillment of the requirements of the degree of Master of Science in Computer Science.

Signature:	Signature:
Name: **Dr. Loay A. George**	Name: **Dr. Ban N. Al-Kallak**
Title: **Senior Researcher**	Title: **Lecturer**
Date: **/ /2005**	Date**: / /2005**

In view of the available recommendation, I forward this thesis for debate by the examination committee.

Signature:
Name: **Dr. Taha S. Bashaga**
Title: **Head of the Department of Computer Science, Al-Nhrain University**
Date: **/ /2005**

Table of Contents