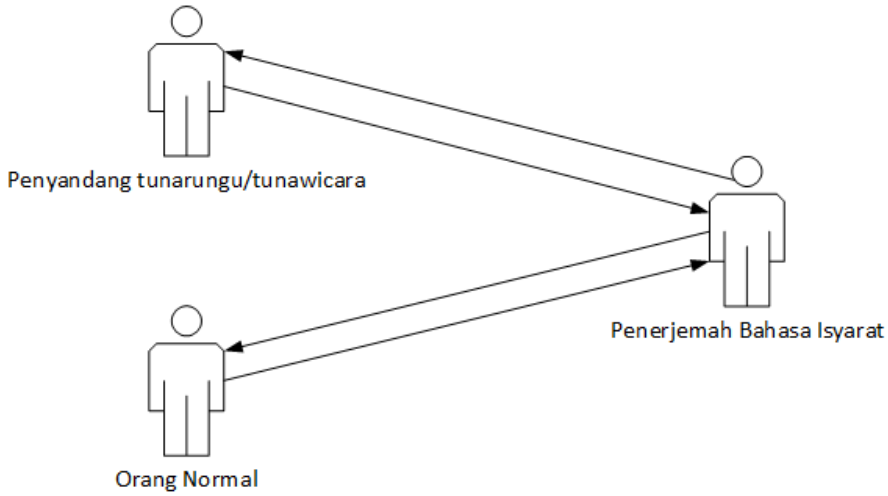
BAB 3
ANALISIS DAN PERANCANGAN

3.1 Analisis Masalah

Adapun alur komunikasi yang sedang berjalan antara penyandang tunarungu dengan orang normal dapat dilihat pada Gambar 3.1 berikut.



Gambar 3.1 Alur Komunikasi

Pada penelitian sebelumnya [3] menggunakan metode ekstraksi ciri *viola and jones*, beberapa huruf bahasa isyarat belum dapat dikenali dengan baik dan belum dapat mengenali bahasa isyarat dinamis. Penelitian ini menggunakan data uji yang sebanyak 80 citra yang terdiri dari 78 data termasuk bahasa isyarat dan 2 data tidak termasuk bahasa isyarat. Hasil pengenalan yang dihasilkan disebutkan 44 terklasifikasi dengan baik dan 36 terklasifikasi tidak tepat [1]. Berdasarkan hal tersebut terdapat permasalahan masih rendahnya akurasi pengenalan bahasa isyarat statis dan dinamis.

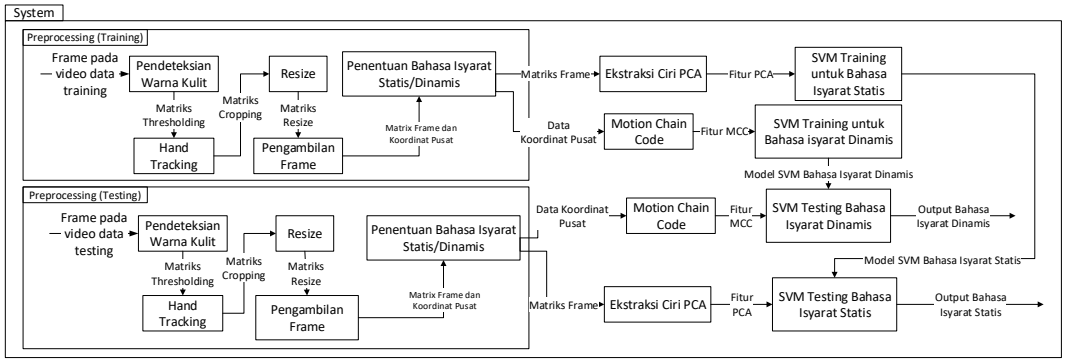
Pada penelitian [3] menggunakan metode ekstraksi ciri PCA dan neural *network* dengan data resize berdimensi 32x32 piksel untuk melakukan pengenalan bahasa isyarat statis mendapatka akurasi sebesar 86%. Pada penelitian lain menggunakan metode PCA dan *Haar-Like Feature* mendapatkan akurasi sebesar 80,42% [4] untuk mengenali huruf bahasa isyarat dari A-Z, kecuali huruf I dan J

karena lebih dari satu gerakan. Kemudian penelitian lain yang menggunakan fitur dari metode *Motion Chain Code* (MCC) untuk mendeskripsikan pola gerakan tangan bahasa isyarat dinamis dalam bahasa Arab mendapatkan tingkat akurasi sebesar 88% [5] dengan data latih sebanyak 1045 video dan data testing sebanyak 243 video.

Pada penelitian ini akan mengimplementasikan metode ekstraksi ciri PCA sebagai ciri untuk bahasa isyarat statis dan metode MCC digunakan sebagai fitur untuk bahasa isyarat dinamis. Diharapkan dengan fitur dari metode PCA dan MCC dapat mengenali huruf bahasa isyarat isyarat statis dan dinamis dengan lebih baik.

3.2 Analisis Sistem

Sistem yang akan dibangun adalah suatu aplikasi berbasis dekstop yang menggunakan teknologi *image processing* dengan data masukan berupa video secara otomatis mengklasifikasikan gerakan bahasa isyarat dan menampilkan hasil klasifikasi berupa teks. Pengenalan bahasa isyarat akan mengkombinasikan algoritma MCC untuk isyarat dinamis dan metode ekstraksi ciri PCA untukgerakan bahasa isyarat statis, dan metode klasifikasi menggunakan *Multiclass* SVM. Pada Gambar 3.2 dapat dilihat alur sistem yang akan dibangun untuk pengenalan bahasa isyarat.



Gambar 3.2 Arsitektur Sistem

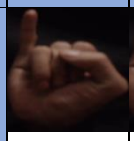
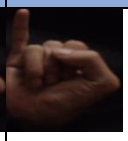
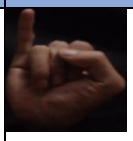
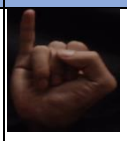
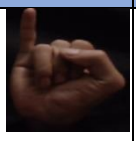
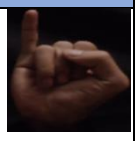
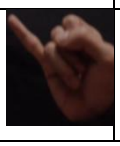
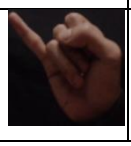
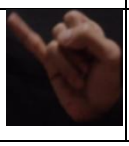
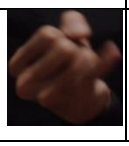
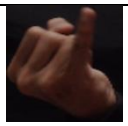
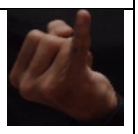
Berdasarkan block diagram pada Gambar 3.2, terdapat dua bagian yaitu pelatihan untuk bahasa isyarat statis, pelatihan untuk bahasa isyarat dinamis dan pengujian.

1. Pada bagian pelatihan tahap pertama yang dilakukan adalah menginput frame pada video data latih bahasa isyarat. Kemudian akan dilakukan *preprocessing* terhadap setiap *frame* pada video data latih bahasa isyarat yang telah diinputkan. Di dalam *preprocessing* dilakukan beberapa tahapan, yaitu pendeteksian kulit, *hand tracking*, *resizing,* pengambilan *frame*, dan penentuan bahasa isyarat statis dan dinamis. Proses pengambilan *frame* dilakukan setelah semua *frame* dari video masukan dilakukan pendeteksian warna kulit sampai *resizing.* Penentuan bahasa isyarat statis atau dinamis yang dilakukan adalah jika gerakan bahasa isyarat termasuk statis maka akan diambil data *frame* dan jika gerakan bahasa isyarat termasuk dinamis akan diambil data koordinat pusatnya saja. Setelah *preprocessing* terhadap setiap *frame* pada video data latih bahasa isyarat, kemudian akan dilakukan proses ekstraksi ciri PCA untuk bahasa isyarat statis Kemudian nilai fitur dari PCA akan digunakan untuk pelatihan SVM untuk bahasa isyarat statis sesuai dengan data video yang diinputkan. Kemudian untuk bahasa isyarat dinamis akan dilakukan proses MCC. Nilai fitur dari MCC akan digunakan untuk pelatihan SVM untuk bahasa isyarat dinamis.
2. Pada bagian pengujian akan dilakukan *preprocessing*, ekstraksi ciri PCA dilakukan jika hasil dari penentuan jenis bahasa isyarat pada proses pengambilan *frame* adalah gerakan bahasa isyarat statis. Kemudian dilakukan pengujian menggunakan SVM *Multiclass* yang telah dilatih dengan bahasa isyarat statis. Jika hasil penentuan jenis bahasa isyarat adalah gerakan bahasa isyarat dinamis, maka dilakukan proses perhitungan MCC. Kemudian dilakukan pengujian menggunakan SVM yang sudah dilatih dengan bahasa isyarat dinamis. Hasil dari pengujian akan menghasilkan keluaran huruf berupa teks.

3.3 Analisis Data Masukan

Pada penelitian ini menggunakan webcam sebagai media untuk pengambilan data. *Frame* dataset yang diambil berupa video sampelnya menggunakan 30 *frame per second* dan ukuran *frame* adalah 320x240 piksel. *Frame* dataset yang diambil adalah bahasa isyarat huruf A-Z sesuai dengan kamus SIBI. Jarak antara *webcam* dan peraga bahasa isyarat saat pengambilan video adalah 80-90cm. Dari gerakan isyarat dari awal sampai akhir akan diambil sebanyak 20 *frame*. Setiap huruf memiliki 40 video data latih sehingga dibutuhkan 26 * 40 = 1040 video. Dataset yang diambil setiap huruf memiliki 20 video untuk data uji, sehingga dibutuhkan 20 * 26 huruf =520 video. Total video yang dibutuhkan dalam penelitian ini adalah sebanyak 1560 video dalam bentuk file .mp4. Pada Tabel 3.1 dapat dilihat contoh data masukkan sebanyak 60 frame pada video berdurasi 2 detik yang digunakan untuk pengenalan bahasa isyarat.

Tabel 3.1 Contoh Data Masukan

Frame	1	2	3	..	58	59	60
I				…			
J				…			

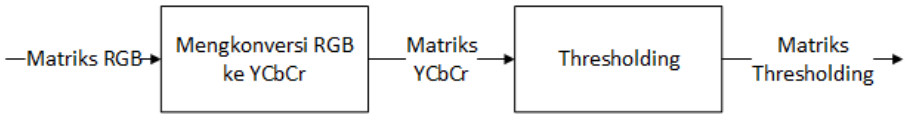
3.4 Analisis Proses *Preprocessing*

Analisis *preprocessing* bertujuan untuk mengolah data masukan sebelum diolah ke tahapan selanjutnya. Pada penelitian ini *preprocessing* terdiri dari beberapa tahapan yaitu pendeteksian warna kulit, *hand tracking* (penentuan region tangan), *resize*, dan pengambilan *frame*.

3.4.1 Pendeteksian Warna Kulit

Pada tahapan ini masukan berupa *frame* pada video akan dilakukan proses pendeteksian warna kulit. Pendeteksian warna kulit bertujuan untuk memisahkan region objek dalam citra berdasarkan pada perbedaan warna. Objek dengan warna

yang memiliki warna kulit dipisahkan dengan warna lainnya. Alur proses pendeteksian warna kulit dapat dilihat pada Gambar 3.3 berikut.



Gambar 3.3 Proses Pendeteksian Warna Kulit

Berdasarkan Gambar 3.3 dapat dijelaskan, Proses pendeteksian warna kulit pada penelitian ini menggunakan algoritma YCbCr bertujuan untuk memisahkan antara komponen warna kulit dengan komponen warna lainnya. Kemudian dari citra YcbCr dilakukan proses *thresholding***.** Pada Gambar 3.4 dapat dilihat potongan *frame* huruf I berukuran 320x240 piksel sebelum dilakukan proses pendeteksian warna kulit.



Gambar 3.4 Frame Huruf I

Selanjutnya mengubah *frame* pada Gambar 3.4 kedalam bentuk matriks RGB. Bentuk matriks RGB akan diubah ukurannya menjadi 12x16 untuk kebutuhan analisis *preprocessing*. Dalam penerapannya ukuran matriks RGB berukuran 320x240. Pada Tabel 3.2 dapat dilihat matriks RGB berukuran 12x16.

Tabel 3.2 Matriks RGB

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	R	7	7	8	15	22	25	33	33	29	25	25	24	23	25	25	26
	G	4	4	5	14	17	22	29	29	25	23	20	19	20	23	23	21
	B	7	7	8	16	21	27	34	33	28	25	24	23	24	26	26	24
2	R	20	13	7	7	7	7	8	15	23	27	26	26	23	24	16	13
	G	17	10	4	4	4	4	5	10	20	22	20	23	18	21	14	10
	B	20	13	7	7	7	7	8	14	22	28	26	26	22	24	15	13
3	R	25	25	19	15	8	7	7	7	7	7	7	14	23	17	19	14
	G	21	21	16	12	5	4	4	4	4	4	4	11	18	14	14	11
	B	24	25	20	15	8	7	7	7	7	7	7	14	22	17	18	15
4	R	8	13	22	28	21	15	159	8	7	7	7	6	6	7	10	12
	G	5	10	19	24	18	12	123	5	4	4	4	3	3	4	7	9
	B	8	13	22	28	21	16	92	8	7	7	7	6	6	7	10	13
5	R	7	8	9	12	18	24	63	123	16	12	8	7	7	6	6	6
	G	4	5	6	9	16	20	32	98	12	9	5	4	4	3	3	3
	B	7	8	9	12	18	23	25	122	16	12	8	7	7	6	6	6

6	R	9	7	8	8	9	11	63	131	123	159	136	18	15	8	7	7
	G	6	4	5	5	6	9	38	93	86	130	104	15	10	5	4	4
	B	9	7	8	8	9	10	27	79	72	118	95	18	14	8	7	7
7	R	18	15	9	8	8	85	38	92	137	127	60	21	15	16	17	16
	G	13	12	6	5	5	59	23	55	97	92	32	14	10	13	13	14
	B	17	15	9	8	8	50	18	43	118	132	25	17	14	18	17	15
8	R	13	15	14	14	10	70	63	31	76	85	60	61	10	8	7	7
	G	10	12	11	12	7	41	36	15	50	55	33	33	7	5	4	4
	B	13	15	14	13	10	33	30	12	40	45	27	25	10	8	7	7
9	R	14	10	11	12	17	30	63	62	47	41	12	9	8	8	9	11
	G	11	7	9	9	14	15	32	35	25	22	5	6	5	5	6	8
	B	14	10	11	12	17	16	26	30	19	18	9	9	8	8	9	11
10	R	16	12	12	10	14	12	18	26	27	8	9	12	11	9	8	8
	G	12	9	9	7	11	10	8	15	14	4	6	9	8	6	5	5
	B	15	12	13	10	14	11	10	13	13	7	9	12	11	9	8	8
11	R	15	16	13	13	10	7	6	7	7	7	13	11	13	13	10	12
	G	11	13	10	10	7	4	3	4	4	4	10	8	10	11	7	9
	B	15	16	13	13	10	7	6	7	7	7	13	11	13	12	10	14
12	R	11	13	11	11	10	8	6	6	7	7	9	11	12	14	16	16
	G	8	10	8	8	7	5	3	3	4	4	5	8	9	11	10	13
	B	11	13	11	11	10	8	6	6	7	7	8	11	12	14	14	16

Langkah selanjutnya adalah melakukan konversi dari RGB ke ruang warna YcbCr menggunakan persamaan (2.1) berikut.

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 0{,}299 & 0{,}587 & 0{,}144 \\ -0{,}169 & -0{,}331 & 0{,}500 \\ 0{,}500 & -0{,}419 & -0{,}081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Contoh nilai RGB pada piksel ke (1,1), yaitu nilai R = 7, G = 4 dan B =7. Maka untuk perhitungan YCbCr pada pixel (1,1) yaitu:

$$Y = (0{,}299 * R) + (0{,}587 * G) + (0{,}144 * B) + 16$$
$$= (0{,}299 * 7) + (0{,}587 * 4) + (0{,}144 * 7) + 16$$
$$= 2{,}093 + 2{,}348 + 1{,}008 + 16$$
$$= 21{,}4$$

$$Cb = (-0{,}169 * R) + (-0{,}331 * G) + (0{,}500 * B) + 128$$
$$= (-0{,}169 * 7) + (-0{,}331 * 4) + (0{,}500 * 7) + 128$$
$$= -1{,}183 + (-1{,}324) + 3{,}5 + 128$$
$$= 128{,}9$$

$$Cr = (0{,}500 * R) + (-0{,}419 * G) + (-0{,}081 * B) + 128$$
$$= (0{,}500 * 7) + (-0{,}419 * 4) + (-0{,}081 * 7) + 128$$
$$= 3{,}5 + (-1{,}676) + (-0{,}567) + 128$$
$$= 129{,}2$$

Maka hasil perhitungan dari Y, Cb, dan Cr pada piksel ke (1, 1) didapat nilai Y= 21,4, Cb= 128,9, dan Cr= 129,2. Hasil dari konversi matriks RGB ke matriks YCbCr dapat dilihat pada Tabel 3.3 berikut.

Tabel 3.3 Matriks YCbCr

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	Y	21.4	21.44	22.47	31.00	35.58	40.27	47.78	47.6	43.37	40.57	38.67	37.64	38.0	40.72	40.7	39.5
	Cb	128.9	128.1	128.9	128.8	129.1	129.9	129.8	129.3	128.8	128.6	129.1	129.1	129.4	129.1	129.1	128.6
	Cr	129.2	29.2	129.2	128.3	130.1	129.0	129.5	129.6	129.7	128.8	130.1	130.1	129.1	128.7	128.7	130.2
2	Y	34.83	27.62	21.44	21.44	21.44	21.4	22.47	28.37	37.78	41.01	39.25	41.01	36.61	38.95	31.16	27.62
	Cb	128.9	128.9	128.9	128.9	128.9	128.9	128.9	129.1	128.4	130.1	129.9	128.9	129.1	128.9	128.1	128.9
	Cr	129.2	129.2	129.2	129.2	129.2	129.2	129.2	130.1	129.3	130.0	130.5	129.2	130.1	129.2	128.9	129.2
3	Y	39.25	39.40	33.95	29.68	22.47	21.44	21.44	21.44	21.44	21.44	21.44	28.65	36.61	31.74	32.49	28.8
	Cb	128.8	129.3	129.4	128.9	128.9	128.	128.3	128.9	128.9	128.9	128.3	128.9	129.1	128.9	129.1	129.4
	Cr	129.7	129.6	129.1	129.2	129.2	129.2	129.2	129.2	129.2	129.2	129.2	129.2	130.1	129.2	130.1	129.1
4	Y	22.47	27.61	36.89	42.49	35.86	29.83	90.02	22.47	21.44	21.44	21.4	20.41	20.41	21.44	24.53	26.7
	Cb	128.9	28.99	128.9	129.3	128.9	129.4	120.4	128.9	128.9	128.9	128.9	128.9	128.9	128.9	128.9	129.4
	Cr	129.2	129.2	129.2	129.6	129.2	129.1	146.2	129.2	129.2	129.2	129.2	129.2	129.2	129.2	129.2	129.1
5	Y	21.44	22.47	23.50	26.59	33.36	38.22	57.22	106.6	30.13	26.59	22.4	21.44	21.44	20.41	20.41	20.4
	Cb	128.9	128.9	128.9	128.9	128.6	128.8	119.2	129.1	129.3	128.9	128.9	128.9	128.9	128.9	128.9	128.3
	Cr	129.2	129.2	129.2	129.2	128.8	129.7	144.0	130.1	129.6	129.2	129.2	129.2	129.2	129.2	129.2	129.2
6	Y	23.50	21.44	22.47	22.47	23.50	26.01	61.03	121.1	113.6	156.8	131.3	32.77	28.37	22.47	21.44	21.4
	Cb	128.9	128.9	128.9	128.9	128.9	128.1	118.2	114.5	114.7	117.0	118	128.9	129.1	128.9	128.9	128.9
	Cr	129.2	129.2	129.2	129.2	129.2	128.9	141.3	148.1	147.6	143.4	144.7	129.2	130.1	129.2	129.2	129.2
7	Y	31.46	29.68	23.50	22.47	22.47	83.24	43.45	81.98	26.34	51.44	56.32	32.94	28.37	31.00	31.16	31.1
	Cb	129.1	128.9	128.9	128.9	128.9	119.1	122.9	115.7	126.8	120.7	119.7	128.3	129.1	129.9	129.3	128.1
	Cr	130.1	129.2	129.2	129.2	129.2	141.7	135.9	147.2	132.9	139.5	142.5	131.2	130.1	129.0	129.6	128.9
8	Y	27.62	29.68	28.65	29.10	24.53	65.74	60.28	35.80	73.8	80.18	57.19	57.21	24.53	22.47	21.49	21.4
	Cb	128.9	128.9	128.9	128.1	128.9	119.0	120.4	123.7	118.6	117.9	120.4	119.2	128.9	128.9	128.9	128.9
	Cr	129.2	129.2	129.2	128.9	129.2	143.1	141.9	136.2	141.8	143.8	141.9	142.6	129.2	129.2	129.2	129.5
9	Y	28.65	24.53	26.15	26.59	31.74	36.07	57.36	59.40	47.46	43.76	23.81	23.50	22.47	22.47	23.50	25.5
	Cb	128.9	128.9	128.6	128.9	128.9	125.9	119.7	120.9	121.2	122.7	128.8	128.9	128.9	128.3	128.3	128.9
	Cr	129.2	129.2	128.8	129.2	129.2	135.4	143.9	141.9	139.4	137.8	131.1	129.2	129.2	129.2	129.2	129.2
10	Y	29.98	26.59	26.74	24.53	28.65	27.04	27.51	34.45	34.16	21.74	23.50	26.59	25.56	23.50	22.47	22.4
	Cb	128.8	128.9	129.4	128.9	128.9	128.1	127.3	125.1	125.3	128.8	128.3	128.9	128.3	128.3	128.9	128.9
	Cr	129.7	129.2	129.1	129.2	129.2	128.9	132.8	133.6	134.1	129.7	129.2	129.2	129.2	129.2	129.2	129.2
11	Y	29.10	30.71	27.62	27.62	24.53	21.44	20.41	21.44	21.44	21.44	27.6	25.56	27.62	28.07	24.53	26.7
	Cb	129.3	128.9	128.9	128.9	128.9	128.9	128.9	128.9	128.9	128.9	128.9	128.9	128.9	128.1	128.9	129.9
	Cr	129.6	129.2	129.2	129.2	129.2	129.2	129.2	129.2	129.2	129.2	129.2	129.2	129.2	128.9	129.2	129
12	Y	25.56	27.62	25.56	25.56	24.53	22.47	20.41	20.41	21.44	21.44	22.77	25.56	26.59	28.65	28.67	30.7
	Cb	128.9	128.9	128.9	128.9	128.9	128.9	128.9	128.9	128.9	128.9	128.8	128.9	128.9	128.9	128.9	128.9
	Cr	129.2	129.2	129.2	129.2	129.2	129.2	129.2	129.2	129.2	129.2	129.7	129.2	129.2	129.2	130.6	129.2

Setelah matriks YCbCr didapatkan, langkah selanjutnya adalah melakukan proses *thresholding* untuk matriks YCbCr. *Thresholding* dilakukan dengan menggunakan operator AND dengan operasi ambang batas sebagai berikut.

$$\begin{cases} Y > 80 \\ 85 < Cb < 135 \\ 135 < Cr < 180 \end{cases}$$

Dimana $T(x, y)$ = [0,255], nilai 255 didapatkan jika semua kondisi terpenuhi [9]. Sebagai contoh nilai matriks YCbCr pada piksel (6,9), yaitu nilai Y=36.07, Cb=125.4, dan Cr=135.9, maka:

$$\begin{cases} 36.07 > 80\,, false \\ 85 < 125.4 < 135, true \\ 135 < 135.9 < 180, true \end{cases}$$

Berdasarkan kondisi tersebut maka nilai $T(6,9) = 0$, karena kondisi pada luminan Y tidak terpenuhi yaitu harus lebih dari 80. Contoh lain pada piksel (6,9), yaitu nilai Y=113.6, Cb=114.7, dan Cr=147.6, maka:

$$\begin{cases} 113.6 > 80\,, true \\ 85 < 114.7 < 135, true \\ 135 < 147.6 < 180, true \end{cases}$$

Berdasarkan kondisi tersebut maka nilai $T(9,6) = 255$, karena semua kondisi terpenuhi. Proses yang sama dilakukan terhadap setiap koordinat pada matriks YCbCr. Sehingga hasil dari *thresholding* dapat dilihat pada Tabel 3.4 berikut.

Tabel 3.4 Matriks *Thresholding*

Y \ X	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	255	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	255	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	255	255	255	255	0	0	0	0	0
7	0	0	0	0	0	0	0	0	255	255	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

3.4.2 Hand Tracking

Hand tracking ini bertujuan untuk melakukan pencarian area tangan dengan luas terbesar dan area tersebut dijadikan sebagai *Region of Interest* (ROI). ROI adalah bagian yang dipilih dari sampel dalam dataset untuk tujuan tertentu (dalam penelitian ini adalah bagian area tangan). Kemudian lebar atau panjang ROI akan membentuk persegi (*bounding area)* mengikuti kontur yang lebih besar. Kontur adalah kumpulan pixel yang merepresentasikan bentuk kurva dalam citra secara teratur. *Bounding area* yang terbentuk akan mengikuti posisi gerakan tangan dan mendapatkan *centroid* posisi tangan atau titik pusat dari tangan. Pada Gambar 3.5 dapat dilihat alur proses *hand tracking*.



