# BAB III ANALISIS DAN PERANCANGAN SISTEM

Bab ini membahas tahap analisis permasalahan dan perancangan dari sistem yang dibangun. Analisis permasalahan membahas permasalahan yang diangkat dalam pengerjaan tugas akhir. Analisis kebutuhan mencantumkan kebutuhan-kebutuhan yang diperlukan oleh sistem. Selanjutnya dibahas mengenai perancangan sistem yang dibuat.



## 3.1 Analisis Perangkat Lunak

Bahasa isyarat merupakan alat komunikasi utama bagi penyandang tunarungu. Bahasa itu tidak mudah dipahami bagi orang pada umumnya, tetapi dapat sangat membantu penyandang tunarungu untuk berkomunikasi dengan sekitarnya. Namun masih banyak yang belum mengerti apa arti isyarat bagi yang diberi atau memberi isyarat.

Perangkat lunak ini bertujuan