Gambar 3.5 Alur Proses *Hand Tracking*

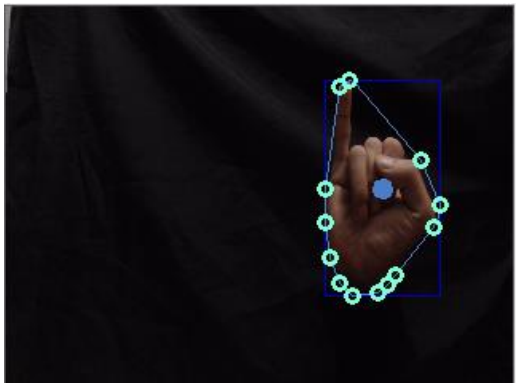
Pada penelitian ini proses *hand tracking* menggunakan fungsi-fungsi yang terdapat pada *library* EmguCV. Tahap pertama yang dilakukan adalah melakukan pencarian kontur terbesar pada frame hasil *threshold* dari tahap pendeteksian warna kulit yang memiliki nilai piksel 0 sebagai latar belakang (*background*) dan 255 sebagai objeknya dalam hal ini adalah tangan. Kontur terbesar yang dicari adalah pixel-pixel yang mempunyai nilai 255. Pencarian dilakukan dengan membandingkan setiap piksel pada *frame* yang sedang terdeteksi dengan piksel sebelumnya untuk mencari luas area kontur terbesar atau area yang memiliki banyak nilai piksel 255 paling banyak. Sebagai contoh matriks pada matriks thresholding pada Tabel 3.5 berikut.

Tabel 3.5 Matriks Hasil *Thresholding*

X / Y	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	255	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	255	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	255	255	255	255	0	0	0	0	0
7	0	0	0	0	0	0	0	0	255	255	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Dalam penerapannya setiap piksel akan ditelusuri untuk mencari piksel dengan nilai 255. Pada koordinat (4,7) akan dibandingkan dengan piksel sebelumnya yaitu koordinat (4,6), jika nilai koordinat yang sedang terdeteksi lebih besar dari piksel sebelumnya, maka nilai piksel akan disimpan ke dalam data kontur terbesar dan koordinatnya. Tahap ini akan terus dilakukan sampai piksel terakhir. Setelah didapat data kontur terbesar, tahap selanjutnya adalah melakukan pembentukan *bounding area* berdasarkan kontur terbesar yang sudah didapatkan. Langkah pertama yang dialakukan adalah mencari minimal area kontur terbesar yang berfungsi untuk mendapatkan kontur yang sesuai atau menghilangkan kontur yang tidak digunakan. Area minimal tersebut disimpan dalam variabel `box`. Kemudian dilakukan pembentukan *bounding area* dengan mengunakan fungsi `Draw` yang terdapat pada *library* EmguCV. Setelah didapatkan *bounding area*, langkah selanjutya adalah mencari *centroid* atau titik tengah berdasarkan *bounding area*

yang sudah didapatkan. Pencarian centroid menggunakan fungsi `center` pada *library* pada Emgu CV. Setelah didapatkan *centroid* (koordinat pusat) dari posisi tangan, langkah selanjutnya adalah menggambar *centroid* pada setiap frame yang terdeteksi dengan menggunakan fungsi `Draw` pada *library* EmguCV. Hasil dari pembentukan bounding area dan pencarian centroid dapat dilihat pada Gambar 3.6 berikut.



Gambar 3.6 Ilustrasi hasil *Hand Tracking*

Jika dilihat dari matriks *thresholding* nilai piksel yang terdapat dalam *bounding area* dapat dilihat pada Tabel 3.6 berikut.

Tabel 3.6 Nilai Piksel *Bounding Area* pada Matriks *Thresholding*

X / Y	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	255	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	255	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	255	255	255	255	0	0	0	0	0
7	0	0	0	0	0	0	0	0	255	255	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Selanjutnya adalah melakukan proses *cropping*, *cropping* merupakan proses penghapusan bagian sudut dari suatu gambar untuk memotong sebagian isi dari gambar guna memperoleh hasil yang diinginkan. *Cropping* dilakukan berdasarkan *bounding area* yang didapatkan menggunakan fungsi ROI pada *library* Emgu CV. Maka didapatkan matriks copping berukuran 4x5. Dalam penerapannya ukuran matriks dapat berubah-ubah sesuai dengan data masukkan bahasa isyarat. Matriks *cropping* dapat dilihat pada Tabel 3.7 berikut.

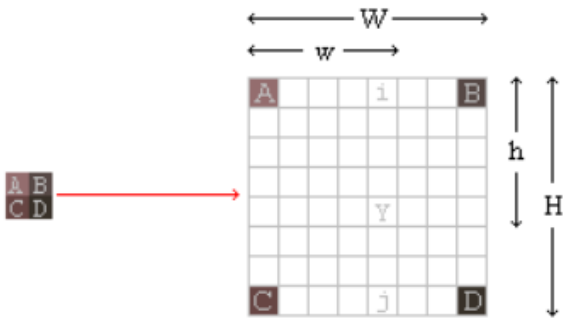
Tabel 3.7 Matriks *Cropping*

X / Y	1	2	3	4	5
1	255	0	0	0	0
2	0	255	0	0	0
3	0	255	255	255	255
4	0	0	255	255	0

Matriks *cropping* pada Tabel 3.7 akan dilakukan *resizing* yang bertujuan untuk menyamakan ukuran citra.

3.4.3 Resize

Resize merupakan proses mengubah resolusi atau ukuran horizontal dan vertikal suatu citra, hal ini bertujuan untuk menyamakan ukuran citra bahasa isyarat yang menjadi masukan dengan ukuran yang sama pada saat pelatihan maupun pengujian. Pada penelitian yang dilakukan oleh Revi [15], disebutkan bahwa pada saat pengujian pengaruh dimensi berpengaruh pada tingkat akurasi. Pada hasil dari pengujian didapatkan akurasi tertinggi pada dimensi 80x80 sebesar 75%. Berdasarkan hal tersebut, pada penelitian ini matriks bahasa isyarat akan diresize menjadi dimensi 80x80 pixel. Metode yang digunakan untuk proses risize adalah metode interpolasi liear. Interpolasi linear merupakan metode yang digunakan untuk mengetahui nilai dari sesuatu yang berada didalam sebuah interval atau didalam dua buah titik yang segaris. Cara kerja dari metode interpolasi linear adalah dengan mengambil empat sisi nilai-nilai piksel dari citra yang akan dilakukan *resize*. Kemudian dicari nilai piksel tengah diantara dua sisi dari citra. Ilustrasi dari metode interpolasi linear dapat dilihat pada Gambar 3.7 berikut.



Gambar 3.7 Ilustrasi Metode Interpolasi Linear

Berdasarkan Gambar 3.7, keempat sisi dari citra adalah nilai piksel A, B, C dan D. Kemudian dicari nilai piksel i, j, dan Y menggunakan persamaan (2.2) berikut.

$$i = A + \frac{w}{W}(B - A)$$

$$j = C + \frac{w}{W}(D - C)$$

$$y = i + \frac{h}{H}(j - i)$$

Dimana:

i, j, y= nilai warna yang akan dicari

A, B, C, D= nilai intensity yang didapatkan dari citra sebelumnya

w= lebar dari ukuran citra yang diinginkan

h= tinggi dari ukuran citra yang diinginkan

W= lebar jarak antara nilai warna yang akan dicari dengan warna sebelumnya (misalkan jarak A dengan i)

H= tinggi jarak antara nilai warna yang akan dicari dengan warna sebelumnya (misalkan jarak y dengan i)

Sebagai contoh matriks citra hasil *cropping* pada proses *hand tracking*, dapat dilihat pada Tabel 3.8 berikut.

Tabel 3.8 Matriks *Cropping*

t \ *l*	**1**	**2**	**3**	**4**	**5**
1	255	0	0	0	0
2	0	255	0	0	0
3	0	255	255	255	255
4	0	0	255	255	0

Pada Tabel 3.8 matriks berukuran 5x4 akan dilakukan *resize* menjadi ukuran 3x3. Dalam penerapannya matriks akan dilakukan *resize* menjadi ukuran 80x80. Langkah pertama yang dilakukan adalah mengambil nilai A, B, C, dan D pada matriks *cropping*. Dimana A adalah nilai piksel (1,1), B adalah nilai piksel (1, *l*), C adalah nilai piksel (*t*,1), dan D adalah nilai piksel (*l*, *t*). dimana nilai *l* adalah lebar matriks sebelum dilakukan resize dan *t* adalah tinggi matriks sebelum dilkukan resize. Pada tabel Tabel 3.9 dapat dilihat matiks *resize* yang nilainya belum terisi.

Tabel 3.9 Nilai Piksel Matriks *Resize* yang belum terisi

Y \ X	1	2	3
1	255	i	0
2		y	
3	0	j	0

Dalam penerapannya nilai yang pertama dicari adalah nilai i dan j, kemudian nilai i dan j disubstitusikan kedalam perhitungan Y untuk mencari nilai diantara dua buah piksel. Setelah itu dilakukan pencarian nilai piksel pada matriks *resize* (rz) yang belum terisi. Untuk pencarian piksel berdasarkan baris matriks seperti antara piksel(1,1) dengan (1,3) menggunakan rumus i atau j. Untuk pencarian piksel berdasarkan kolom matriks seperti nilai antara piksel(1,1) dengan piksel(1,3) dihitung menggunakan rumus Y. Lakukan tahap ini sampai semua piksel terisi. Kemudian hitung nilai i, j, Y, piksel(2,1), dan piksel(2,3) mengunakan persamaan (2.2) seperti berikut.

$$i = A + \frac{w}{W}(B - A)$$

$$= 255 + \frac{2}{3}(0 - 255) \quad = 255 - 170 \quad = 85$$

$$j = C + \frac{w}{W}(D - C)$$

$$= 0 + \frac{2}{3}(0 - 0) \quad = 0$$

$$y = i + \frac{h}{H}(j - i)$$

$$= 85 + \frac{2}{3}(0 - 85) = 85 - 56{,}6 = 28{,}3$$

$$rz(2,1) = A + \frac{h}{H}(C - A)$$

$$= 85 + \frac{2}{3}(0 - 255) = 85 - 170 = 85$$

$$rz(2,3) = B + \frac{h}{H}(D - B)$$

$$= 0 + \frac{2}{3}(0 - 0) = 0$$

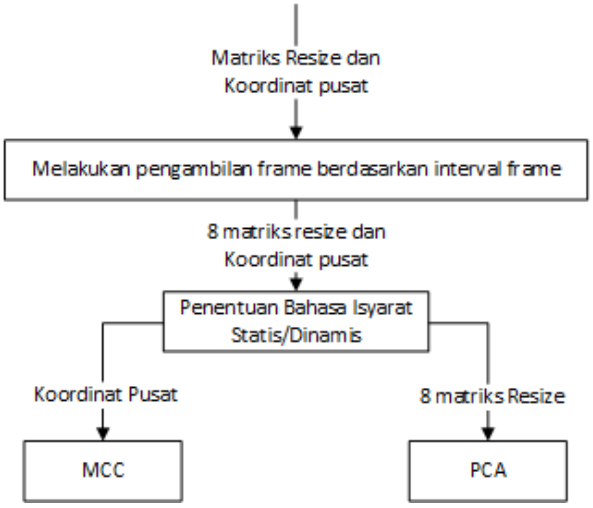
Setelah semua nilai piksel terisi, kemudian setiap nilai piksel dilakukan thresholding kembali. Jika nilai piksel lebih dari sama dengan 80 maka nilai piksel menjadi 255, dan jika nilai piksel kurang dari 80 maka nilai piksel menjadi 0. Maka didapatkan matriks seperti pada tabel Tabel 3.10 berikut.

Tabel 3.10 Matriks *Resize*

X / Y	1	2	3
1	255	255	0
2	255	0	0
3	0	0	0

3.4.4 Pengambilan *Frame*

Pada tahapan ini akan dilakukan proses pengambilan *frame* sehingga hanya *frame* yang mengandung gerakan bahasa isyarat saja yang akan digunakan untuk proses selanjutnya. Pada penelitian sebelumnya telah dilakukan perbandingan jumlah pengambilan *frame*, yaitu pengambilan 10 *frame*, 20 *frame*, dan 30 *frame*. Dimana disbutkan bahwa pengambilan *frame* dengan jumlah 20 dan 30 *frame* akurasi akan meningkat tetapi waktu komputasinya lebih lama, karena jumlah *frame* yang diambil memberikan ciri lebih detail [15]. Berdasarkan hal tersebut jumlah *frame* yang akan diambil dari data masukan video adalah 20 *frame*. Untuk lebih jelasnya dapat dilihat alur pengambilan *frame* pada Gambar 3.8 berikut.



Gambar 3.8 Alur Proses Pengambilan *Frame*

Berdasarkan Gambar 3.8 dapat dijelaskan data masukan untuk proses pengambilan *frame* adalah hasil *resize*. Pengambilan frame akan dilakukan berdasarkan interval *frame*. Tujuan dari pengambilan *frame* adalah untuk mengambil *frame* yang hanya mengandung gerakan bahasa isyarat saja. Penentuan interval *frame* dilakukan menggunakan perhitungan berikut.

$$Iv = \frac{Jum_frame\ -5}{n}$$

Dimana:

Iv= interval *frame*

Jum_frame = jumlah *frame* keseluruhan

n = jumlah *frame* yang akan diambil

Contoh pada data masukan video memiliki jumlah *frame*= 65 *frame* dan jumlah frame yang akan diambil sebanyak 8 *frame* untuk kebutuhan analisis. Interval *frame* dapat dihitung:

$$Iv = \frac{Jum_frame\ -5}{n}$$

$$= \frac{65\ -5}{8}$$

$= 7{,}5$ (dibulatkan kebawah = 7)

Berdasarkan hasil perhitungan *interval*, maka setiap interval 7 *frame* akan diambil *frame* beserta nilai koordinat pusatnya. Berdasarkan interval *frame* yang akan diambil adalah *frame* ke 7, 14, 21, 28, 35, 42, 49, dan 56 akan diambil dan diambil koordinat pusatnya. Setelah didapatkan *frame* dan koordinat pusat yang mengandung gerakan bahasa isyat, proses selanjutnya adalah menentukan jenis gerakan bahasa isyarat termasuk kedalam bahasa isyarat statis atau bahasa isyrat dinamis. Penentuan jenis gerakan bahasa isyarat dilakukan menggunakan persamaan berikut.

$$s = x_{max} - x_{min}$$

Dimana:

s= Jarak

x_{max}= nilai maksimum koordinat x

x_{min}= nilai minimum koordinat x

Kemudian dari jarak tersebut dilakukan pengecekan kondisi menggunakan ambang batas pada persamaan berikut.

$$jenisisyarat = \begin{cases} Dinamis & s > 15 \\ Statis & s \leq 15 \end{cases}$$

Sebagai contoh pertama koordinat pusat pada huruf A dapat dilihat pada Tabel 3.11 berikut.

Tabel 3.11 Koordinat Huruf A

Frame Ke-	Koordinat Pusat	
	X	Y
1	181	133
2	183	119
3	183	119
4	184	119
5	184	119
6	185	119
7	185	117
8	184	116

Berdasarkan Tabel 3.11 didapatkan nilai x_{max}= 185 dan nilai x_{min}= 181, maka dapat dihitung jaraknya seperti berikut.

$s = x_{max} - x_{min}$

$s = 185 - 181 \quad = 4$

Hasil perhitungan jarak koordinat pusat untuk huruf A adalah 4. Kemudian lakukan pengecekan kondisi dan didapatkan hasil bahwa 4 < 15, maka huruf A termasuk bahasa isyarat statis. Maka matriks *frame* akan digunakan untuk proses ekstraksi ciri PCA, dalam penerapannya matriks yang digunakan berjumlah 20 *frame*. Pada Tabel 3.12 dan Tabel 3.13 dapat dilihat hasil dari pengambilan *frame*.

Tabel 3.12 Matriks Frame ke-1 huruf I

	1	2	3	4	5	6	7	8	9	10	…	80
1	0	0	0	0	0	0	0	0	0	0	…	0
2	0	0	0	0	0	0	0	0	0	0	…	0
3	0	0	255	0	0	0	0	0	0	0	…	0
4	0	0	0	0	0	0	0	0	0	0	…	0
5	0	0	0	0	0	0	0	0	0	0	…	0
6	0	0	0	0	0	0	0	0	0	0	…	0
7	0	0	0	0	0	0	0	0	0	0	…	0
8	0	0	0	0	0	0	0	0	0	0	…	0
9	0	0	0	0	0	0	0	0	0	0	…	0
10	0	0	0	0	0	0	0	0	0	0	…	0
…	..	…	…	…	…	…	…	…	…	…	…	…
80	0	0	0	0	0	0	0	0	0	0	…	0

.

.

.

Tabel 3.13 Matriks Frame ke-8 huruf I

	1	2	3	4	5	6	7	8	9	10	…	80
1	0	0	0	0	0	0	0	0	0	0	…	0
2	0	0	0	0	0	0	0	0	0	0	…	0
3	0	0	0	0	255	0	0	0	0	0	…	0
4	0	0	0	0	0	0	0	0	0	0	…	0
5	0	0	0	0	0	0	0	0	0	0	…	0
6	0	0	0	0	0	0	0	0	0	0	…	0
7	0	0	0	0	0	0	0	0	0	0	…	0
8	0	0	0	0	0	0	0	0	0	0	…	255
9	0	0	0	0	0	0	0	0	255	0	…	255
10	0	0	0	0	0	0	0	0	0	0	…	0
…	..	…	…	…	…	…	…	…	…	…	…	…
80	0	0	0	0	0	0	0	0	0	0	…	0

Contoh lain untuk penentuan gerakan bahasa isyarat adalah koordinat pusat huruf pada Tabel 3.14 berikut.

Tabel 3.14 Koordinat Pusat huruf J

Frame Ke-	Koordinat Pusat	
	X	Y
1	103	126
2	96	112
3	113	113
4	111	111
5	161	161
6	157	157
7	149	149
8	152	152

Berdasarkan Tabel 3.14 didapatkan nilai x_{max}= 161 dan nilai x_{min}= 96, maka dapat dihitung jaraknya seperti berikut.

$$s = x_{max} - x_{min}$$

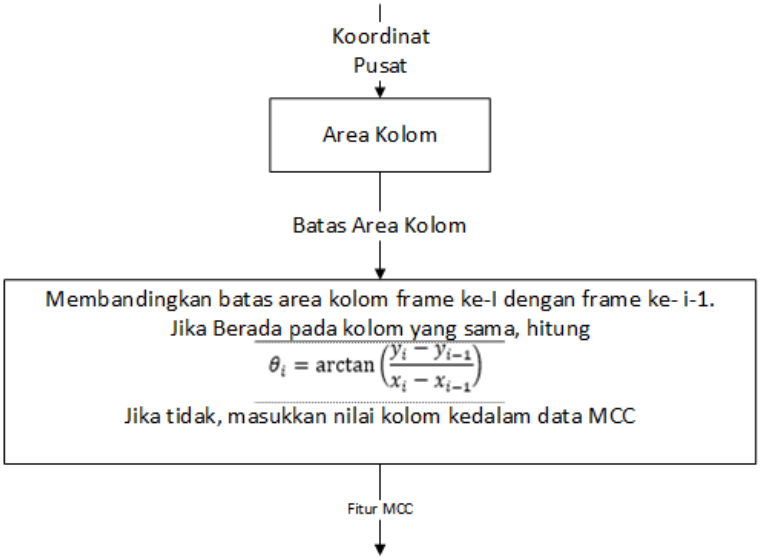
$$s = 161 - 96$$

$$= 65$$

Hasil perhitungan jarak koordinat pusat untuk huruf A adalah 4. Kemudian lakukan pengecekan kondisi dan didapatkan hasil bahwa 65 > 15, maka huruf J termasuk bahasa isyarat dinamis. Koordinat pusat pada Tabel 3.14 akan digunakan untuk proses selanjutnya yaitu proses Algoritma *Motion Chain Code* (MCC).

3.5 Analisis Proses Algoritma *Motion Chain Code* (MCC)

Pada penelitian ini *Motion Chain Code* (MCC) digunakan untuk mengekstraksi gerakan relatif tangan dengan mengurangkan sebuah centroid tangan di dalam dua buah *frame* [5]. Nilai keluaran dari MCC akan digunakan sebagai fitur bahasa isyarat dinamis pada saat pengujian. Pada Gambar 3.9 dapat dilihat alur proses dari algoritma *motion chain code*.



Gambar 3.9 Alur Proses Algoritma Motion Chain Code

Berdasarkan Gambar 3.9 dapat dijelaskan, data koordinat pusat yang digunakan adalah koordinat pusat dari pengambilan *frame*. Langkah pertama adalah menentukan area kolom yang dibagi menjadi empat kolom. Setiap area kolom memiliki batas area masing-masing. Untuk menghitung batas area setiap kolom dibutuhkan nilai minimum dan nilai maksimum dari nilai koordinat X. Contoh koordinat pusat pada gerakan bahasa isyarat huruf J pada Tabel 3.15 berikut.

Tabel 3.15 Koordinat Pusat Huruf J

Frame Ke-	Koordinat Pusat	
	X	Y
1	103	126
2	96	112
3	113	113
4	111	111
5	161	161
6	157	157
7	149	149
8	152	152

Berdasarkan Tabel 3.15 nilai minimum pada koordinat X (x_{min}) berada pada koordinat 79 dan nilai maksimum koordinat x (x_{max}) berada pada koordinat 103. Kemudian hitung batas area setiap kolom menggunakan persamaan (2.8). Berikut perhitungan batas area setiap kolom:

$$batas_1 = \frac{x_{min} + x_{max}}{2}$$

$$= \frac{79 + 103}{2} = \frac{182}{2} = 91$$

$$batas_2 = \frac{batas_1 + x_{max}}{2}$$

$$= \frac{91 + 103}{2} = \frac{194}{2} = 97$$

$$batas_3 = \frac{batas_2 + x_{max}}{2}$$

$$= \frac{97 + 103}{2} = \frac{200}{2} = 100$$

Berdasarkan hasil perhitungan, didapatkan persamaan area setiap kolom menggunakan batas-batas area kolom dari hasil perhitungan adalah sebagai berikut.

$$Kolom = \begin{Bmatrix} 9, & 0 < x < 91 \\ 10, & 91 < x < 97 \\ 11, & 97 < x < 100 \\ 12, & 100 < \mathrm{x} < x_n \end{Bmatrix} \quad (3.1)$$

Area kolom memiliki nilai 9, 10, 11 dan 12 dikarenakan untuk membedakan nilainya dengan nilai arah sudut gerakan yang memiliki nilai dari 1 sampai 8 yang dapat dilihat pada persamaan (2.10) berikut [5].

$$Nilai\ Sudut\ Gerakan = \begin{Bmatrix} 1, & \theta < 22{,}5 \text{ dan } \theta > 337{,}5^o \\ 2, & 22{,}5^O < \theta < 67{,}5^o \\ 3, & 67{,}5^O < \theta < 110{,}5^o \\ 4, & 110{,}5^O < \theta < 157{,}5^o \\ 5, & 157{,}5^O < \theta < 202{,}5^o \\ 6, & 202{,}5^O < \theta < 242{,}5^o \\ 7, & 242{,}5^O < \theta < 292{,}5^o \\ 8, & 292{,}5^O < \theta < 337{,}5^o \end{Bmatrix}$$

Setelah didapatkan area kolom maka langkah selanjutnya adalah melakukan pengecekan kondisi setiap koordinat. Jika koordinat (x_i) dan koordinat sebelumnya (x_{i-1}) tidak berada pada kolom yang sama, maka tentukan nomor kolom koordinat *frame* yang sedang terdeteksi berada. Jika koordinat (x_i) dan koordinat sebelumnya

(x_{i-1}) berada pada kolom yang sama, maka lakukan perhitungan nilai sudut menggunakan persamaan (2.10) berikut.

$$\theta_i = \arctan\left(\frac{y_i - y_{i-1}}{x_i - x_{i-1}}\right)$$

Dimana:

θ_i = Nilai sudut gerakan pada frame-i

x_i= Nilai koordinat x

y_i= nilai koordinat y

x_{i-1}= nilai koordinat x sebelumnya

y_{i-1}= nilai koordinat y sebelumnya

Kemudian dari hasil perhitungan nilai sudut, tentukan nilai arah gerakan berdasarkan θ atau nilai sudut yang didapatkan sebelumnya menggunakan kondisi pada persamaan (2.10). Proses pengecekan kondisi menggunakan data koordinat pada Tabel 3.15 hasil dari pengambilan *frame*. Berikut pengecekan kondisi setiap koordinat sampai *frame* terakhir.

1. Pengecekan *Frame*-2 dengan *Frame*-1

Pengecekan kondisi untuk *frame*-2 dengan *frame*-1 menggunakan data masukan pada Tabel 3.16 berikut.

Tabel 3.16 Perbandingan *frame* ke-1 dan *frame* ke-2

No	*Frame*	Koordinat Pusat
1		X = 103 Y = 126
2		X = 96 Y = 112

Berdasarkan persamaan (3.1), nilai koordinat x pada *frame* ke-2 dan *frame* ke-1 pada Tabel 3.16 tidak berada pada area kolom yang sama, maka masukkan nilai area kolom untuk *frame* ke-2 yaitu 12 kedalam data MCC: (12).

2. Pengecekan *Frame*-3 dengan *Frame*-2

Pengecekan kondisi untuk *frame*-3 dengan *frame*-2 menggunakan data masukan pada Tabel 3.17 berikut.

Tabel 3.17 Perbandingan *frame* ke-2 dan *frame* ke-3

No	*Frame*	Koordinat Pusat
2		X = 96 Y = 112
3		X = 95 Y = 113

Berdasarkan persamaan (3.1), nilai koordinat x pada *frame* ke-2 dan *frame* ke-3 pada Tabel 3.17 berada pada area kolom yang sama, maka lakukan perhitungan perhitungan nilai sudut menggunakan persamaan (2.10) seperti berikut.

$$\theta_i = \arctan\left(\frac{y_i - y_{i-1}}{x_i - x_{i-1}}\right)$$

$$\theta_1 = \arctan\left(\frac{y_3 - y_2}{x_3 - x_2}\right)$$

$$\theta_1 = \arctan\left(\frac{113 - 112}{95 - 96}\right)$$

$$= \arctan\left(\frac{1}{-1}\right)$$

$$= \arctan(-1)$$

$$= -45^o$$

Jika hasil dari perhitungan < 0 , maka hasil dari θ ditambah 360. Maka 360 +(-45) = 315^o. Berdasarkan persamaan (2.9) nilai sudut 315^o memiliki nilai arah sudut gerakan = 8. Masukkan nilai 8 kedalam data MCC: (12, 8).

3. Pengecekan *Frame*-4 dengan *Frame*-3

Pengecekan kondisi untuk *frame*-4 dengan *frame*-3 menggunakan data masukan pada Tabel 3.18 berikut.

Tabel 3.18 Perbandingan *frame* ke-3 dan *frame* ke-4

No	*Frame*	Koordinat Pusat
3		X = 95 Y = 113
4		X = 94 Y = 111

Berdasarkan persamaan (3.1), nilai koordinat x pada *frame* ke-3 dan *frame* ke-4 pada Tabel 3.17 berada pada area kolom yang sama, maka lakukan perhitungan perhitungan nilai sudut menggunakan persamaan (2.10) seperti berikut.

$$\theta_i = \arctan\left(\frac{y_i - y_{i-1}}{x_i - x_{i-1}}\right)$$

$$\theta_2 = \arctan\left(\frac{y_4 - y_3}{x_4 - x_3}\right)$$

$$\theta_2 = \arctan\left(\frac{111 - 113}{94 - 95}\right)$$

$$= \arctan\left(\frac{-2}{-1}\right)$$

$$= \arctan(2)$$

$$= 63{,}43^o$$

Berdasarkan persamaan (2.9), nilai sudut $63{,}43^o$ memiliki nilai arah sudut gerakan = 2. Masukkan nilai 2 kedalam data MCC: (12, 8, 2).

4. *Frame*-4 dengan *Frame*-5

Pengecekan kondisi untuk *frame*-5 dengan *frame*-4 menggunakan data masukan pada Tabel 3.19 berikut.

Tabel 3.19 Perbandingan *frame* ke-4 dan *frame* ke-5

No	*Frame*	Koordinat Pusat
4		X = 95 Y = 113
5		X = 94 Y = 111

Berdasarkan persamaan (3.1), nilai koordinat x pada *frame* ke-4 dan *frame* ke-5 pada Tabel 3.19 tidak berada pada area kolom yang sama, maka masukkan nilai area kolom untuk *frame* ke-2 yaitu A kedalam data MCC: (12, 8, 2, 9).

5. Pengecekan *Frame*-6 dengan *Frame*-5

Pengecekan kondisi untuk *frame*-6 dengan *frame*-5 menggunakan data masukan pada Tabel 3.20 berikut.

Tabel 3.20 Perbandingan *frame* ke-5 dan *frame* ke-6

No	*Frame*	Koordinat Pusat
5		X = 76 Y = 161
6		X = 85 Y = 157

Berdasarkan persamaan (3.1), nilai koordinat x pada *frame* ke-5 dan *frame* ke-6 pada Tabel 3.20 berada pada area kolom yang sama, maka lakukan perhitungan perhitungan nilai sudut menggunakan persamaan (2.10) seperti berikut.

$$\theta_i = \arctan\left(\frac{y_i - y_{i-1}}{x_i - x_{i-1}}\right)$$

$$\theta_3 = \arctan\left(\frac{y_6 - y_5}{x_6 - x_5}\right)$$

$$\theta_3 = \arctan\left(\frac{157 - 161}{85 - 76}\right)$$

$= \arctan\left(\frac{-4}{9}\right)$

$= \arctan(-0{,}444)$

=-23,96^o

Jika hasil dari perhitungan < 0 , maka hasil dari θ ditambah 360. Maka 360 +(-23,96) = 336^o. Berdasarkan persamaan (2.9) nilai sudut 336^o memiliki nilai arah sudut gerakan = 8. Masukkan nilai 8 kedalam data MCC: (12, 8, 2, 9, 8).

6. *Frame*-6 dengan *Frame*-7

Pengecekan kondisi untuk *frame*-7 dengan *frame*-6 menggunakan data masukan pada Tabel 3.21 berikut.

Tabel 3.21 Perbandingan *frame* ke-6 dan *frame* ke-7

No	*Frame*	Koordinat Pusat
6		X = 85 Y = 157
7		X = 86 Y = 149

Berdasarkan persamaan (3.1), nilai koordinat x pada *frame* ke-6 dan *frame* ke-7 pada Tabel 3.21 berada pada area kolom yang sama, maka lakukan perhitungan perhitungan nilai sudut menggunakan persamaan (2.10) seperti berikut.

$\theta_i = \arctan\left(\frac{y_i - y_{i-1}}{x_i - x_{i-1}}\right)$

$\theta_4 = \arctan\left(\frac{y_7 - y_6}{x_7 - x_6}\right)$

$\theta_4 = \arctan\left(\frac{149 - 157}{86 - 85}\right)$

$= \arctan\left(\frac{-8}{1}\right)$

$= \arctan(-8)$

=-82,87^o

Jika hasil dari perhitungan < 0 , maka hasil dari θ ditambah 360. Maka 360 +(-82,87) = 277,13^o. Berdasarkan persamaan (2.9) sudut 277^o memiliki nilai arah sudut gerakan = 7. Masukkan nilai 7 kedalam data MCC: (12, 8, 2, 9, 8, 7).

7. *Frame*-7 dengan *Frame*-8

Pengecekan kondisi untuk *frame*-8 dengan *frame*-7 menggunakan data masukan pada Tabel 3.22 berikut.

Tabel 3.22 Perbandingan *frame* ke-7 dan *frame* ke-8

No	*Frame*	Koordinat Pusat
7		X = 86 Y = 149
8		X = 87 Y = 152

Berdasarkan persamaan (3.1), nilai koordinat x pada *frame* ke-7 dan *frame* ke-8 pada Tabel 3.22 berada pada area kolom yang sama, maka lakukan perhitungan perhitungan nilai sudut menggunakan persamaan (2.10) seperti berikut.

$$\theta_i = \arctan\left(\frac{y_i - y_{i-1}}{x_i - x_{i-1}}\right)$$

$$\theta_5 = \arctan\left(\frac{y_8 - y_7}{x_8 - x_7}\right)$$

$$\theta_5 = \arctan\left(\frac{152 - 149}{87 - 86}\right)$$

$$= \arctan\left(\frac{3}{1}\right) = \arctan(3)$$

$$=71^o$$

Berdasarkan persamaan (2.9), maka sudut 71^o memiliki nilai arah sudut gerakan = 3. Masukkan nilai 3 kedalam data MCC: (12, 8, 2, 9, 8, 7, 3). Perulangan sudah selesai dilakukan, maka data fitur dari proses MCC untuk bahasa isyarat huruf J yang didapatkan adalah (12, 8, 2, 9, 8, 7, 3). Data inilah yang akan digunakan sebagai fitur untuk gerakan bahasa isyarat dinamis pada saat proses pelatihan dan pengujian.

3.6 Binerisasi

Pada proses binerisasi merupakan proses merubah nilai piksel citra menjadi nilai biner yaitu 0 dan 1. Proses binerisasi dilakukan dengan cara merubah setiap piksel yang memiliki nilai 255 diubah menjadi nilai 1. Pada Tabel 3.23 dapat dilihat hasil dari proses binerisasi yang didapatkan dari matriks *frame* ke-1 pada huruf I pengambilan *frame*.

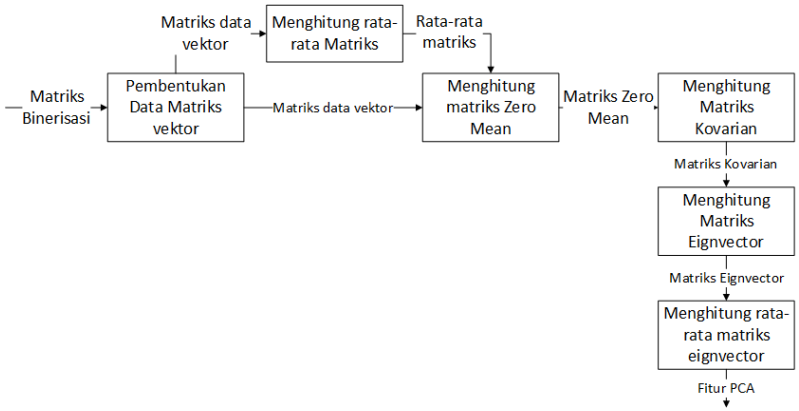
Tabel 3.23 Matriks Binerisasi

	1	2	3	4	5	6	7	8	9	10	…	80
1	0	0	0	0	0	0	0	0	0	0	…	0
2	0	0	0	0	0	0	0	0	0	0	…	0
3	0	0	1	0	0	0	0	0	0	0	…	0
4	0	0	0	0	0	0	0	0	0	0	…	0
5	0	0	0	0	0	0	0	0	0	0	…	0
6	0	0	0	0	0	0	0	0	0	0	…	0
7	0	0	0	0	0	0	0	0	0	0	…	0
8	0	0	0	0	0	0	0	0	0	0	…	0
9	0	0	0	0	0	0	0	0	0	0	…	0
10	0	0	0	0	0	0	0	0	0	0	…	0
…	..	…	…	…	…	…	…	…	…	…	…	…
80	0	0	0	0	0	0	0	0	0	0	…	0

Matriks pada Tabel 3.23 akan digunakan pada proses ekstraksi ciri PCA untuk fitur bahasa isyarat statis.

3.7 Analisis Proses Ekstraksi Ciri *Principal Component Analysis* (PCA)

Pada proses ekstraksi ciri PCA merupakan proses teknik-teknik sederhana dalam statistika. Proses ekstraksi ciri PCA menggunakan data masukan berupa matriks hasil dari proses binerisasi. Alur ekstraksi ciri PCA dapat dilihat pada Gambar 3.10 berikut.



Gambar 3.10 Alur Proses Ekstraksi Ciri PCA

Setelah dilakukan proses binerisasi yaitu mengubah setiap nilai matriks yang memiliki nilai 255 di setiap matriks *frame* menjadi nilai 1. Sehingga didapatkan matriks biner yang hanya mempunyai dua nilai yaitu 0 dan 1. Selanjutnya adalah pembentukan matriks data citra berukuran M x N. Dimana N adalah jumlah citra yang dilatih dan M merupakan dimensi dari citra. Kemudian matriks setiap *frame* yang memiliki ukuran 80x80 akan diubah menjadi matriks satu dimensi berukuran 6400x1. Kemudian seluruh matriks *frame* berukuean 6400x1 akan digabungkan menjadi satu matriks berukuran 6400x8 yang dapat dilihat pada Tabel 3.24 berikut.

Tabel 3.24 Matriks Data Citra 6400x8

M \ N	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0
14	0	0	0	0	0	1	1	1
15	0	0	0	0	0	1	1	1
16	0	1	0	1	0	1	0	0
17	1	1	0	1	1	1	0	0
18	1	1	0	1	1	1	0	0
19	1	1	1	1	1	1	0	0
20	1	1	1	1	1	0	0	0
…	…	…	…	…	…	…	…	…
6400	0	0	0	0	0	0	0	0

Kemudian dari matriks data citra dihitung nilai rata-rata dengan mengunakan persamaan (2.3) berikut.

$$\bar{x}_N = \frac{\sum_{i=1}^{M} x_i}{M} = \frac{x_1 + x_2 + \cdots + x_M}{M}$$

Hitung rata-rata nilai untuk seriap N atau setiap frame seperti berikut.

$$\bar{x}_N = \frac{x_1 + x_2 + \cdots + x_M}{M}$$

$$\bar{x}_1 = \frac{0 + 0 + +0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + \cdots + 0}{6400}$$

$$\bar{x}_1 = \frac{3719}{6400} = 0{,}5810938$$

Lakukan perhitungan tersebut sampai $\bar{x}_8$, hasil dari perhitungan nilai rata-rata dapat dilihat pada Tabel 3.25 beikut.

Tabel 3.25 Nilai Rata-rata($\bar{x}$)

$\bar{x}_N$	$\bar{x}_1$	$\bar{x}_2$	$\bar{x}_3$	$\bar{x}_4$	$\bar{x}_5$	$\bar{x}_6$	$\bar{x}_7$	$\bar{x}_8$
$\bar{x}$	0.58109	0.58109	0.57031	0.57125	0.57015	0.57203	0.56656	0.56265

Setelah didapatkan nilai rata-rata ($\bar{x}$), langkah selanjutnya menghitung nilai *zero mean* yaitu setiap nilai pada data sampel dikurangi nilai rata-rata tiap fitur terkait. *Zero mean* berfungsi untuk menghilangkan *noise* yang dapat mengganggu keakuratan pada perhitungan PCA. Untuk menghitung matriks *zero mean* dapat dimodelkan menggunakan persamaan (2.4) berikut.

$$Y = D - \bar{x}$$

$$Y = \begin{bmatrix} D_{11} - \bar{x}_1 & D_{12} - \bar{x}_2 & \cdots & D_{1,n} - \bar{x}_n \\ D_{21} - \bar{x}_n & D_{22} - \bar{x}_2 & \cdots & D_{1,n} - \bar{x}_n \\ \cdots & \cdots & \cdots & \cdots \\ D_{m,1} - \bar{x}_1 & D_{m,1} - \bar{x}_2 & \cdots & D_{m,n} - \bar{x}_n \end{bmatrix}$$

$$= \begin{bmatrix} 0 - 0.58109 & 0 - 0.58109 & \cdots & 0 - 0.56265 \\ 0 - 0.58109 & 0 - 0.58109 & \cdots & 0 - 0.56265 \\ \cdots & \cdots & \cdots & \cdots \\ 0 - 0.58109 & 0 - 0.58109 & \cdots & 0 - 0.56265 \end{bmatrix}$$

$$Matriks\ Y = \begin{bmatrix} -0.58109 & -0.58109 & \cdots & -0.56265 \\ -0.58109 & -0.58109 & \cdots & -0.56265 \\ \cdots & \cdots & \cdots & \cdots \\ -0.58109 & -0.58109 & \cdots & -0.56265 \end{bmatrix}$$

Hasil yang didapatkan dari perhitungan matriks *zero mean* (*Y*) dapat dilihat pada Tabel 3.26 berikut.

Tabel 3.26 Matriks *Zero Mean*

i / j	1	2	3	4	5	6	7	8
1	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	-0.57203	-0.56656	-0.56266
2	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	-0.57203	-0.56656	-0.56266
3	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	-0.57203	-0.56656	-0.56266
4	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	-0.57203	-0.56656	-0.56266
5	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	-0.57203	-0.56656	-0.56266
6	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	-0.57203	-0.56656	-0.56266
7	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	-0.57203	-0.56656	-0.56266
8	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	-0.57203	-0.56656	-0.56266
9	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	-0.57203	-0.56656	-0.56266
10	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	-0.57203	-0.56656	-0.56266
11	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	-0.57203	-0.56656	-0.56266
12	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	-0.57203	-0.56656	-0.56266
13	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	-0.57203	-0.56656	-0.56266
14	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	0.42797	0.43344	0.43734
15	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	0.42797	0.43344	0.43734
16	-0.58109	0.41891	-0.57031	0.42875	-0.57016	0.42797	-0.56656	-0.56266
17	0.41891	0.41891	-0.57031	0.42875	0.42984	0.42797	-0.56656	-0.56266
19	0.41891	0.41891	-0.57031	0.42875	0.42984	0.42797	-0.56656	-0.56266
20	0.41891	0.41891	0.42969	0.42875	0.42984	0.42797	-0.56656	-0.56266
…	…	…	…	…	…	…	…	…
6400	-0.58109	-0.58109	-0.57031	-0.57125	-0.57016	-0.57203	-0.56656	-0.56266

Setelah mendapatkan matriks *zero mean*, langkah selanjutnya adalah menghitung matriks kovarian. Matriks kovarian digunakan untuk memperoleh fitur/ciri dari *sample* tangan. Matriks kovarian diperoleh dengan cara mengkalikan matriks *zero mean* dengan matriks trasposenya. Matriks kovarian dihitung menggunakan persamaan (2.5).

$$C = \frac{1}{M-1} Y^T Y$$

$$C = \frac{1}{6400-1} \begin{bmatrix} D_{1,1} & D_{2,1} & \cdots & D_{6400,1} \\ D_{1,2} & D_{2,2} & \cdots & D_{6400,2} \\ \cdots & \cdots & \cdots & \cdots \\ D_{1,8} & D_{2,8} & \cdots & D_{6400,8} \end{bmatrix} \text{x} \begin{bmatrix} D_{1,1} & D_{1,2} & \cdots & D_{1,8} \\ D_{2,1} & D_{2,2} & \cdots & D_{2,8} \\ \cdots & \cdots & \cdots & \cdots \\ D_{6400,1} & D_{6400,2} & \cdots & D_{6400,8} \end{bmatrix}$$

$$= \frac{1}{6400-1}\begin{bmatrix} -0.58109 & -0.58109 & \cdots & -0.58109 \\ -0.58109 & -0.58109 & \vdots & -0.58109 \\ \cdots & \cdots & \cdots & \cdots \\ -0.56265 & -0.56265 & \cdots & -0.56265 \end{bmatrix} * \begin{bmatrix} -0.58109 & -0.58109 & \cdots & -0.56265 \\ -0.58109 & -0.58109 & \vdots & -0.56265 \\ \cdots & \cdots & \cdots & \cdots \\ -0.58109 & -0.58109 & \cdots & -0.56265 \end{bmatrix}$$

$$= \frac{1}{6399}\begin{bmatrix} 1557.9 & 1450.9 & \cdots & 1384.48 \\ 1450.91 & 1557.9 & \vdots & 1391.48 \\ \cdots & \cdots & \cdots & \cdots \\ 1384.48 & 1391.4814 & \cdots & 1574.87 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1557.9}{6399} & \frac{1450.9}{6399} & \cdots & \frac{1384.48}{6399} \\ \frac{1450.91}{6399} & \frac{1557.9}{6399} & \vdots & \frac{1391.48}{6399} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{1384.48}{6399} & \frac{1391.48}{6399} & \cdots & \frac{1574.87}{6399} \end{bmatrix}$$

$$C = \begin{bmatrix} 0.24346 & 0.22674 & \cdots & 0.21636 \\ 0.22674 & 0.24346 & \vdots & 0.21745 \\ \cdots & \cdots & \cdots & \cdots \\ 0.21636 & 0.21745 & \cdots & 0.24611 \end{bmatrix}$$

Dari perhitungan matriks kovarian yang sudah didapatkan maka hasil yang didapatkan dari perhitungan matriks kovarian dapat dilihat pada Tabel 3.27 berikut.

Tabel 3.27 Matriks Kovarian

i / j	1	2	3	4	5	6	7	8
1	0.24346	0.22674	0.22988	0.22512	0.22560	0.22607	0.21815	0.21636
2	0.22674	0.24346	0.22769	0.22559	0.22356	0.22451	0.22003	0.21745
3	0.22988	0.22769	0.24509	0.22737	0.22627	0.22520	0.21801	0.21758
4	0.22512	0.22559	0.22737	0.24496	0.22824	0.22217	0.21545	0.21205
5	0.22560	0.22356	0.22627	0.22824	0.24512	0.22420	0.21935	0.21501
6	0.22607	0.22451	0.22520	0.22217	0.22420	0.24485	0.22391	0.22474
7	0.21815	0.22003	0.21801	0.21545	0.21935	0.22391	0.24561	0.22313
8	0.21636	0.21745	0.21758	0.21205	0.21501	0.22474	0.22313	0.24611

Setelah didapatkan matriks kovarian maka akan dilakukan perhitungan nilai *eigenvalue* dan *eigenvector*. Nilai *eigenvalue* dapat dihitung mengunakan persamaan (2.7) berikut.

$C - Z = |C - \lambda I|$

$|C - \lambda I| = 0$

$$\begin{bmatrix} C_{11} & C_{12} & \dots C_{1,n} \\ C_{21} & C_{22} & \dots C_{2,n} \\ \dots & \dots & \dots \dots \\ C_{m,1} & C_{m,2} \dots & C_{m,n} \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \dots & 0 \\ 0 & 1 \dots & 0 \\ \dots & \dots\dots & \dots \\ 0 & 0 \dots & 1 \end{bmatrix} = 0$$

$$\begin{bmatrix} 0.24346 & 0.22674 & \dots 0.21636 \\ 0.22674 & 0.24346 & \dots 0.21745 \\ \dots & \dots & \dots \dots \\ 0.21636 & 0.21745 & \dots 0.24611 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \dots & 0 \\ 0 & \lambda \dots & 0 \\ \dots & \dots\dots & \dots \\ 0 & 0 \dots & \lambda \end{bmatrix} = 0$$

$$\begin{bmatrix} 0.24346 - \lambda & 0.22674 & \cdots & 0.21636 \\ 0.22674 & 0.24346 - \lambda & \cdots & 0.21745 \\ \cdots & \cdots & \dots & \cdots \\ 0.21636 & 0.21745 & \cdots & 0.24611 - \lambda \end{bmatrix} = 0$$

Berdasarkan matriks yang didapatkan, kemudian didapatkan persamaan berikut menggunakan *characteristic polynomial* [10].

$$-\lambda^8 + 1.95866\lambda^7 - 0.29188\lambda^6 + 0.01857\lambda^5 - 0.0006435\lambda^4 + 0.00001\lambda^3 - 1.574568e-7\lambda^2 + 1.033097e-9\lambda - 2.861807e-12 = 0$$

Maka didapatkan nilai *eignvalue* sebagai berikut.

$\lambda_1 = 1.801801$ $\lambda_2 = 0.47871$ $\lambda_3 = 0.20153$ $\lambda_4 = 0.10211$ $\lambda_5 = 0.04943$

$\lambda_6 = 0{,}03455$ $\lambda_7 = 0.03275$ $\lambda_8 = 0.01818$

Setelah didapatkan eignvalue, langkah selanjutnya adalah menghitung matriks eignvector menggunakan fungsi pda *library* Emgu CV didapatkan hasil dari matriks *eigenvector* pada Tabel 3.28 berikut.

Tabel 3.28 Matriks *Eigenvector*

i / j	1	2	3	4	5	6	7	8
1	0.35539	0.35491	0.35652	0.35333	0.35458	0.35619	0.34981	0.34758
2	-0.21589	-0.14241	-0.24864	-0.40133	-0.24890	0.17614	0.46996	0.62960
3	-0.27672	-0.15426	-0.30472	0.21134	0.42655	-0.15671	0.64060	-0.38110
4	-0.24481	-0.47817	-0.14535	0.29119	0.48233	0.11404	-0.43136	0.41687
5	-0.41787	0.44785	-0.01330	0.44255	-0.21118	-0.53116	-0.02484	0.31850
6	-0.23109	0.33099	-0.47970	0.25703	-0.21005	0.66297	-0.14447	-0.19065
7	-0.00697	-0.53314	0.30863	0.51336	-0.54193	0.13804	0.19685	-0.07366
8	0.68009	-0.08339	-0.60955	0.24328	-0.12381	-0.25262	0.00943	0.14339

Kemudian hasil dari matriks *eigenvector* dihitung nilai rata-ratanya dari *eigenvector* setiap *eigenvalue* (rata-rata setiap kolom) menggunakan persamaan berikut.

$$\bar{x}fitur_i = \frac{\sum_{j=1}^{i} x_{ji}}{j} = \frac{x_{11} + x_{21} + \cdots + x_{ji}}{j}$$

Hitung rata-rata nilai untuk seriap N atau setiap frame seperti berikut.

$$\bar{x}fitur_i = \frac{x_{11} + x_{21} + \cdots + x_{ji}}{j}$$

$$\bar{x}fitur_1 = \frac{0.3545 + (-0.4137) + (-0.4048) + 0.2116 + \cdots + -0.2522}{8}$$

$$\bar{x}fitur_1 = \frac{3475}{6400} = 0{,}5431$$

Lakukan perhitungan tersebut sampai $\bar{x}fitur_8$, hasil dari perhitungan nilai rata-rata dapat dilihat pada Tabel 3.29 berikut.

Tabel 3.29 Nilai Fitur Huruf I

Huruf	Nilai Fitur PCA							
I	-0.04473	-0.03220	-0.14201	0.23884	-0.009	0.06336	0.13325	0.15132

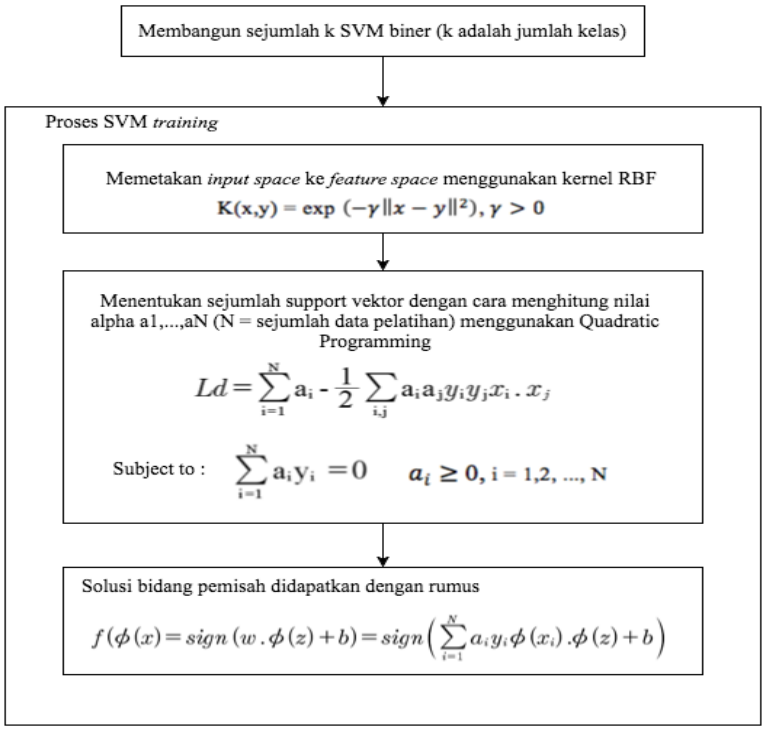
3.8 Analisis Proses SVM

Dalam penelitian ini untuk representasi data menggunakan format *sparse data representation* dengan vektor sebagai inputnya. Format data input untuk klasifikasi SVM dalam penelitian ini adalah [+1 1:0.113965141 2:-0.117635819]. masukan pertama adalah +1 atau -1 menyatakan dua kelas yaitu kelas positif dan kelas negatif. Angka kedua menyatakan dimensi(row_id) dan angka ketiga setelah tanda':' menyatakan bobot fitur, setiap fitur dipisahkan dengan spasi.

3.8.1 Analisis Proses *Training* SVM

Training SVM bertujuan untuk menemukan vektor α dan konstanta b yang akan digunakan untuk menemukan fungsi pemisah (*hyperplane*). Vektor α > 0 dinamkan *support vector* yang diperlukan untuk mewakili fungsi keputusan yang optimal. Konstanta b menentukan lokasi pemisah *relative* terhadap titik asal. Fungsi kernel yang digunakan dalam penelitian ini adalah fungsi kernel RBF yang

didefinisikan dengan persamaan $K(x_i - x) = exp(-\gamma||x - x_i||^2)$. Pada Gambar 3.11 dapat dilihat block diagram proses pelatihan menggunakan SVM.

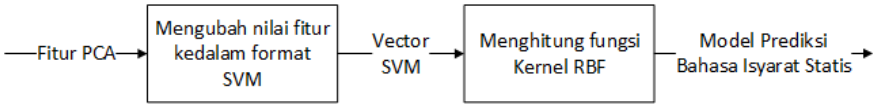


Gambar 3.11 Block Diagram Proses Pelatihan Multiclass SVM

Berdasarkan Gambar 3.11 dapat dijelaskan proses pelatihan SVM diawali dengan membangun sejumlah k svm binner, kemudian hitung setiap k svm biner menggunakan kernel RBF. Lalu menentukan support vector dengan cara menghitung nilai alpha menggunakan *Quadratic Programming*.

3.8.1.1 Analisis Training *One vs All* Bahasa Isyarat Statis

Pada training one vs all untuk bahasa isyarat stais perlu menemukan fungsi pemisah (hyperplane) dimana setiap fungsi dilatih dengan data dari dua kelas. Alur training one vs all untuk bahasa isyarat statis dapat dilihat pada Gambar 3.12 berikut.



Gambar 3.12 Alur Training Bahasa Isyarat Statis

Dalam metode ini diambil contoh kasus yang telah melalui tahap ekstraksi ciri PCA untuk bahasa isyarat dinamis. Bahasa isyarat statis memiliki kasus 24 kelas, yaitu dari bahasa isyarat huruf A sampai huruf Z kecuali huruf J dan Z. Bahasa isyarat yang akan dibahas adalah pencarian *hyperplane* untuk huruf A. Dimana A diberikan label +1 sedangkan yang lainnya diberi label -1. Gambaran label kelas data pelatihan fitur PCA huruf bahasa isyarat statis dapat dilihat pada Tabel 3.30 berikut.

Tabel 3.30 Data Pelatihan Fitur PCA dalam format SVM

Data Ke-	Huruf		
	A	B	C
1	0.14852	0.06914	0.11091
2	0.16297	-0.13190	0.01307
3	0.09838	0.26217	-0.07366
4	-0.14623	0.12434	0.05672
5	-0.07194	-0.02461	0.28383
6	-0.07293	0.07806	0.02082
7	0.17158	-0.08201	0.08002
8	0.07336	0.07207	-0.12839
Label (y)	1	-1	-1

Sebagai contoh pada tahap pelatihan SVM untuk bahasa isyarat statis, yang akan dilakukan perhitungan adalah data fitur dari ekstraksi ciri PCA. Pada tahap selanjutnya yaitu melakukan kernelisasi menggunakan fungsi kernel *RBF* yang didefinisikan dengan menggunakan persamaan $K(x_i - x) = exp(-\gamma||x - x_i||^2)$. Untuk menghitung kernel RBF langkah pertma adalah menghitung nilai x-x_i. Nilai x data pelatihan pada Tabel 3.30 akan digunakan untuk perhitungan kernel atau perhitungan *dot product*. Nilai x didapat dari nilai setiap fitur bahasa isyarat statis. Nilai pada kolom A merupakan nilai x_1, nilai pada kolom B merupakan nilai x_2, nilai pada kolom C merupakan nilai x_3. Perhitungan kernel atau perhitungan dot product dapat dilihat pada Tabel 3.31 berikut.

Tabel 3.31 Proses Perhitungan $x_i - x$

Huruf	$x_i - x$	Perhitungan Bobot
A	$x_1 - x_1$	$\begin{bmatrix} 0.14852 \\ 0.16297 \\ 0.09838 \\ -0.14623 \\ -0.07194 \\ -0.07293 \\ 0.17158 \\ 0.07336 \end{bmatrix} - \begin{bmatrix} 0.14852 \\ 0.16297 \\ 0.09838 \\ -0.14623 \\ -0.07194 \\ -0.07293 \\ 0.17158 \\ 0.07336 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$
B	$x_1 - x_2$	$\begin{bmatrix} 0.14852 \\ 0.16297 \\ 0.09838 \\ -0.14623 \\ -0.07194 \\ -0.07293 \\ 0.17158 \\ 0.07336 \end{bmatrix} - \begin{bmatrix} 0.06914 \\ -0.13190 \\ 0.26217 \\ 0.12434 \\ -0.02461 \\ 0.07806 \\ -0.08201 \\ 0.07207 \end{bmatrix} = \begin{bmatrix} 0.07938 \\ 0.29487 \\ -0.16379 \\ -0.27057 \\ -0.04733 \\ -0.15099 \\ 0.25359 \\ 0.00129 \end{bmatrix}$
C	$x_1 - x_3$	$\begin{bmatrix} 0.14852 \\ 0.16297 \\ 0.09838 \\ -0.14623 \\ -0.07194 \\ -0.07293 \\ 0.17158 \\ 0.07336 \end{bmatrix} - \begin{bmatrix} 0.11091 \\ 0.01307 \\ -0.07366 \\ 0.05672 \\ 0.28383 \\ 0.02082 \\ 0.08002 \\ -0.12839 \end{bmatrix} = \begin{bmatrix} 0.03761 \\ 0.1499 \\ 0.17204 \\ -0.20295 \\ -0.35577 \\ -0.09375 \\ 0.09156 \\ 0.20175 \end{bmatrix}$

Hasil dari perhitungan $x_i - x$ dapat dilihat pada Tabel 3.32 berikut.

Tabel 3.32 Hasil perhitungan $x_i - x$

$x_1 - x_1$	$x_1 - x_2$	$x_1 - x_3$
0	0.07938	0.03761
0	0.29487	0.1499
0	-0.16379	0.17204
0	-0.27057	-0.20295
0	-0.04733	-0.35577
0	-0.15099	-0.09375
0	0.25359	0.09156
0	0.00129	0.20175

Setelah mendapatkan hasil dari perhitungan $x - x_i$, maka langkah selanjutnya adalah menggunakan hasil perhitungan panjang vektor dengan menggunakan rumus berikut.

$$||x - x_i|| = \sqrt{\sum_{i=1}^{n}(x_i - x_i)^2} = \sqrt{(x_1 - x_1)^2 + (x_1 - x_2)^2 + \cdots + (x_n - x_n)^2}$$

Perhitungan panjang vektor dapat dilihat pada Tabel 3.33 berikut.

Tabel 3.33 Proses Perhitungan Panjang Vektor

$\|\|x_1\text{-}x_1\|\|$	$\sqrt{0^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2}$	0
$\|\|x_1\text{-}x_2\|\|$	$\sqrt{0.07938^2 + 0.29487^2 + (-0.16379)^2 + (-0.27057)^2 + (-0.04733)^2 + (-0.15099)^2 + 0.25329^2 + 0.00129^2}$	0.53163
$\|\|x_1\text{-}x_3\|\|$	$\sqrt{0.03781^2 + 0.1499^2 + 0.17204^2 + (-0.20295)^2 + (-0.35577)^2 + (-0.095375)^2 + 0.09156^2 + 20175^2}$	0.52831

Setelah didapatkan nilai dari perhitungan panjang vektor, kemudian dimasukan kedalam persamaan $K(x_i - x) = exp(-\gamma||x - x_i||^2)$ untuk menghitung kernel RBF, dilakukan perhitungan dengan kernel pada semua data dengan y = 0.5. dimana K adalah matriks kernel RBF.

$$K(1,1) = exp\left(-\gamma||x1 - x1||^2\right)$$
$$= \exp(-0.5(0^2)) = \exp(0) = 1$$
$$K(1,2) = exp\left(-\gamma||x1 - x2||^2\right)$$
$$= exp\,(-0.5(0.5316319^2)) = exp\,(-0.14132) = 0.86821$$
$$K(1,3) = exp\left(-\gamma||x2 - x1||^2\right)$$
$$= exp\,(-0.5(0.52832^2)) = exp\,(-0.13956) = 0.86974$$
$$K(2,1) = exp\left(-\gamma||x2 - x1||^2\right)$$
$$= exp\,(-0.5(00.53163^2)) = exp\,(-0.14132) = 0.86821$$
$$K(2,2) = exp\left(-\gamma||x1 - x1||^2\right)$$
$$= exp\,(-0.5(0^2)) = \exp(0) = 1$$
$$K(2,3) = exp\left(-\gamma||x2 - x1||^2\right)$$
$$= exp\,(-0.5(0.55224^2)) = exp\,(-0.15248) = 0.85857$$
$$K(3,1) = exp\left(-\gamma||x2 - x1||^2\right)$$

$= exp\ (-0.5(0.52832^2)) = exp\ (-0.13956) = 0.86974$

$K(3,2) = exp\left(-\gamma\left||x2 - x1|\right|^2\right)$

$= exp\ (-0.5(0.55224^2)) = exp\ (-0.15284) = 0.85857$

$K(3,3) = exp\left(-\gamma\left||x1 - x1|\right|^2\right)$

$= exp\ (-0.5(0^2)) = exp\ (0) = 1$

Setelah dilakukan perhitungan kernel K, maka dapat dilihat hasilnya pada Tabel 3.34 berikut.

Tabel 3.34 Hasil Perhitungan Kernel

K(1,1)	**K(1,2)**	**K(1,3)**
1	0.86821	0.86974
K(2,1)	**K(2,2)**	**K(2,3)**
0.86821	1	0.85857
K(3,1)	**K(3,2)**	**K(3,3)**
0.86974	0.86974	1

Berdasarkan hasil perhitungan kernel, matriks kernel K yang terbentuk adalah sebagai berikut.

$$K = \begin{bmatrix} 1 & 0.86821 & 0.86974 \\ 0.86821 & 1 & 0.85857 \\ 0.86974 & 0.86974 & 1 \end{bmatrix}$$

Tahap selanjutnya adalah melakukan perhitungan terhadap *y*. Untuk nilai *y* adalah nilai dari label atau nilai dari kelas yang telah diberikan sebelumnya. Nilai y tersebut dapat dilihat pada tabel Tabel 3.35 berikut.

Tabel 3.35 Nilai label pada *y*

y_1	y_1	y_3
1	-1	-1

Selanjutnya dilakukan perhitungan $y_i y_j$ sebanyak N data, y_1 sebagai huruf A, y_2 sebagai huruf B dan y_3 sebagai huruf *C*. Berikut perhitungan perhitungan $y_i y_j$.

y_1y_1 = (1)(1) = 1

y_1y_2 = (1)(-1) = -1

y_1y_3 = (1)(-1) = -1

y_2y_1 = (-1)(1) = -1

y_2y_2 = (-1)(-1) = 1

y_2y_3 = (-1)(-1) = 1

y_3y_1 = (-1)(1) = -1

y_3y_2 = (-1)(-1) = 1

y_3y_3 = (-1)(-1) = 1

Setelah didapatkan hasil dari perhitungan terhadap *y*. Hasil dari perhitungan tersebut dapat dilihat pada Tabel 3.36 berikut.

Tabel 3.36 Hasil perhitungan y_iy_j

y_1y_1	y_1y_2	y_1y_3
1	-1	-1
$y2y_1$	y_2y_2	y_2y_3
-1	1	1
y_3y_1	y_3y_2	y_3y_3
-1	1	1

Sehingga matriks yang terbentuk dari perhitungan y_iy_j hasilnya sebagai berikut:

$$y_iy_j = \begin{bmatrix} 1 & -1 & -1 \\ -1 & 1 & 1 \\ -1 & 1 & 1 \end{bmatrix}$$

Kemudian hasil dari perhitungan exp $(-\gamma\|x - x_i\|^2)$yang akan berkorelasi dengan a_ia_j dalam persamaan 2.23. Dengan menggunakan kernel K sebagai pengganti dot-product exp $(-\gamma\|x - x_i\|^2)$dalam persamaan dualitas Lagrange multiplier 2.23, didapatkan:

Maksimalkan:

$Ld_{max} = a_1 + a_2 + a_3 - \frac{1}{2}({a_1}^2 + 1.7364294 a_1 a_2 + 1.7394828 a_1 a_3 + {a_2}^2 + 1.717147 a_2 a_3 + {a_3}^2)$

Syarat 1: $a_1 - a_2 - a_3 = 0$

Syarat 2: $a_1, a_2, a_3 \geq 0$

Dalam fungsi tujuan, suku kedua sudah dikalikan dengan $y_i y_j$. Persamaan tersebut memenuhi standar *Quadaratic programming* sehingga bisa dibantu penyelesainnya dengan solver komersial untuk *Quadratic Programming* (QP). Dengan bantuan perangkat lunak, didapatkan hasil sebagai berikut.

a_1 = 0.5

a_2 = 0.25

a_3 = 0.25

b = 1

Hasil ini menunjukan bahwa semua data *training* adalah *support vector*. Karena nilai a > 0. Sementara nilai b didapatkan dari proses pelatihan yang dilakukan. Setelah menemukan semua a dan b, maka model SVM sudah siap digunakan untuk prediksi dengan menggunakan fungsi pemisah berikut.

$$f(x_{testing}) = \sum_{i=0}^{n} {a_i}^T y_i \, K(x_{training}, x_{testing}) + b$$

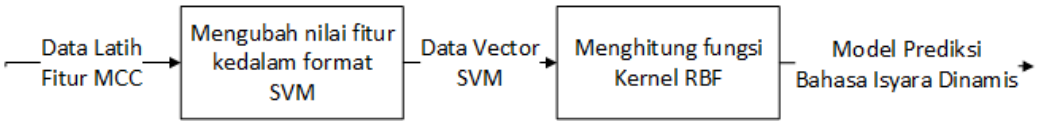
Untuk persamaan *f*($x_{testing}$) adalah sebagai berikut.

$$f(x_{testing}) = \sum_{i=0}^{n} {a_i}^T y_i \, K(x_{training}, x_{testing}) + b$$

$$= \begin{bmatrix} 0.5 \\ 0.25 \\ 0.25 \end{bmatrix}^T \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} \exp\left(-0.5 \left\| x_{training} - x_{testing} \right\|^2\right) + 1$$

3.8.1.2 Analisis *Training One vs All* Bahasa Isyarat Dinamis

Pada training one vs all untuk bahasa isyarat dinamis perlu menemukan fungsi pemisah (hyperplane) dimana setiap fungsi dilatih dengan data dari dua kelas. Alur training one vs all untuk bahasa isyarat statis dapat dilihat pada Gambar 3.12 berikut.



Gambar 3.13 Alur Training SVM Bahasa Isyarat Dinamis

Pada analisis *training one vs all* untuk bahasa isyarat dinamis ini memiliki kasus 2 kelas, yaitu huruf J dan Z. Data fitur yang digunakan untuk bahasa isyarat dinamis adalah hasil dari MCC. Pada Tabel 3.37 dapat dilihat data fitur MCC untuk bahasa isyarat dinamis dalam format SVM.

Tabel 3.37 Data Latih Bahasa Isyarat Dinamis

Data Ke-	Huruf	
	J	Z
1	12	11
2	7	7
3	7	9
4	9	5
5	10	11
6	9	12
7	6	10
8	3	9
Label (y)	1	-1

Pada tahap selanjutnya akan dilakukan kernelisasi menggunakan fungsi kernel RBF yang didefinisikan menggunakan persamaan berikut $K(x_i - x) = exp\left(-\gamma||x - x_i||^2\right)$ untuk menghitung kernel RBF yang pertama dilakukan adalah menghitung $x - x_i$ denga menggunakan nilai x pada Tabel 3.37. Nilai dari huruf J merupakan nilai x_1, nilai dari huruf Z merupakan nilai x_2. Perhitungan $x - x_i$ dapat dilihat pada Tabel 3.38 berikut.

Tabel 3.38 Perhitungan *x* - x_i

Huruf	$x - x_i$	Perhitungan $x - x_i$
J	$x_1 - x_1$	$\begin{bmatrix}12\\7\\7\\9\\10\\9\\6\\3\end{bmatrix} - \begin{bmatrix}12\\7\\7\\9\\10\\9\\6\\3\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix}$
Z	$x_1 - x_2$	$\begin{bmatrix}12\\7\\7\\9\\10\\9\\6\\3\end{bmatrix} - \begin{bmatrix}11\\7\\9\\5\\11\\12\\10\\9\end{bmatrix} = \begin{bmatrix}1\\0\\-2\\4\\-1\\-3\\-4\\-6\end{bmatrix}$

Berdasarkan perhitungan $x - x_i$ pada Tabel 3.38 didapat hasil perhitungan pada Tabel 3.39 berikut.

Tabel 3.39 Hasil Perhitungan $x - x_i$

$x_1 - x_1$	$x_1 - x_2$
0	1
0	0
0	-2
0	4
0	-1
0	-3
0	-4
0	-6

Kemudian dari hasil perhitungan $x - x_i$ akan digunakan untuk menghitung panjang vektor menggunakan rumus berikut.

$$\left|\left|x - x_i\right|\right| = \sqrt{\sum_{i=1}^{n}(x_i - x_i)^2} = \sqrt{(x_1 - x_1)^2 + (x_1 - x_2)^2 + \cdots + (x_n - x_n)^2}$$

Proses perhitungan panjang vektor dapat dilihat pada Tabel 3.40 berikut.

Tabel 3.40 Perhitungan Panjang Vektor

$\left\|\|x_1 - x_1\right\|\|$	$\sqrt{0^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2}$	0
$\left\|\|x_1 - x_2\right\|\|$	$\sqrt{1^2 + 0^2 + (-2)^2 + 4^2 + (-1)^2 + (-3)^2 + (-4)^2 + (-6)^2}$	9.110434

Hasil dari perhitungan panjang vektor pada Tabel 3.40 akan digunakan untuk menghitung kernel RBF menggunakan persamaan $K(x_i - x) = exp(-\gamma\left\|\|x - x_i\right\|\|^2)$. Kemudian lakukan perhitungan kernel RBF dengan $\gamma = 0{,}5$.

$$K(1,1) = exp(-\gamma\left\|\|x_{1-}x_1\right\|\|^2)$$
$$= exp(-0.5(0^2))$$
$$= exp(0) = 1$$

$$K(1,2) = exp(-\gamma\left\|\|x_{1-}x_2\right\|\|^2)$$
$$= exp(-0.5(9.110434^2))$$
$$= exp(-41.5)$$
$$= 0.000000000000000000947936$$

$$K(2,1) = exp(-\gamma\left\|\|x_{2-}x_1\right\|\|^2)$$
$$= exp(-0.5(9.110434^2))$$
$$= exp(-41.5)$$
$$= 0.000000000000000000947936$$

$$K(2,2) = exp(-\gamma\left\|\|x_{1-}x_1\right\|\|^2)$$
$$= exp(-0.5(0^2))$$
$$= exp(0) = 1$$

Setelah didapatkan hasil perhitungan kernel RBF, maka nilai untuk K(1,1), K(1,2), K(2,1),K(2,2) dapat dilihat pada Tabel 3.34 berikut.

Tabel 3.41 Hasil perhitungan Kernel RBF

K(1,1)	**K(1,2)**
1	0.000000000000000000947936
K(2,1)	**K(2,2)**
0.000000000000000000947936	1

Setelah dilakukan perhitungan, maka matriks kernel K yang terbentuk adalah sebagai berikut.

$$K = \begin{bmatrix} 1 & 0.00000000000000000947936 \\ 0.00000000000000000947936 & 1 \end{bmatrix}$$

Tahap selanjutnya adalah melakukan perhitungan terhadap *y*. Untuk nilai *y* adalah nilai dari label atau nilai dari kelas yang telah diberikan sebelumnya. Nilai y tersebut dapat dilihat pada tabel Tabel 3.35 berikut.

Tabel 3.42 Nilai label pada *y*

y_1	y_1
1	-1

Selanjutnya dilakukan perhitungan $y_i y_j$ sebanyak N data, y_1 sebagai huruf A, y_2 sebagai huruf B dan y_3 sebagai huruf *C*. Berikut perhitungan $y_i y_j$.

y_1y_1 = (1)(1) = 1

y_1y_2 = (1)(-1) = -1

y_2y_1 = (-1)(1) = -1

y_2y_2 = (-1)(-1) = 1

Setelah didapatkan dilakukan perhitungan terhadap *y*. Hasil dari perhitungan tersebut dapat dilihat pada Tabel 3.36 berikut.

Tabel 3.43 Hasil perhitungan y_iy_j

y_1y_1	y_1y_2
1	-1
y_2y_1	y_2y_2
-1	1

Sehingga matriks yang terbentuk dari perhitungan $y_i y_j$ sebagai berikut:

$$y_i y_j = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

Kemudian hasil dari perhitungan exp $(-\gamma\|x - x_i\|^2)$yang akan berkorelasi dengan $a_i a_j$ dalam persamaan 2.23. Dengan menggunakan kernel K sebagai pengganti dot-product exp $(-\gamma\|x - x_i\|^2)$dalam persamaan dualitas Lagrange multiplier 2.23, didapatkan:

Maksimalkan:

$$Ld_{max} = a_1 + a_1 + a_3 - \frac{1}{2}({a_1}^2 + 0.000000000000000000947936 a_1 a_2 + {a_2}^2)$$

Syarat 1: $a_1 - a_2 = 0$

Syarat 2: $a_1, a_2 \geq 0$

Dalam fungsi tujuan, suku kedua sudah dikalikan dengan $y_i y_j$. Persamaan tersebut memenuhi standar *Quadaratic programming* sehingga bisa dibantu penyelesainnya dengan solver komersial untuk *Quadratic Programming* (QP). Dengan bantuan perangkat lunak, didapatkan hasil sebagai berikut.

a_1 = 1

a_2 = 1

b = 1

Hasil ini menunjukan bahwa semua data *training* adalah *support vector*. Karena nilai a > 0. Sementara nilai b didapatkan dari proses pelatihan yang dilakukan. Setelah menemukan semua a dan b, maka model SVM sudah siap digunakan untuk prediksi dengan menggunakan fungsi pemisah berikut.

$$f(x_{testing}) = \sum_{i=0}^{n} {a_i}^T \, y_i \, K(x_{training}, x_{testing}) + b$$

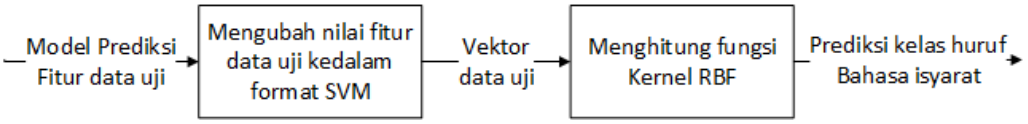
Untuk persamaan f(x) adalah sebagai berikut.

$$f(x_{testing}) = \sum_{i=0}^{n} {a_i}^T \, y_i \, K(x_{training}, x_{testing}) + b$$

$$= \begin{bmatrix} 1 \\ 1 \end{bmatrix}^T \begin{bmatrix} 1 \\ -1 \end{bmatrix} \exp\left(-0.5 \left\| x_{training} - x_{testing} \right\|^2\right) + 1$$

3.8.2 Analisis Proses SVM *Testing*

Setelah mendapatkan model prediksi, selanjutnya dapat menentukan data bahasa isyarat masuk ke dalam kelas positif atau negatif dengan nilai w dan *hyperplane* dari data *training*. Jika nilai hasil uji lebih besar dari nilai *hyperplane* maka huruf bahasa isyarat tersebut masuk dalam kelas positif, jika lebih kecil dari nilai *hyperplane* maka huruf bahasa isyarat tersebut masuk dalam kelas negatif. Berikut alur proses dapat dilihat pada Gambar 3.14 berikut.



Gambar 3.14 Alur Proses SVM *Testing*

Seperti yang telah dilakukan sebelumnya, SVM sebenernya digunakan untuk klasifikasi biner (dua kelas), dalam kasus *multiclass* SVM yang digunakan adalah dengan mengkombinasikan beberapa SVM biner tersebut. Metode "*One Vs All"* salah satunya. Dibangun *k* SVM biner, dengan *k* adalah jumlah kelas. Untuk setiap model klasifikasi ke-i dilatih dengan menggunakan keseluruhan data.

3.8.2.1 Analisis *Multiclass One vs All* Bahasa Isyarat Statis

Pengujian *multiclass one vs all* bahasa isyrat statis menggunakan data test bahasa isyarat statis yang akan diuji dan dihitung menggunakan data model prediksi yang sudah dilatih dengan bahasa isyrat statis. Dalam kasus ini adalah bahasa isyarat statis huruf A, B, *C*, dan A_{test} sebagai data uji bahasa isyarat statis. Berikut gambaran sebagian vektor data latih dan data uji fitur PCA yang digambarkan dalam bentuk Tabel 3.44 berikut.

Tabel 3.44 Data Pengujian Bahasa Isyarat Statis

Data Ke-	Huruf			
	A	B	C	A_{test}
1	0.14852	0.06914	0.11091	0.20556
2	0.16297	-0.13190	0.01307	0.01021
3	0.09838	0.26217	-0.07366	-0.00222
4	-0.14623	0.12434	0.05672	0.09730
5	-0.07194	-0.02461	0.28383	-0.16166
6	-0.07293	0.07806	0.02082	-0.00222
7	0.17158	-0.08201	0.08002	0.00044
8	0.07336	0.07207	-0.12839	0.21685
Label (y)	1	-1	-1	1

Berikut model prediksi bahasa isyarat statis pada Tabel 3.45 yang didapatkan pada proses pelatihan.

Tabel 3.45 Model Prediksi Bahasa Isyarat Statis

y_i = +1	y_i = -1	Model SVM
A	Bukan A	$f(x_{testing}) = \sum_{i=0}^{n} {a_i}^T\, y_i\, K(x_{training}, x_{testing}) + b$ $= \begin{bmatrix} 0.5 \\ 0.25 \\ 0.25 \end{bmatrix}^T \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} \exp\left(-0.5\left\|x_{training} - x_{testing}\right\|^2\right) + 1$
B	Bukan B	$f(x_{testing}) = \sum_{i=0}^{n} {a_i}^T\, y_i\, K(x_{training}, x_{testing}) + b$ $= \begin{bmatrix} 0.5 \\ 0.25 \\ 0.25 \end{bmatrix}^T \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} \exp\left(-0.5\left\|x_{training} - x_{testing}\right\|^2\right) + 1$
C	Bukan C	$f(x_{testing}) = \sum_{i=0}^{n} {a_i}^T\, y_i\, K(x_{training}, x_{testing}) + b$ $= \begin{bmatrix} 0.5 \\ 0.25 \\ 0.25 \end{bmatrix}^T \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} \exp\left(-0.5\left\|x_{training} - x_{testing}\right\|^2\right) + 1$

Berdasarkan Tabel 3.45 dapat dijelaskan $x_{training}$ merupakan data vektor *training*, misalnya data huruf A pada Tabel 3.44 dan $x_{testing}$ merupakan data vektor yang akan diuji, contoh data uji huruf A_{test} padaTabel 3.44. Selanjutnya dilakukan perhitungan fungsi pemisah dengan melakukan kernelisasi menggunakan fungsi kernel RBF yang didefinisikan sebagai $K(x_{training}, x_{testing}) = exp(-\gamma||x - y||^2), \gamma > 0$, dapat dilihat pada Tabel 3.46 berikut.

Tabel 3.46 Proses Perhitungan x-x_i Untuk K($x_{training}$, $x_{testing}$)

Huruf	Xtraining - Xtesting
A	$x_{1-}x_{test} = \begin{bmatrix} 0.14852 \\ 0.16297 \\ 0.09838 \\ -0.14623 \\ -0.07194 \\ -0.07293 \\ 0.17158 \\ 0.07336 \end{bmatrix} - \begin{bmatrix} 0.20556 \\ 0.01021 \\ -0.00222 \\ 0.09730 \\ -0.16166 \\ -0.00222 \\ 0.00044 \\ 0.21685 \end{bmatrix} = \begin{bmatrix} -0.0570 \\ 0.15276 \\ 0.1006 \\ -0.24353 \\ 0.08972 \\ -0.07071 \\ -0.17114 \\ -0.14349 \end{bmatrix}$
B	$x_2 - x_{test} = \begin{bmatrix} 0.06914 \\ -0.13190 \\ 0.26217 \\ 0.12434 \\ -0.02461 \\ 0.07806 \\ -0.08201 \\ 0.07207 \end{bmatrix} - \begin{bmatrix} 0.20556 \\ 0.01021 \\ -0.00222 \\ 0.09730 \\ -0.16166 \\ -0.00222 \\ 0.00044 \\ 0.21685 \end{bmatrix} = \begin{bmatrix} -0.13642 \\ -0.14211 \\ 0.26439 \\ 0.02704 \\ 0.13705 \\ 0.08028 \\ -0.08245 \\ -0.14478 \end{bmatrix}$

C	$x_3 - x_{test} = \begin{bmatrix} 0.11091 \\ 0.01307 \\ -0.07366 \\ 0.05672 \\ 0.28383 \\ 0.02082 \\ 0.08002 \\ -0.12839 \end{bmatrix} - \begin{bmatrix} 0.20556 \\ 0.01021 \\ -0.00222 \\ 0.09730 \\ -0.16166 \\ -0.00222 \\ 0.00044 \\ 0.21685 \end{bmatrix} = \begin{bmatrix} -0.09465 \\ 0.00286 \\ -0.07144 \\ -0.04058 \\ 0.44549 \\ 0.02304 \\ 0.07958 \\ -0.34524 \end{bmatrix}$

Hasil dari perhitungan x-xi untuk K($x_{training}$, $x_{testing}$) untuk setiap huruf dapat dilihat pada Tabel 3.47 berikut.

Tabel 3.47 Hasil Perhitungan x-x_i Untuk K($x_{training}$, $x_{testing}$)

$x_1 - x_{test}$	$x_2 - x_{test}$	$x_3 - x_{test}$
-0.05704	-0.13642	-0.09465
0.15276	-0.14211	0.00286
0.1006	0.26439	-0.07144
-0.24353	0.02704	-0.04058
0.08972	0.13705	0.44549
-0.07071	0.08028	0.02304
0.17114	-0.08245	0.07958
-0.14349	-0.14478	-0.34524

Setelah mendapatkan hasil dari perhitungan *x*-x_i untuk K($x_{training}$, $x_{testing}$), maka langkah selanjutnya adalah menghitung panjang vektor pada Tabel 3.48 berikut.

Tabel 3.48 Hasil Perhitungan Panjang Vektor ($x_{training}$,$x_{testing}$)

$\|\|x_1 - x_{test}\|\|$	$\sqrt{\begin{matrix}(-0.5704)^2 + 0.15276^2 \\ +0.1006^2 + (-0.24353)^2 \\ +0.08972^2 + (-0.07071^2 \\ + 0.17114^2 + (-0.14339)^2\end{matrix}}$	0.3986781
$\|\|x_2 - x_{test}\|\|$	$\sqrt{\begin{matrix}(-0.13642)^2 + (-0.14211)^2 \\ +0.26439^2 + 0.02704^2 \\ +0.13705^2 + 0.08028^2 \\ + (-0.08245)^2 + (-0.14478)^2\end{matrix}}$	0.4030208

$\|\|x_2 - x_{test}\|\|$	$\sqrt{\begin{matrix}(-0.09465)^2 + 0.00286^2 \\ +(-0.07144)^2 + (-0.04058)^2 \\ +0.44549^2 + 0.02304^2 \\ +\ 0.07958^2 + \ (-0.34524)^2\end{matrix}}$	0.583295

Setelah didapatkan nilai dari perhitungan panjang vektor, kemudia dimasukkan kedalam persamaan $K(x_{training}, x_{testing})exp(-\gamma||x-y||^2)$.

Hitung $K(x_{training}, x_{testing})$ untuk huruf A.

$$K(x_{training}, x_{testing}) = exp(-\gamma||x_1 - x_{test}||^2)$$
$$= exp(-0.5(0.3986781^2))$$
$$= exp\ (-0.0794721) = 0.9236038$$

Hitung $K(x_{training}, x_{testing})$ untuk huruf B.

$$K(x_{training}, x_{testing}) = exp(-\gamma||x_2 - x_{test}||^2)$$
$$= exp(-0.5(0.4030208^2))$$
$$= exp\ (-0.081212867) = 0.92199741$$

Hitung $K(x_{training}, x_{testing})$ untuk huruf C.

$$K(x_{training}, x_{testing}) = exp(-\gamma||x_3 - x_{test}||^2)$$
$$= exp(-0.5(0.583295^2))$$
$$= exp\ (-0.170116514) = 0.843567$$

Setelah didatkan nilai dari fungsi kernel, hitung nilai prediksi fungsi pemisah untuk setiap huruf seperti berikut.

Perhitungan nilai Prediksi untuk huruf A, maka didapatkan:

$$f(x_{testing}) = \begin{bmatrix}0.5\\0.25\\0.25\end{bmatrix}^T \begin{bmatrix}1\\-1\\-1\end{bmatrix} \exp\left(-0.5\left\|x_{training} - x_{testing}\right\|^2\right) + 1$$
$$= 0.25\ (0.9236038) + 1$$
$$= 1.23090094$$

Perhitungan nilai Prediksi untuk huruf B, maka didapatkan:

$$f(x_{testing}) = \begin{bmatrix}0.5\\0.25\\0.25\end{bmatrix}^T \begin{bmatrix}1\\-1\\-1\end{bmatrix} \exp\left(-0.5\left\|x_{training} - x_{testing}\right\|^2\right) + 1$$

$$= 0.25\ (0.92199741) + 1$$

$$= \ 1.23049935$$

Perhitungan nilai Prediksi untuk huruf C, maka didapatkan:

$$f(x_{testing}) = \begin{bmatrix} 0.5 \\ 0.25 \\ 0.25 \end{bmatrix}^T \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} \exp\left(-0.5\left\|x_{training} - \ x_{testing}\right\|^2\right) + 1$$

$$= 0.25\ (0.843567) + 1$$

$$= \ 1.210891631$$

Nilai prediksi huruf bahasa isyarat statis dari setiap fungsi pemisah diberikan dalam matriks berikut. Setiap kolom dari matriks mewakili nilai prediksi dari setiap fungsi pemisah.

γ = [1.23090094 1.23049935 1.210891631]

Berdasarkan nilai maksimum γ, maka hasil dari klasifikasi *multiclass* adalah bahwa data *testing* merupakan bagian dari kelas A.

3.8.2.2 Analisis *Multiclass One vs All* Bahasa Isyarat Dinamis

Pada pengujian SVM menggunakan *multiclass one vs all* untuk bahasa isyarat dinamis akan menggunakan data uji bahasa isyrat dinamis huruf J dan Z. Data uji akan dihitung dan diuji berdasarkan model prediksi bahasa isyarat dinamis yang sudah didapatkan sebelumnya. Gambaran vektor data latih bahasa isyarat dinamis dapat dilihat pada Tabel 3.49 berikut.

Tabel 3.49 Data Pengujian Bahasa Isyarat Dinamis

Data Ke-	Huruf		
	J	Z	J_{test}
1	12	11	12
2	7	7	7
3	7	9	6
4	9	5	10
5	10	11	9
6	9	12	7
7	6	10	6
8	3	9	5
Label (y)	1	-1	1

Berikut model prediksi bahasa isyarat dinamis pada Tabel 3.50 yang didapatkan pada proses pelatihan.

Tabel 3.50 Model Prediksi Bahasa Isyarat Dinamis

$y_i = +1$	$y_i = -1$	Model SVM
J	Bukan J	$f(x_{testing}) = \sum_{i=0}^{n} {a_i}^T\, y_i\, K(x_{training}, x_{testing}) + b$ $= \begin{bmatrix}1\\1\end{bmatrix}^T \begin{bmatrix}1\\-1\end{bmatrix} \exp\left(-0.5\left\|x_{training} - x_{testing}\right\|^2\right) + 1$
Z	Bukan Z	$f(x_{testing}) = \sum_{i=0}^{n} {a_i}^T\, y_i\, K(x_{training}, x_{testing}) + b$ $= \begin{bmatrix}1\\1\end{bmatrix}^T \begin{bmatrix}1\\-1\end{bmatrix} \exp\left(-0.5\left\|x_{training} - x_{testing}\right\|^2\right) + 1$

Pada Tabel 3.50 menunjukan semua fungsi pemisah dari ke 2 kelas yang berarti ada 2 fungsi pemisah yang memisahkan satu kelas dengan 2 kelas lainnya. Kemudian $x_{training}$ merupakan data vektor *training*, misalnya data huruf J pada Tabel 3.49 dan xt_{esting} merupakan data vektor testing, misalnya data uji huruf J_{test} pada Tabel 3.49. Selanjutnya dilakukan perhitungan fungsi pemisah dengan melakukan kernelisasi menggunakan fungsi kernel RBF dapat dilihat pada Tabel 3.51 berikut.

Tabel 3.51 Perhitungan $x_{training}$ - $x_{testing}$

Huruf	Perhitungan $x_{training}$ - $x_{testing}$
J	$x_{1-}x_{test} = \begin{bmatrix}12\\7\\7\\9\\10\\9\\6\\3\end{bmatrix} - \begin{bmatrix}12\\7\\6\\10\\9\\7\\6\\5\end{bmatrix} = \begin{bmatrix}0\\0\\1\\-1\\1\\2\\0\\-2\end{bmatrix}$
Z	$x_2 - x_{test} = \begin{bmatrix}11\\7\\9\\5\\11\\12\\10\\9\end{bmatrix} - \begin{bmatrix}12\\7\\6\\10\\9\\7\\6\\5\end{bmatrix} = \begin{bmatrix}-1\\0\\2\\-5\\2\\5\\4\\4\end{bmatrix}$

Hasil perhitungan $x_{training} - x_{testing}$ dapat dilihat pada tabel

Tabel 3.52 Hasil Perhitungan $x_{training}$ - $x_{testing}$

$x_1 - x_{test}$	$x_2 - x_{test}$
0	-1
0	0
1	2
-1	-5
1	2
2	5
0	4
-2	4

Kemudian hasil dari perhitungan pada Tabel 3.52 akan digunakan untuk menghitung panjang vektor $\|x_{training} - x_{testing}\|$, dapat dilihat pada tabel

Tabel 3.53 Perhitungan Panjang Vektor

$\|\|x_1 - x_{test}\|\|$	$\sqrt{0^2 + 0^2 + 1^2 + (-1)^2 + 1^2 + 2^2 + 0^2 + (-2)^2}$	3.316625
$\|\|x_2 - x_{test}\|\|$	$\sqrt{(-1)^2 + 0^2 + 2^2 + (5)^2 + 2^2 + 5^2 + 4^2 + 4^2}$	9.539392

Setelah nilai dari panjang vektor didapatkan, kemudian hasil dari panjang vektor dimasukan kedalam persamaan $K(x_{training}, x_{testing})$ seperti berikut.

Perhitungan fungsi kernel untuk huruf J:

$$K(x_{training}, x_{testing}) = exp(-\gamma||x_1 - x_{test}||^2)$$
$$= exp(-0.5(3.316625^2))$$
$$= \exp(-5{,}5) = 0.004086771$$

Perhitungan fungsi kernel untuk huruf Z:

$$K(x_{training}, x_{testing}) = exp(-\gamma||x_2 - x_{test}||^2)$$
$$= exp(-0.5(9.539392^2))$$
$$= \exp(-45.5) = 0.00000000000000000000174$$

Setelah didapatkan perhitungan fungsi kernel masing-masing huruf, maka hitung nilai prediksi untuk setiap huruf seperti berikut.

1. Perhitungan nilai prediksi untuk huruf J

$$f(x_{testing}) = \sum_{i=0}^{n} {a_i}^T \, y_i \, K(x_{training}, x_{testing}) + b$$

$$f(x_{testing}) = \begin{bmatrix}1\\1\end{bmatrix}^T \begin{bmatrix}1\\-1\end{bmatrix} \exp\left(-0.5\left\|x_{training} - x_{testing}\right\|^2\right) + 1$$

$$= 1.004086771$$

2. Perhitungan nilai prediksi untuk huruf Z

$$f(x_{testing}) = \sum_{i=0}^{n} {a_i}^T \, y_i \, K(x_{training}, x_{testing}) + \text{b}$$

$$f(x_{testing}) = \begin{bmatrix}1\\1\end{bmatrix}^T \begin{bmatrix}1\\-1\end{bmatrix} \exp\left(-0.5\left\|x_{training} - x_{testing}\right\|^2\right) + 1$$

$$= 1.00000000000000000174$$

Nilai prediksi huruf bahasa isyarat dinamis dari setiap fungsi pemisah diberikan dalam matriks berikut.

γ = [1.004086771 1.00000000000000000174]

Berdasarkan nilai maksimum *y*, setiap kolom dari matriks mewakili nilai prediksi dari setiap fungsi pemisah. Hasil dari klasifikasi *multiclass* adalah bahwa data *testing* merupakan bagian dari kelas J.

3.9 Analisis Kebutuhan Sistem

Analisis kebutuhan sistem terdiri dari analisis kebutuhan Non-Fungsional dan analisis kebutuhan fungsional. Adapun analisis kebutuhan Non Fungsional dan Fungsional adalah sebagai berikut:

3.9.1 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional menjelaskan hal-hal yang diperlukan untuk membangun sebuah sistem. Analisis kebutuhan non fungsional terdiri dari analisis kebutuhan perangkat keras dan analisis kebutuhan perangkat lunak.

3.9.1.1 Analisis Kebutuhan Perangkat Keras

Analisis kebutuhan perangkat keras pada penelitian ini berdasarkan pada kebutuhan perangkat keras pada saat membangun program. Adapun spesifikasi

minimum perangkat keras untuk menjalankan Microsoft Visual Studio 2017 [16]. Kebutuhan perangkat keras dapat dilihat pada Tabel 3.54 berikut.

Tabel 3.54 Kebutuhan Perangkat Keras untuk Pengembang

No	Perangkat Keras	Spesifikasi
1	Processor	Intel Dual Core 1.8 Ghz
2	RAM	2 GB
3	HDD	40 GB *space* kosong

Adapun spesifikasi minimum perangkat keras untuk menjalankan program berdasarkan pada spesifikasi minimum Visual C++ Redistributable [17]. Untuk menjalankan library EmguCV dibutuhkan visual C++ Redistributable. Kebutuhan perangkat keras dapat dilihat pada Tabel 3.55 berikut.

Tabel 3.55 Spesifikasi Minimum Perangkat Keras untuk Pengguna

No	Perangkat Keras	Spesifikasi
1	Processor	Intel Pentium 4 900 Mhz
2	RAM	128 MB
3	HDD	25 MB *space* kosong

3.9.1.2 Analisis Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat keras pada penelitian ini berdasarkan pada kebutuhan perangkat keras pada saat membangun program. Adapun spesifikasi minimum perangkat lunak untuk menjalankan Microsoft Visual Studio 2017 [16]. Spesifikasi minimum kebutuhan perangkat lunak dapat dilihat pada Tabel 3.56 berikut.

Tabel 3.56 Spesifikasi Minimum Kebutuhan Perangkat Lunak untuk Membangun Program

No	Perangkat Lunak	Perangkat Lunak
1	Sistem Operasi	Windows 7
2	Compiler	Microsoft Visual Studio 2017
3	*Library*	EmguCV

Adapun kebutuhan perangkat lunak untuk menjalankan program yang terdiri dari Sistem operasi dan Visual C++ Redistributable. Penggunaan visual c++ digunakan sebagai syarat untuk dapat menjalankan *library* EmguCV. Spesifikasi minimum kebutuhan perangkat lunak untuk menjalankan program dapat dilihat pada Tabel 3.57 berikut.

Tabel 3.57 Spesifikasi Minimum Kebutuhan Perangkat Lunak Untuk Menjalankan Program

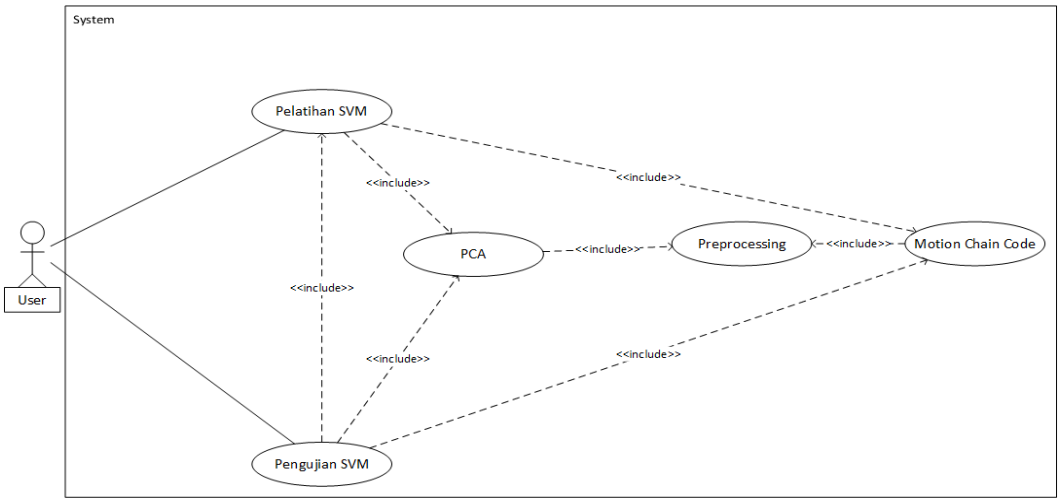
No	Perangkat Lunak	Perangkat Lunak
1	Sistem Operasi	Windows 7
2	Runtime	Visual C++ Redistributable

3.9.2 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menjelaskan hal-hal yang harus dapat dilakukan oleh perangkat lunak yang nantinya akan dibangun. Analisis kebutuhan fungsional pada penelitian kali ini akan menggunakan UML 2.0 sebagai acuan. Pada analisis kebutuhan fungsional terdapat *use case diagram*, *use case scenario*, *activity diagram*, *class diagram* dan *sequence diagram*.

3.9.2.1 *Use Case Diagram*

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, membuat sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor 32 adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu [8]. Berikut adalah *use case diagram* untuk aplikasi yang akan dibangun.



Gambar 3.15 *Use Case Diagram*

3.9.2.2 Identifikasi Aktor

Aktor yang teridentifikasi dan terlibat dalam sistem ini hanya satu yakni pengguna. Pengguna disini adalah penyandang tunarungu atau tunawicara yang akan mengoperasikan sistem ini untuk berkomunikasi dengan orang normal.

3.9.2.3 Definisi *Use Case*

Definisi *Use Case* dapat dilihap pada Tabel 3.58 yang mendeskripsikan setiap *use case*.

Tabel 3.58 Definisi Use Case

No	Use Case	Deskripsi
1	Preprocessing	Proses untuk melakukan pengolahan citra
2	Motion Chain Code	Proses untuk fitur pola gerakan tangan
2	PCA	Proses untuk mendapatkan fitur PCA
3	Pelatihan SVM	Proses pelatihan SVM
4	Pengujian SVM	Proses pengujian SVM

3.9.2.4 *Use Case Scenario*

Use Case Scenario merupakan penjelasan proses-proses yang terjadi pada aplikasi, dengan tujuan untuk lebih memudahkan dalam pembangunana sistem.

3.9.2.4.1 *Use Case Scenario Preprocessing*

Use Case Scenario preprocessing menjelaskan interaksi antara aktor dengan *use case preprocessing*. Untuk lebih lengkapnya akan dijelaskan pada Tabel 3.59 berikut.

Tabel 3.59 *Use Case Scenario Preprocessing*

Nama *Use Case*	Preprocessing	
***Use Case* terkait**	PCA, *Motion Chain Code*, Pengujian SVM, Pelatihan SVM	
Tujuan	Menginput video dan mengolahnya menjadi beberapa frame	
Kondisi Awal	Citra masih berupa kumpulan frame (video)	
Kondisi Akhir	Menampilkan beberapa potongan frame bahasa isyarat yang akan digunakan untuk ekstraksi ciri	
Aktor	Pengguna	
Alur Utama	**Tahap**	**Aksi**
	1	Pennggguna menginput video isyarat tangan
	2	Sistem menampilkan video
	3	Sistem melakukan proses pendeteksian warna kulit, hand tracking, *resize*, dan pengambilan *frame*
	4	Sistem menentukan gerakan bahasa isyarat termasuk statis atau dinamis.

3.9.2.4.2 *Use Case Scenario Motion Chain Code*

Use Case Scenario motion chain code menjelaskan interaksi antara aktor dengan *use case motion chain code*. Untuk lebih lengkapnya akan dijelaskan pada Tabel 3.59 berikut.

Tabel 3.60 *Use Case Scenario* Motion Chain Code

Nama *Use Case*	Motion Chain Code	
***Use Case* terkait**	*Preprocessing*, Pengujian SVM, Pelatihan SVM	
Tujuan	Mendeskripsikan pola gerakan tangan	
Kondisi Awal	Citra masih berupa kumpulan frame (video)	
Kondisi Akhir	Mendapatkan nilai dari pola gerakan tangan	
Aktor	Pengguna	
Alur Utama	**Tahap**	**Aksi**
	1	Pennggguna menginput video isyarat tangan
	2	Sistem menampilkan video
	3	Sistem melakukan *preprocessing*
	4	Jika bahasa isyarat dinamis sistem melakukan tahapan-tahapan *motion chain code* dan mendapatkan nilai MCC.

3.9.2.4.3 *Use Case Scenario* PCA

Use Case Scenario ekstraksi ciri menjelaskan interaksi antara aktor dengan use case ekstraksi ciri. Untuk lebih lengkapnya akan dijelaskan pada Tabel 3.61 berikut.

Tabel 3.61 *Use Case Scenario* PCA

Nama *Use Case*	PCA	
***Use Case* terkait**	*preprocessing*, Pengujian SVM, Pelatihan SVM	
Tujuan	Mengekstrak ciri/informasi dari objek didalam citra	
Kondisi Awal	Pengambilan frame bahasa isyarat hasil *preprocessing*	
Kondisi Akhir	Hasil ekstraksi ciri PCA berupa data vektor ciri bahasa isyarat	
Aktor	Pengguna	
Alur Utama	**Tahap**	**Aksi**
	1	Pengguna menekan tombol open video
	2	Sistem melakukan *preprocessing*
	3	Jika bahasa isyarat statis sistem melakukan perhitungan PCA berdasarkan data masukan

3.9.2.4.4 Use Case Skenario Pelatihan

Use Case Scenario Pelatihan menjelaskan interaksi antara aktor dengan use case Pelatihan. Untuk lebih lengkapnya akan dijelaskan pada Tabel 3.62 berikut.

Tabel 3.62 *Use Case Scenario* Pelatihan

Nama *Use Case*	Pelatihan	
***Use Case* terkait**	*Preprocessing*, PCA, *Motion chain Code*, Pengujian SVM	
Tujuan	Melakukan pemodelan data latih	
Kondisi Awal	Data latih dan data kelas	
Kondisi Akhir	Hasil hitung data bobot kelas berupa nilai vektor	
	Pengguna	
Alur Utama	**Tahap**	**Aksi**
	1	Pengguna memilih menu pelatihan
	2	Pengguna menekan tombol open video
		Sistem melakukan *preprocessing*
	3	Jika gerakan bahasa isyarat statis maka sistem akan mengambil frame dan melakukan ekstraksi ciri PCA
	4	Sistem melakukan pelatihan untuk bahasa isyarat statis
	3	Jika gerakan bahasa isyarat dinamis maka sistem akan mengambil koordinat pusat dan melakukan proses MCC
	4	Sistem melakukan pelatihan untuk isyarat dinamis

3.9.2.4.5 *Use Case Scenario* Pengujian

Use Case Scenario Pengujian menjelaskan interaksi antara aktor dengan use case pengujian. Untuk lebih lengkapnya akan dijelaskan pada Tabel 3.63 berikut.

Tabel 3.63 *Use Case Scenario* Pengujian

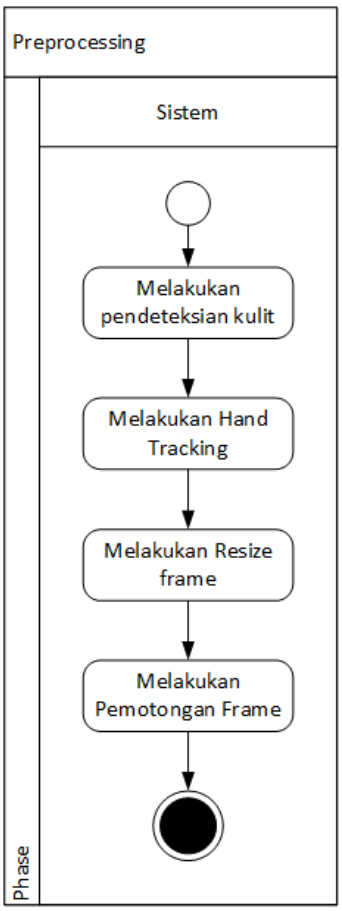
Nama *Use Case*	Pengujian	
***Use Case* terkait**	*preprocessing*, PCA, *motion chain code*, Pelatihan SVM	
Tujuan	Menghitung dan mengkonversi hasil dari video bahasa isyarat berupa teks	
Kondisi Awal	Video bahasa isyarat	
Kondisi Akhir	Menampilkan video bahasa isyarat dan menampilkan karakter digital hasil konversi	
	Pengguna	
Alur Utama	**Tahap**	**Aksi**
	1	Pengguna memilih menu pengujian
	2	Pengguna menekan tombol open video
	3	Sistem melakukan *preprocessing*
	4	Jika gerakan bahasa isyarat termasuk statis maka sistem akan melakukan proses ekstraksi ciri PCA
	5	Sistem akan melakukan pengujian dengan bahasa isyarat statis yang sudah dilatih dan keluaran huruf berupa teks
	6	Jika gerakan bahasa isyarat termasuk dinamis maka sistem akan melakukan ekstraksi ciri *motion chain code*.
	7	Sistem akan melakukan pengujian dengan bahasa isyarat dinamis yang sudah dilatih dan keluaran huruf berupa teks

3.9.2.5 *Activity Diagram*

Activity diagram merupakan gambaran alur kerja atau *workflow* atau aktivitas dari sebuah sistem [8]. *Activity diagram* berfungsi memodelkan bagaimana alur *use case* bekerja. Berikut adalah *activity diagram preprocessing*, ekstraksi ciri PCA, MCC, pelatihan dan pengujian.

3.9.2.5.1 *Activity Diagram Preprocessing*

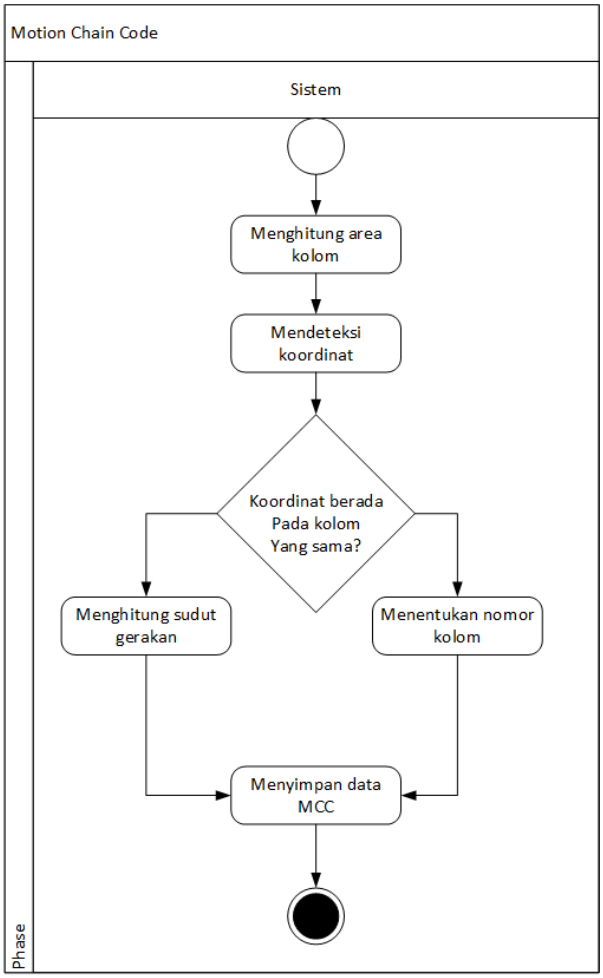
Activity diagram preprocessing dimulai dengan melakukan pendeteksian kulit sampai mendapatkan frame yang siap untuk diolah pada tahapan ekstraksi ciri. Untuk lebih jelasnya dapat dilihat pada Gambar 3.16 berikut.



Gambar 3.16 ***Activity Diagram Preprocessing***

3.9.2.5.2 *Activity Diagram Motion Chain Code*

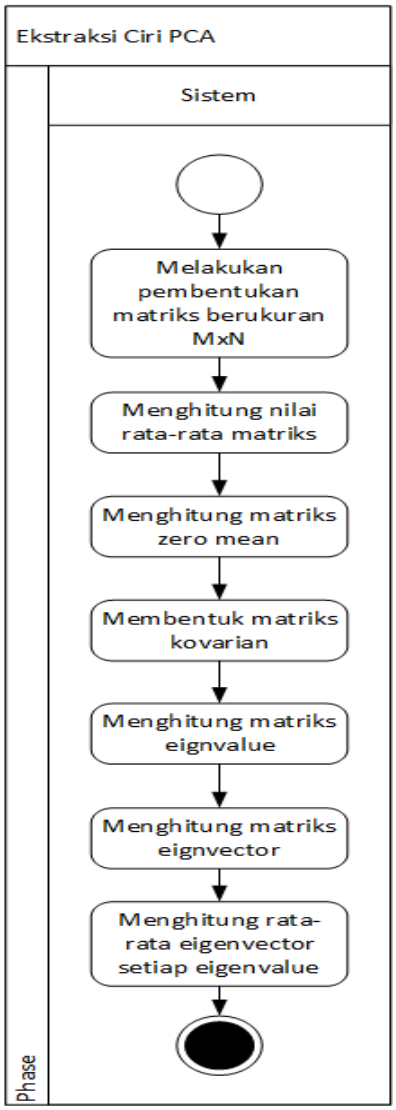
Activity diagram motion chain code dimulai dengan nilai koordinat hasil *preprocessing* hingga menghasilkan nilai pola gerakan tangan. Untuk lebih jelasnya dapat dilihat pada Gambar 3.18 berikut.



Gambar 3.17 ***Activity Diagram Motion Chain Code***

3.9.2.5.3 *Activity Diagram* Ekstraksi Ciri PCA

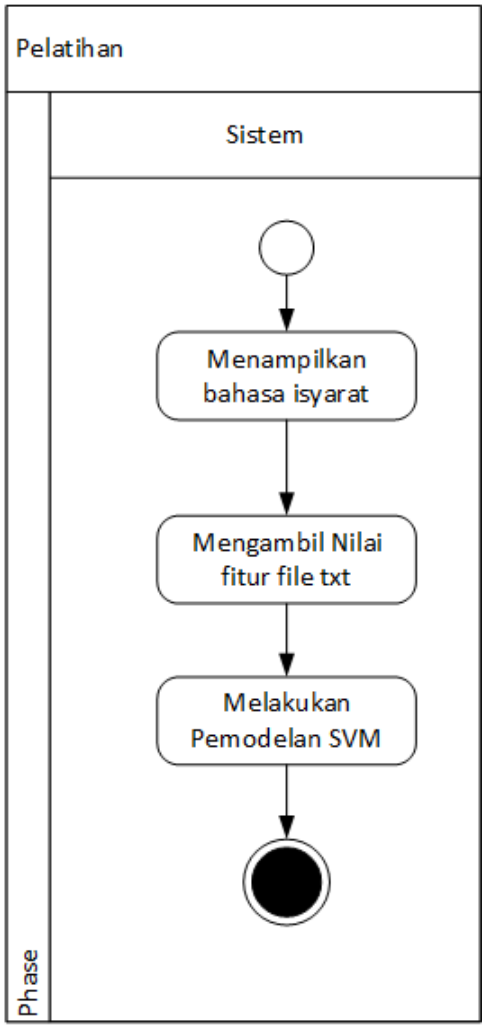
Activity diagram ekstraksi ciri dimulai dengan potongan-potongan hasil *preprocessing* hingga menghasilkan nilai fitur ciri PCA. Untuk lebih jelasnya dapat dilihat pada Gambar 3.18 berikut.



Gambar 3.18 *Activity Diagram* Ekstraksi Ciri PCA

3.9.2.5.4 *Activity Diagram* Pelatihan

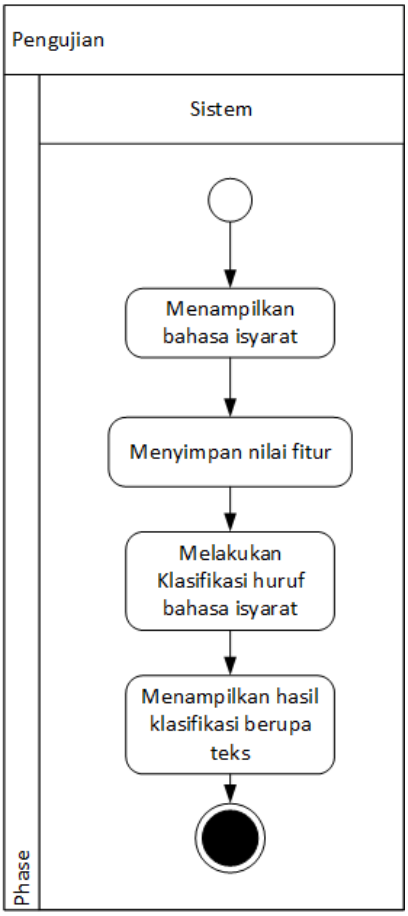
Activity diagram pelatihan dimulai dengan vektor ciri dengan format SVM hingga pelatihan SVM selesai dilakukan. Untuk lebih jelasnya dapat dilihat pada Gambar 3.19 berikut.



Gambar 3.19 Activity Diagram Pelatihan

3.9.2.5.5 *Activity Diagram* Pengujian

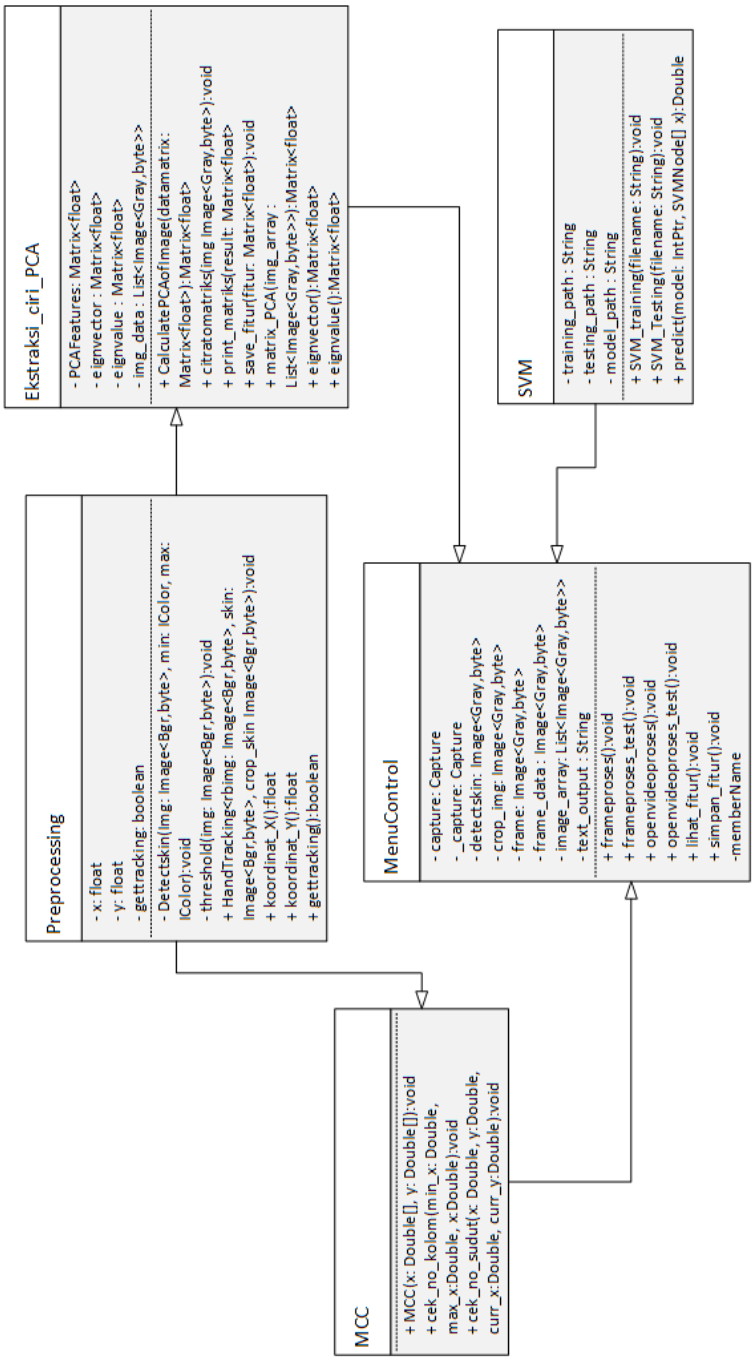
Activity diagram pengujian dimulai pengguna memilih pengujian dan sistem akan menampilkan hasil berupa karakter digital hasil pengujian svm. Untuk lebih jelasnya dapat dilihat pada Gambar 3.20 berikut.



Gambar 3.20 *Activity Diagram* Pengujian

3.9.2.6 *Class Diagram*

Class diagram digunakan untuk menampilkan kelas-kelas dan paket-paket di dalam sistem aplikasi simulasi analisis sentimen yang akan dibangun. *Class diagram* memberikan gambaran sistem secara statis dan relasi antar kelas seperti relasi antar kelas *preprocessing*, *traning*, dan yang lainnya [8]. Beberapa diagram akan menampilkan subset dari kelas-kelas dan relasinya. Adapun *class diagram*-nya dapat dilihat pada Gambar 3.21 berikut.



Gambar 3.21 *Class Diagram*

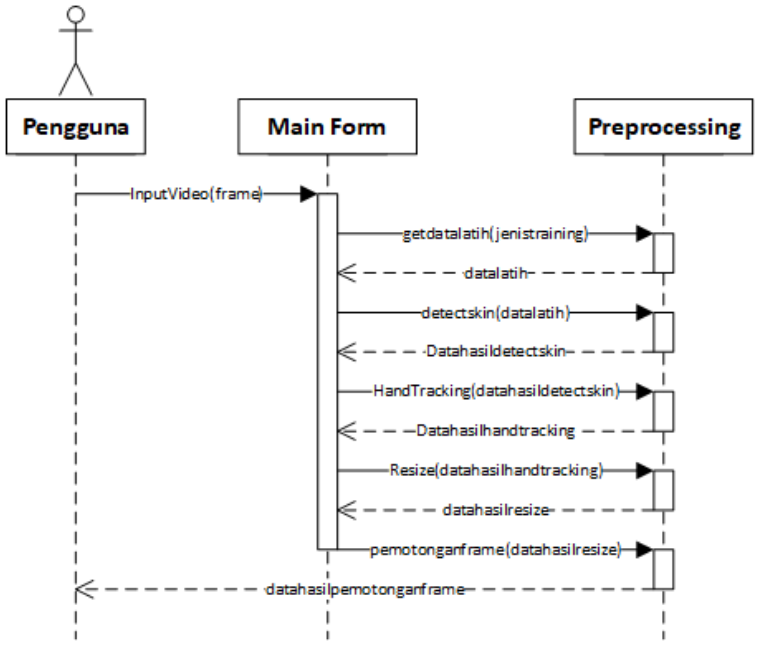
3.9.2.7 *Sequence Diagram*

Sequence diagram memodelkan bagaimana sistem yang sedang dibangun akan menangani tugasnya. Diagram ini menggambarkan semua urutan interaksi pada bagian-bagian dari sistem. Dengan menggunakan *sequence diagram*, pengembang bisa menjelaskan interaksi apa yang akan dipanggil ketika sebuah use

case dieksekusi dan dalam urutan seperti apa interaksi-interaksi tersebut akan terjadi [8]. Pada diagram ini, dimensi vertikal mempresentasikan waktu. Bagian paling atas dari diagram menjadi titik awal dan waktu berjalan ke bawah sampai dengan bagian dasar dari diagram.

3.9.2.7.1 *Sequence Diagram Preprocessing*

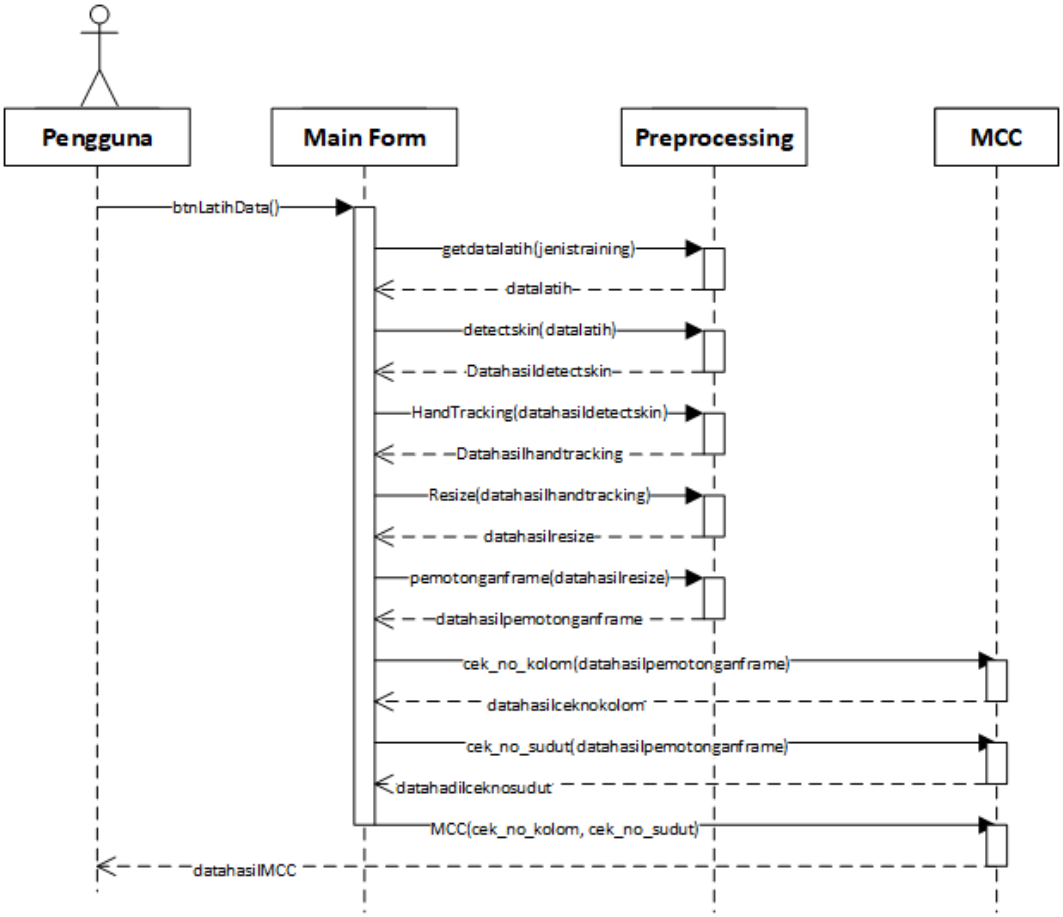
Sequence diagram preprocessing dapat dilihat pada Gambar 3.22 berikut.



Gambar 3.22 ***Sequence Diagram Preprocessing***

3.9.2.7.2 *Sequence Diagram Motion Chain Code*

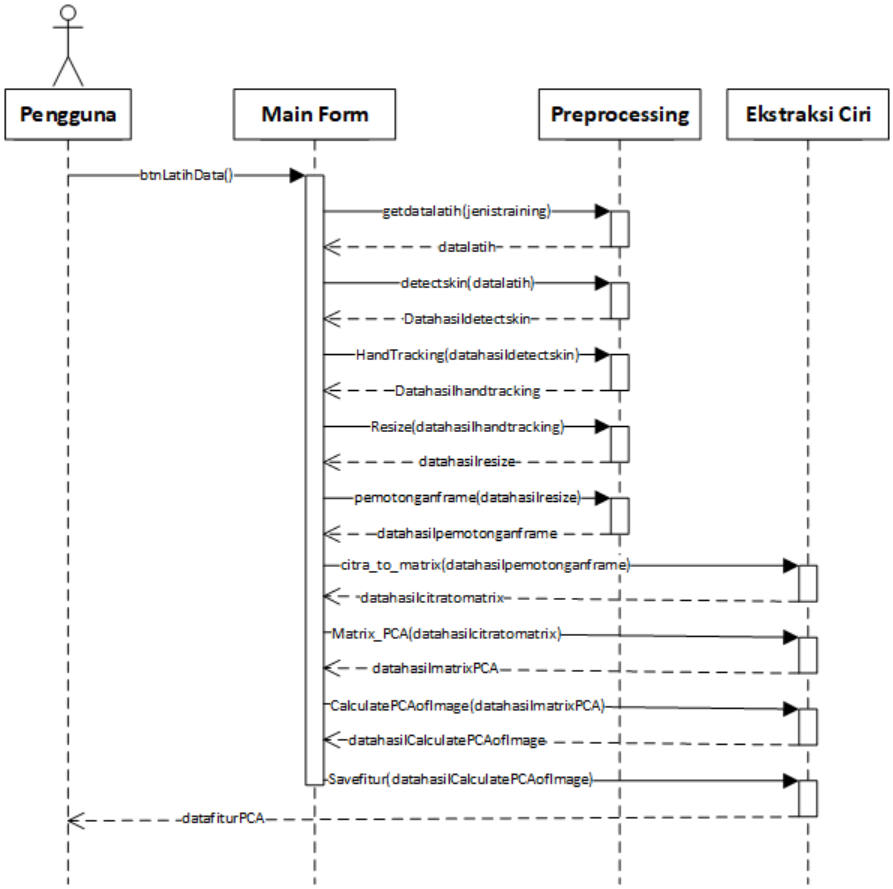
Sequence diagram motion chain code dapat dilihat pada Gambar 3.23 berikut.



Gambar 3.23 ***Sequence Diagram Motion Chain Code***

3.9.2.7.3 *Sequence Diagram* Ekstraksi Ciri PCA

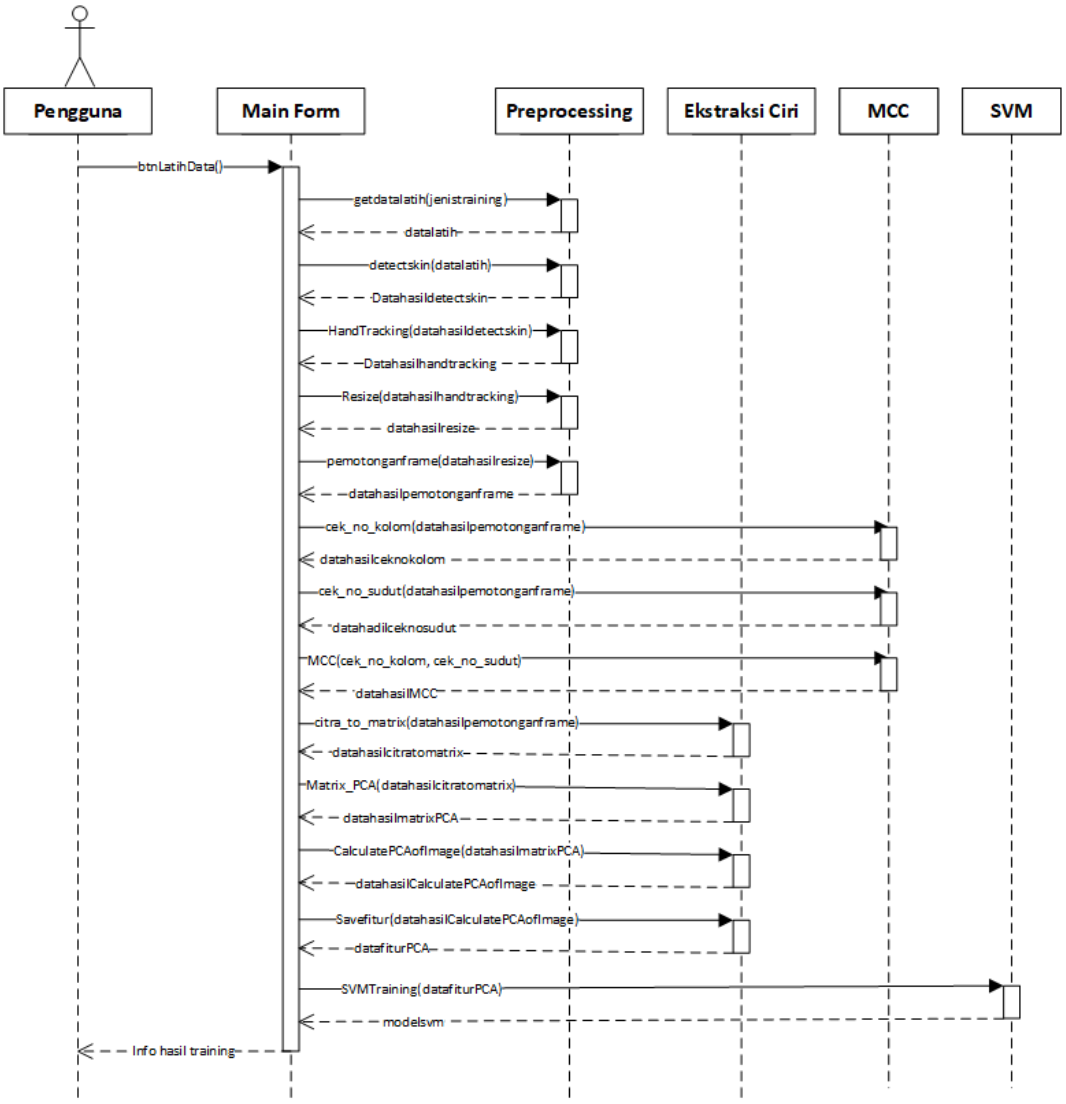
Sequence diagram ekstraksi ciri PCA dapat dilihat pada Gambar 3.24 berikut.



Gambar 3.24 *Sequence Diagram* Ekstraksi Ciri PCA

3.9.2.7.4 *Sequence Diagram* Pelatihan

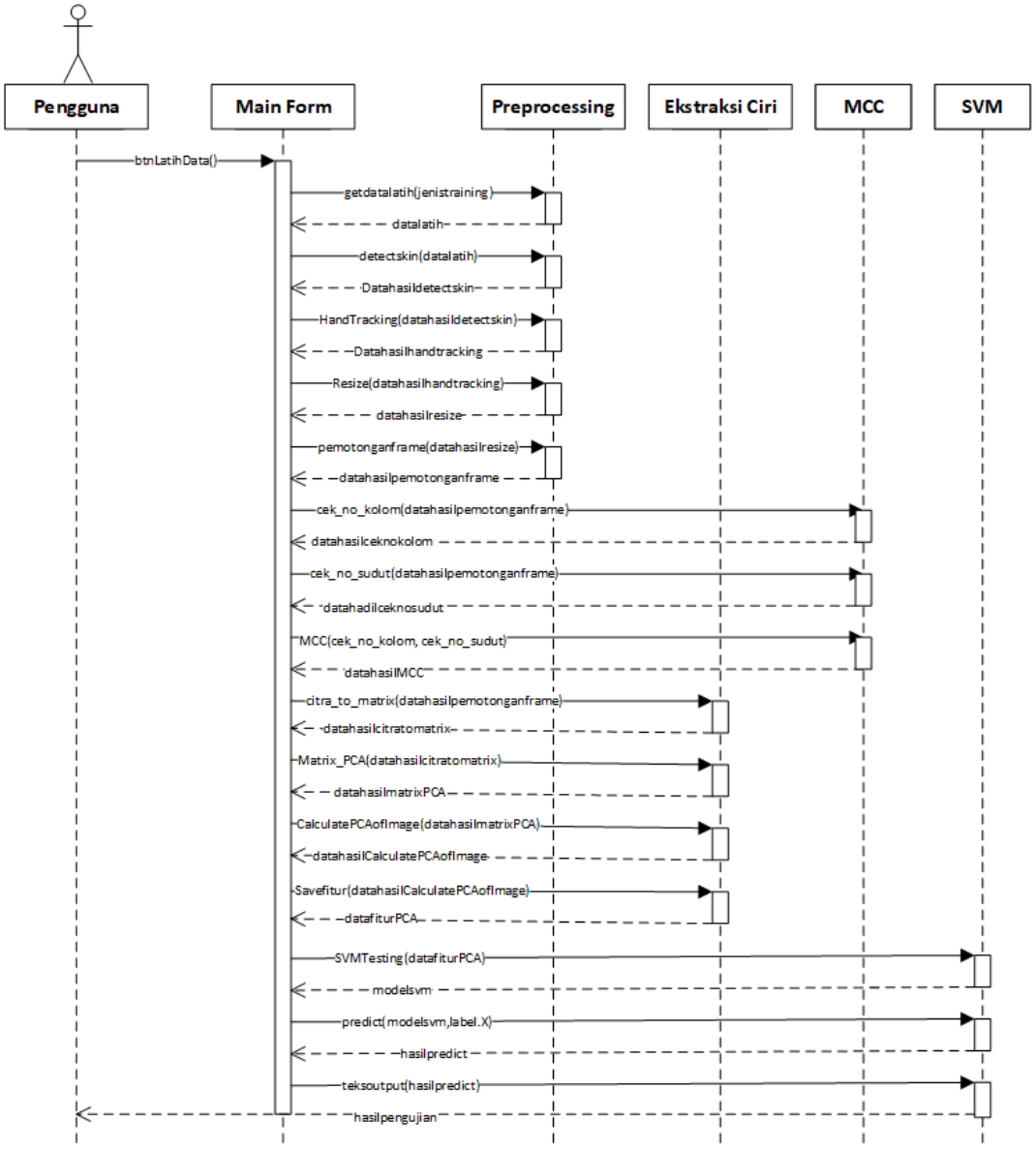
Sequence diagram pelatihan dapat dilihat pada Gambar 3.25 berikut.



Gambar 3.25 *Sequence Diagram* Pelatihan

3.9.2.7.5 *Sequence Diagram* Pengujian

Sequence diagram pengujian dapat dilihat pada Gambar 3.26Gambar 3.22 berikut.



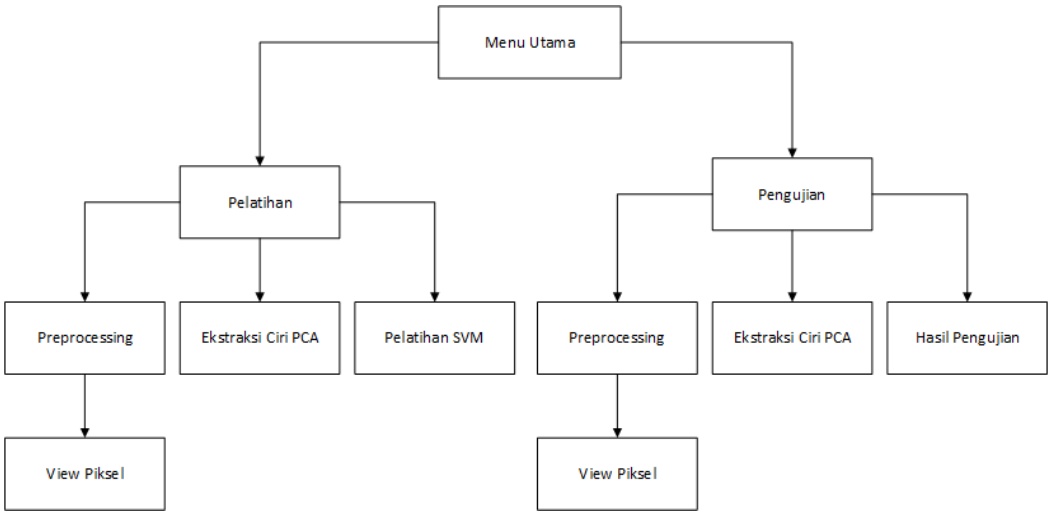
Gambar 3.26 *Sequence Diagram* Pengujian

3.10 Perancangan Sistem

Dalam perancangan sistem ini, dibuat beberapa perancangan seperti struktur menu, perancangan antar muka, pesan (*Alert)* dan jaringan semantik.

3.10.1 Struktur Menu

Perancangan struktur menu merupakan gambar alur pemakaian sistem, sehingga sistem yang dibangun mudah dipahami dan mudah digunakan. Berikut adalah perancangan struktur menu yang akan diterapkan pada aplikasi adapun perancangan struktur menu pada Gambar 3.27 berikut.



Gambar 3.27 Struktur Menu

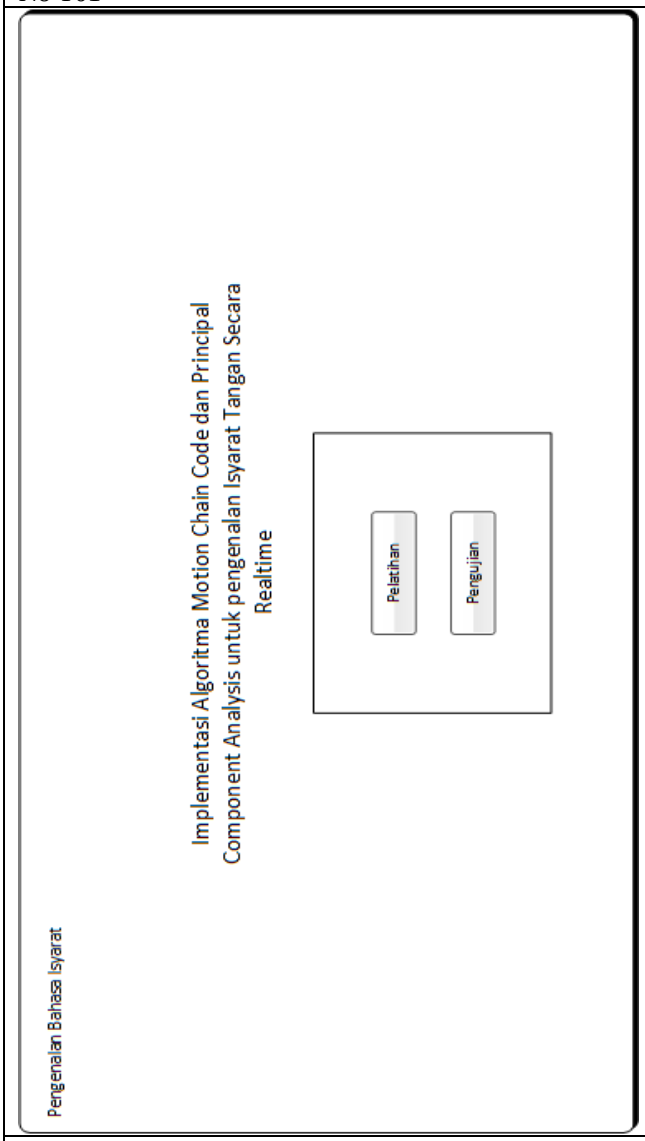
3.10.2 Perancangan Antarmuka

Perancangan antarmuka merupakan rancangan untuk menentukan tampilan dari sistem yang akan dibuat. Berikut adalah perancangan pada aplikasi yang akan dibangun.

3.10.2.1 Perancangan Antarmuka Menu Utama

Pada perancangan antarmuka menu utama adalah pada saat pertama kali menjalankan aplikasi. Pada perancangan halaman ini juga ditampilkan button pelatihan dan pengujian.

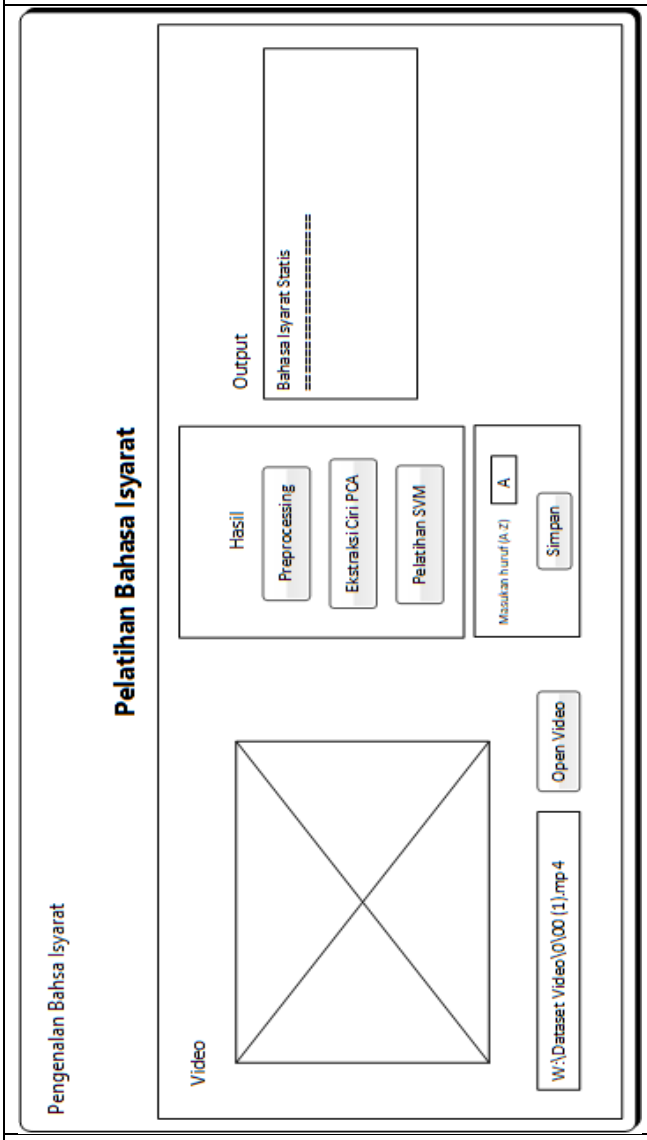
Tabel 3.64 Perancangan Antarmuka Menu Utama

<table>
<tr><td>No T01</td><td>- Klik Pelatihan menuju T02
- Klik Pengujian menuju T03</td></tr>
<tr><td colspan="2">
</td></tr>
<tr><td colspan="2">Ukuran : 800x600
Font : Segoe UI</td></tr>
</table>

3.10.2.2 Perancangan Antarmuka Pelatihan

Pada perancangan antarmuka pelatiha adalah perancangan halaman pelatihan dimana pada *form* ini dapat dilakukan penambahan data latih dan melihat setiap proses yang dilakukan.

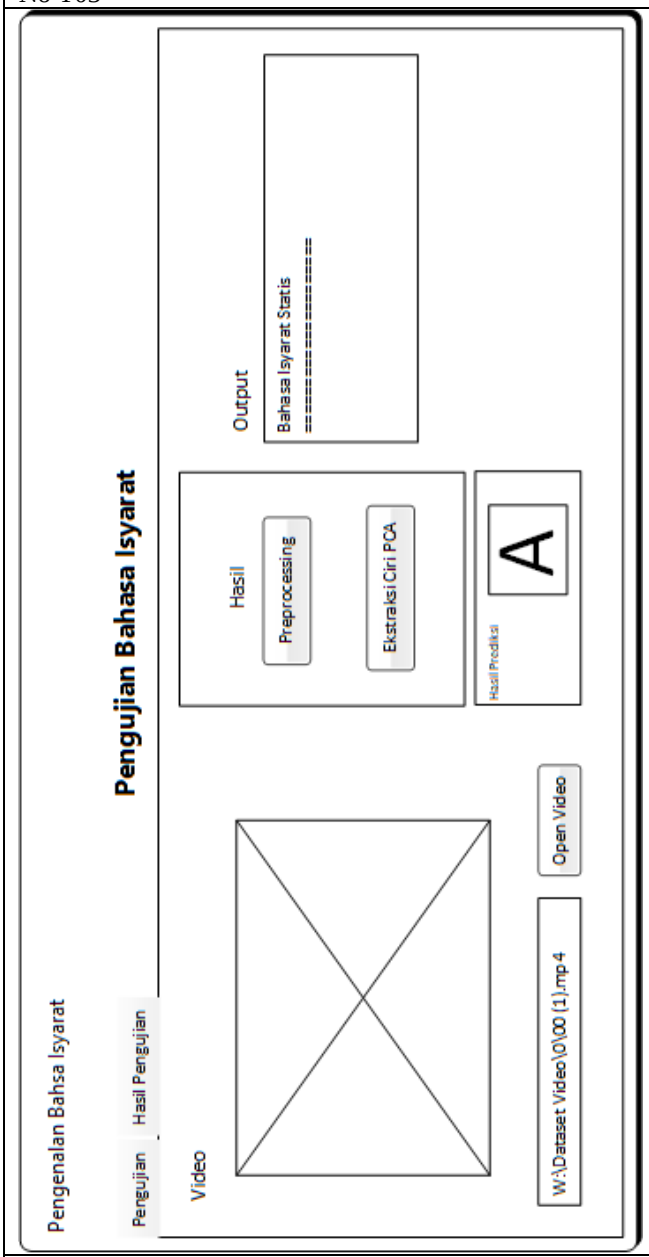
Tabel 3.65 Perancangan Antarmuka Pelatihan

<table>
<tr><td>No T02</td><td rowspan="2">- Klik tombol Open Video untuk mengambil data masukan video dari folder
- Klik Tombol Simpan untuk menyimpan fitur kedalam bentuk .txt
- Klik Preprocessing untuk menuju T04
- Klik Ekstraksi Ciri PCA untuk menuju T05
- Klik Pelatihan SVM menuju T06</td></tr>
<tr><td>
</td></tr>
<tr><td colspan="2">Ukuran : 800x600
Font : Segoe UI</td></tr>
</table>

3.10.2.3 Perancangan Antarmuka Pengujian

Pada perancangan antarmuka pengujian adalah perancangan halaman pengujian dimana dalam form ini dapat dilakukan pengujian setiap video data uji dan pengujian akurasi.

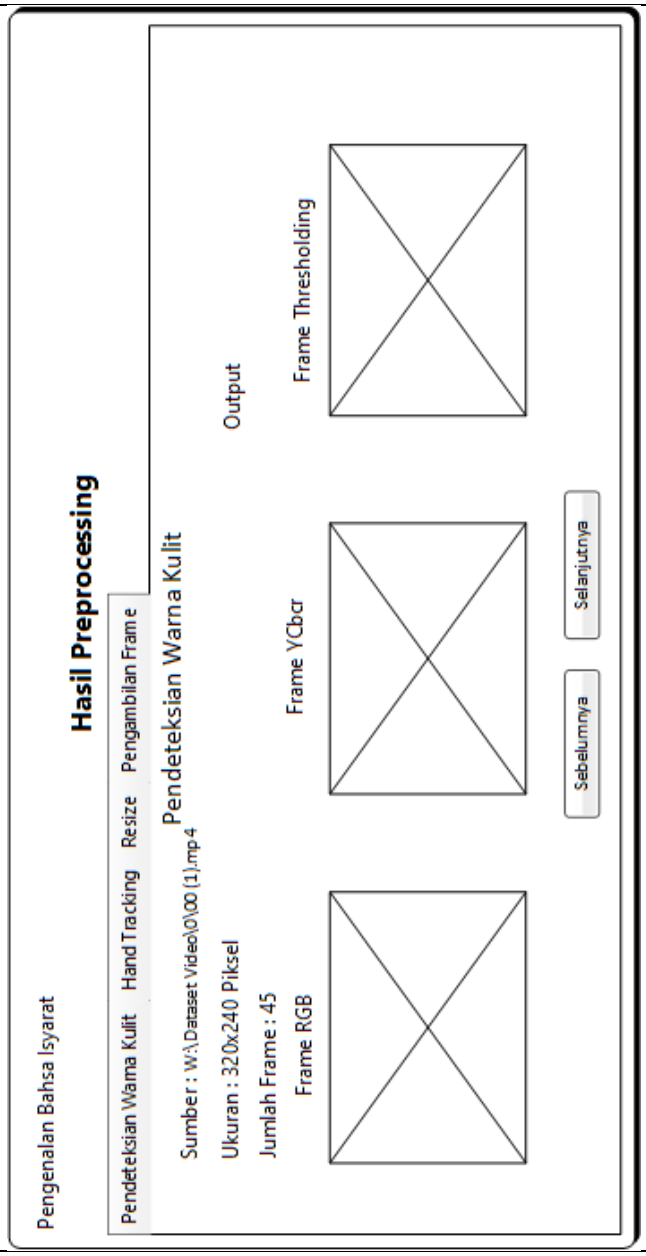
Tabel 3.66 Perancangan Antarmuka Pengujian

No T03	- Klik tombol Open Video untuk mengambil data masukan video dari folder dan melakukan pengujian - Klik Preprocessing untuk menuju T04 - Klik Ekstraksi Ciri PCA untuk menuju T05
	
Ukuran : 800x600 Font : Segoe UI	

3.10.2.4 Perancangan Antarmuka *Preprocessing*

Pada perancangan antarmuka *preprocessing* adalah perancangan halaman preprocessing dinama pengguna dapat melihat setiap proses dari *preprocessing*.

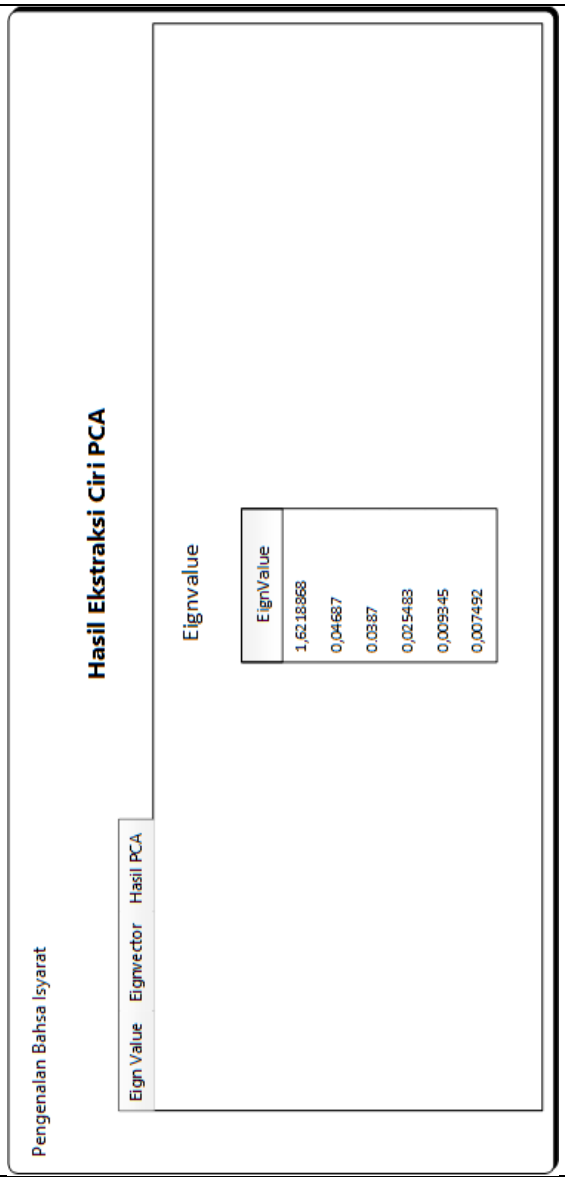
Tabel 3.67 Perancangan Antarmuka *Preprocessing*

<table>
<tr><td>No T04</td><td rowspan="2">- Klik Selanjutnya untuk melihat frame yang selanjutnya
- Klik Sebelumnya untuk melihat frame sebelumnya
- Klik Gambar menuju T06
- Klik Hand Tracking untuk melihat proses hand tracking
- Klik Resize untuk melihat proses resize
- Klik Pengambilan Frame untuk melihat frame yang diambil</td></tr>
<tr><td>
</td></tr>
<tr><td colspan="2">Ukuran : 800x600
Font : Segoe UI</td></tr>
</table>

3.10.2.5 Perancangan Antarmuka Ekstraksi Ciri PCA

Pada perancangan antarmuka ekstraksi ciri PCA adalah perancangan halaman ekstraksi ciri PCA dinama pengguna dapat melihat nilai dari setiap proses dari ekstrakci ciri PCA.

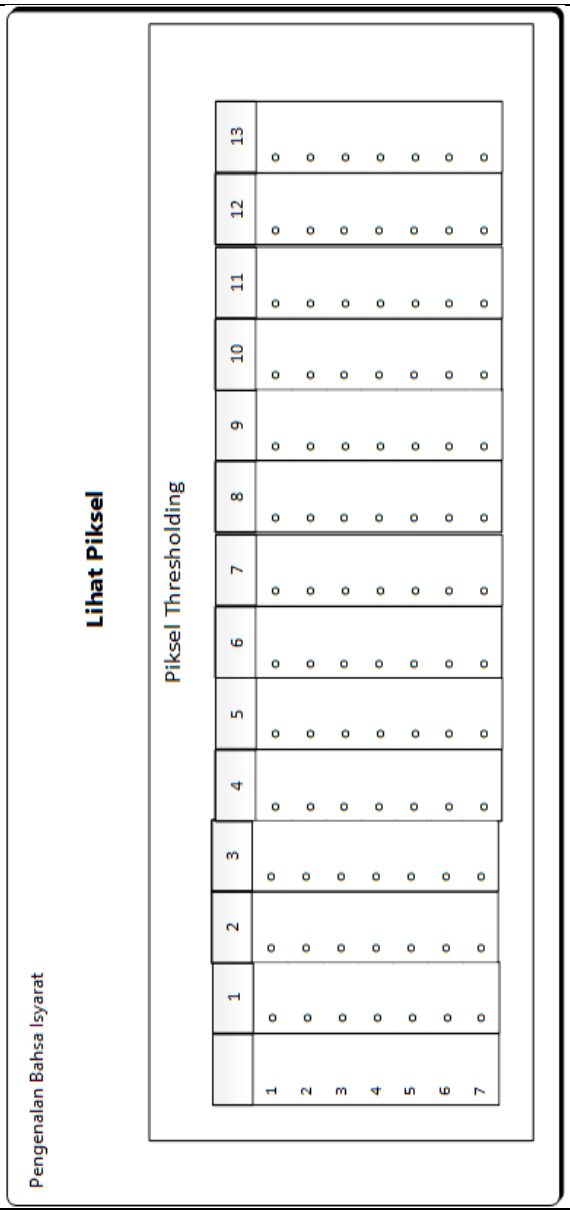
Tabel 3.68 Perancangan Antarmuka Ekstraksi Ciri PCA

No T05	- Klik tab eignvector untuk melihat nilai eignvector Klik hasil PCA untuk melihat hasil PCA
	
Ukuran : 800x600 Font : Segoe UI	

3.10.2.6 Perancangan Antarmuka Lihat Piksel

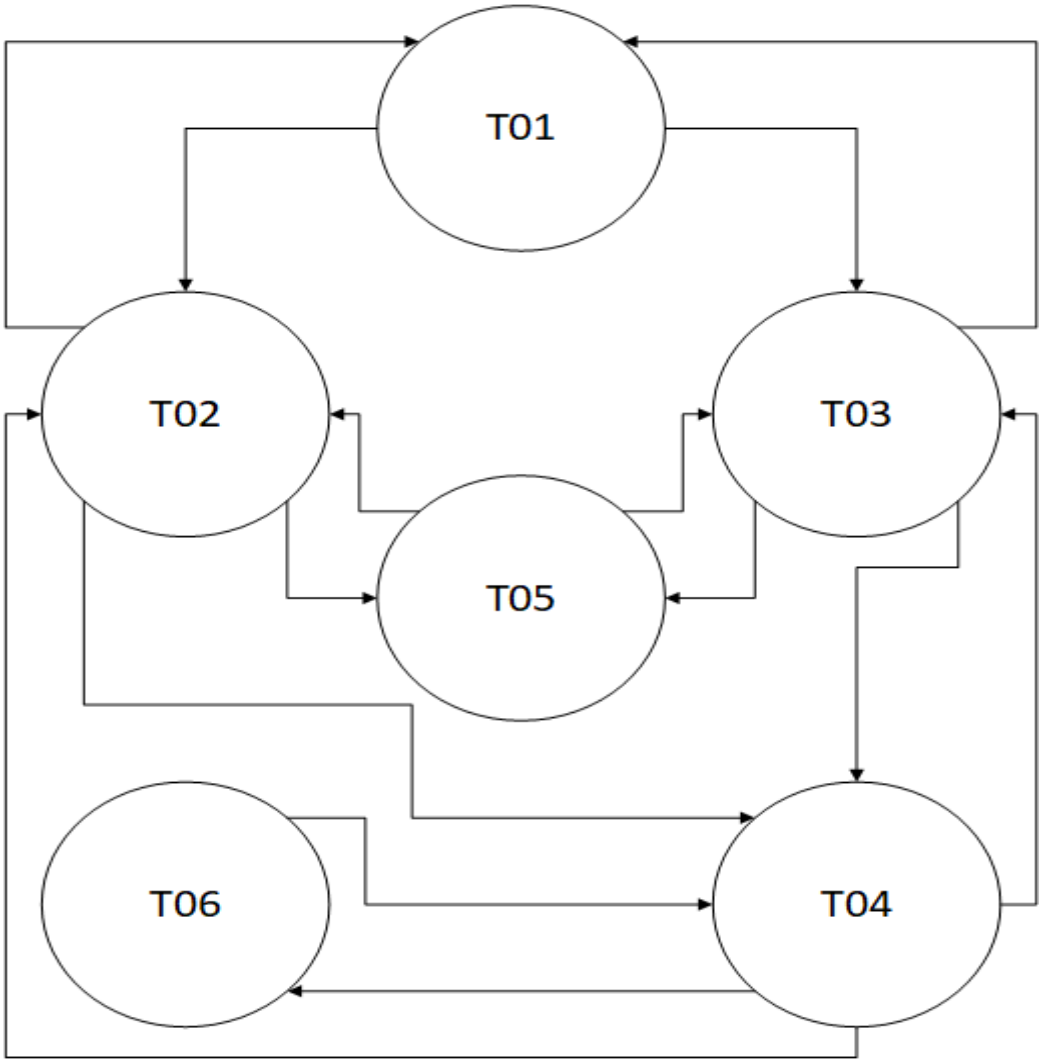
Pada perancangan T06 adalah perancangan halaman view piksel dinama pengguna dapat melihat nilai dari setiap piksel frame yang dipilih.

Tabel 3.69 Perancangan Antarmuka Lihat Piksel

No T06	-
	
Ukuran : 800x600 Font : Segoe UI	

3.10.3 Jaringan Semantik

Jaringan semantik menggambarkan keterhubungan dari navigasi menu dari satu antarmuka ke antarmuka lain. Jaringan semantik yang terbentuk pada sistem ini dapat dilihat pada Gambar 3.28 berikut.



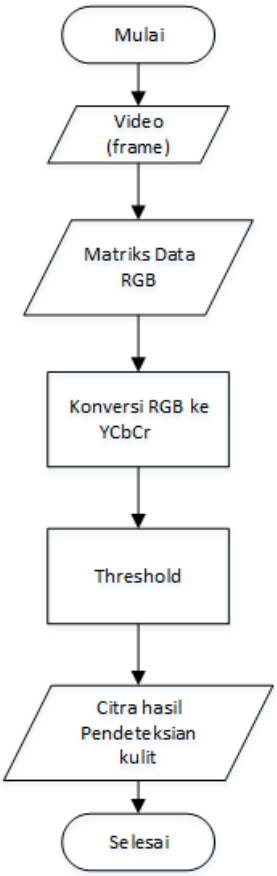
Gambar 3.28 Jaringan Semantik

3.10.4 Perancangan Metode

Perancangan metode merupakan perancangan untuk mengetahui alur proses yang terjadi didalam metode yang ada. Adapun perancangan metode yang akan dijelaskan.

3.10.4.1 Perancangan Metode Pendeteksian Warna Kulit

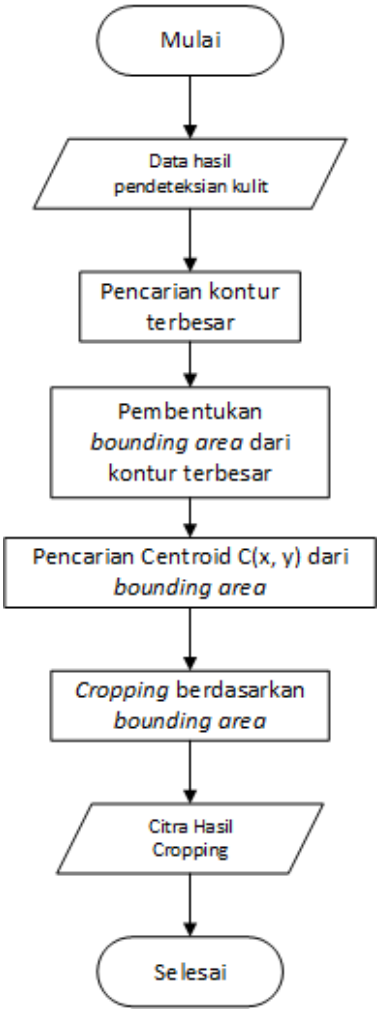
Perancangan metode pendeteksian warna kulit dapat dilihat pada



Gambar 3.29 Perancangan Metode Pendeteksian Warna Kulit

3.10.4.2 Perancangan Metode *Hand Tracking*

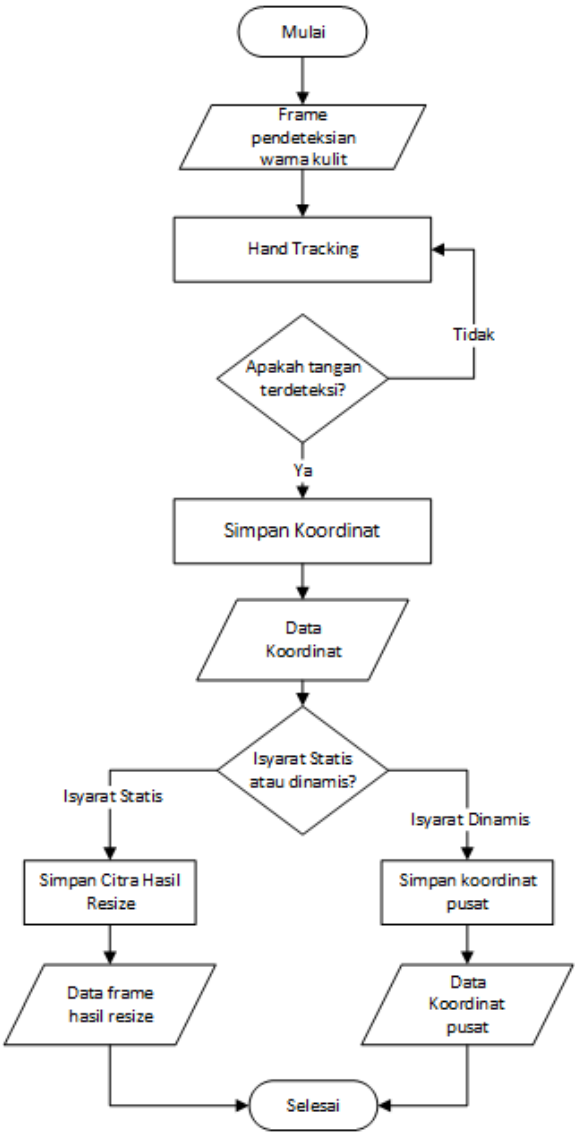
Perancangan metode hand tracking dapat lihat pada Gambar 3.30 berikut.



Gambar 3.30 Perancangan Metode *Hand Tracking*

3.10.4.3 Perancangan Metode Pengambilan *Frame*

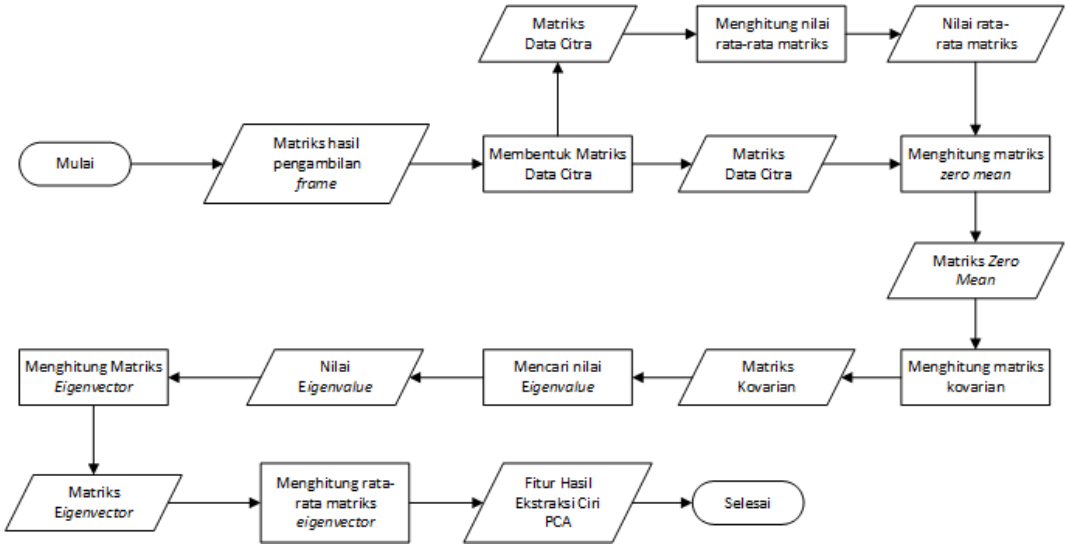
Perancangan metode pengambilan frame dapat dilihat pada Gambar 3.31 berikut.



Gambar 3.31 Perancangan Metode Pengambilan *Frame*

3.10.4.4 Perancangan Metode Ekstraksi Ciri PCA

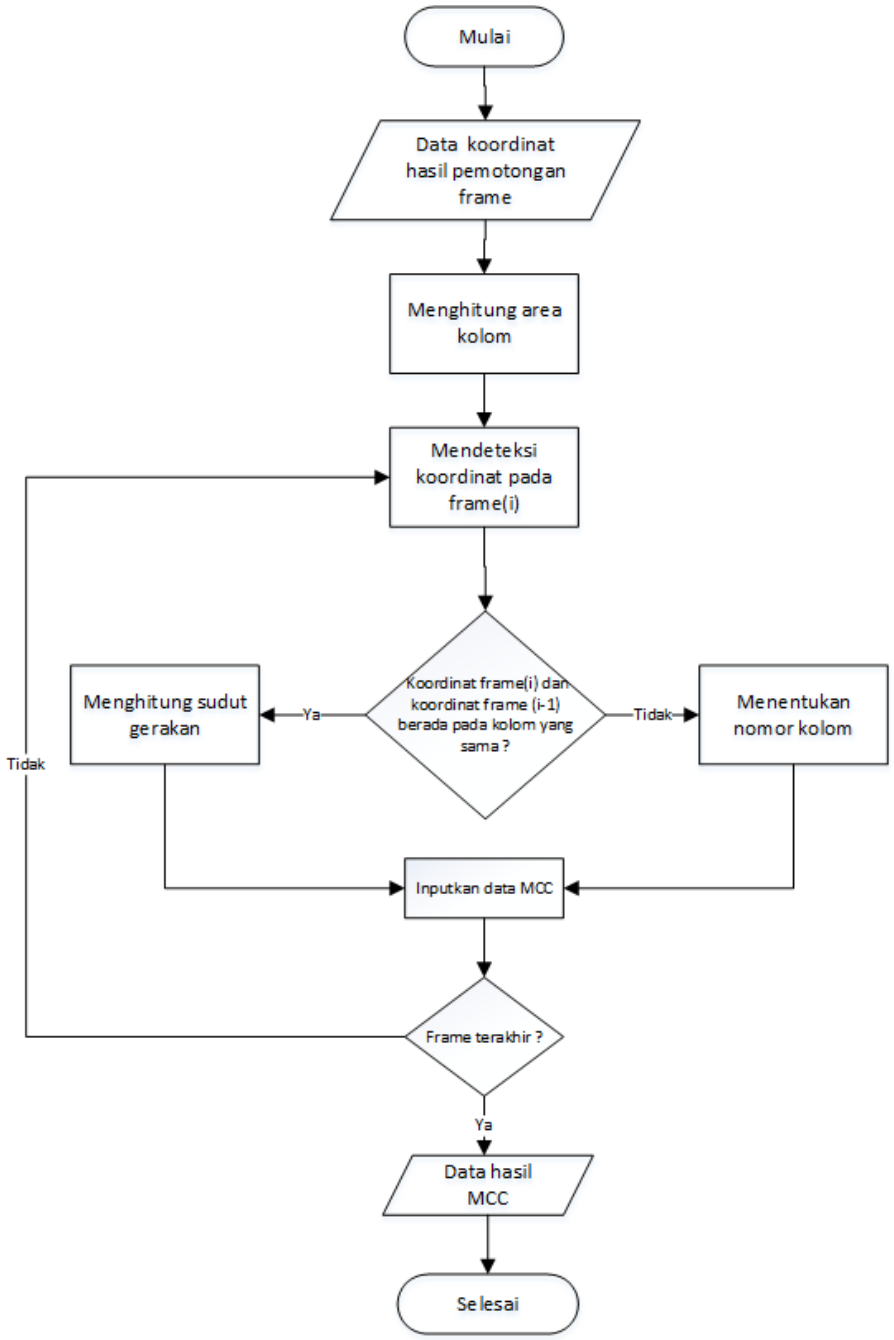
Perancangan metode ekstraksi ciri pca dapat dilihat pada Gambar 3.32 berikut.



Gambar 3.32 Perancangan Metode Ekstraksi Ciri PCA

3.10.4.5 Perancangan Metode *Motion Chain Code*

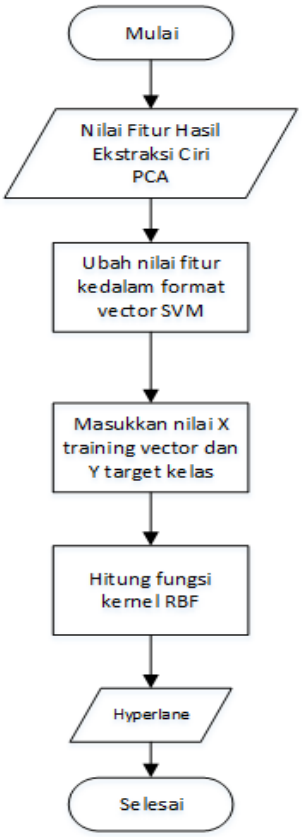
Perancangan metode motion chain code dapat dilihat pada Gambar 3.33 berikut.



Gambar 3.33 Perancangan Metode *Motion Chain Code*

3.10.4.6 Perancangan Metode Pelatihan *Multiclass* SVM

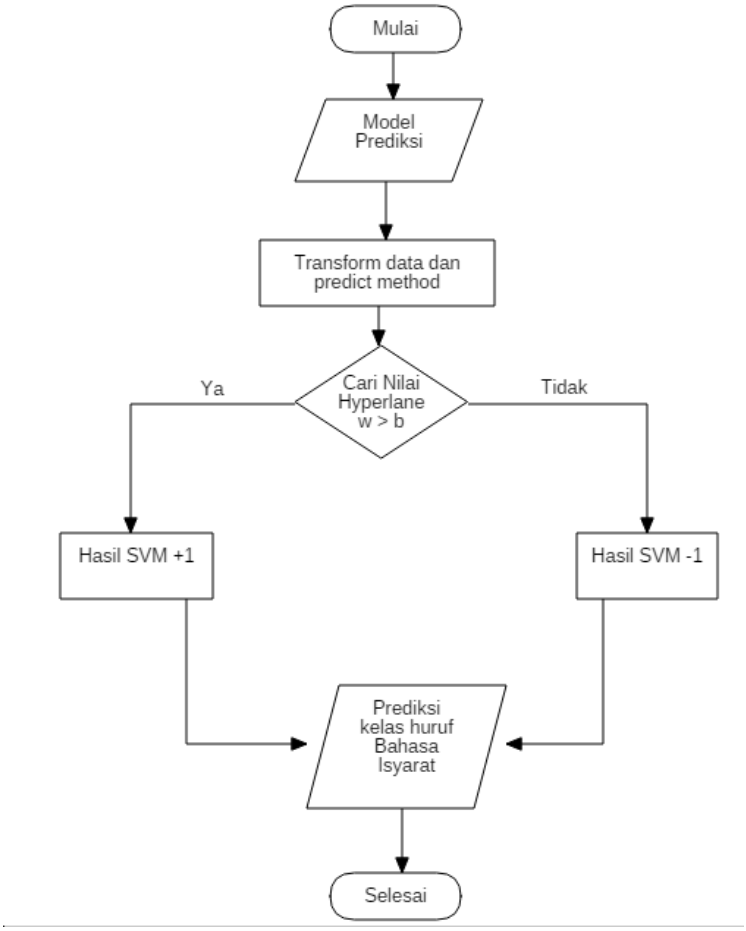
Perancangan metode pelatihan *multiclass* SVM dapat dilihat pada Gambar 3.34 berikut.



Gambar 3.34 Perancangan Metode Pelatihan

3.10.4.7 Perancangan Metode Pengujian *Multiclass* SVM

Perancangan metode Pengujian *multiclass* SVM dapat dilihat pada Gambar 3.35 berikut.



Gambar 3.35 Perancangan Metode Pengujian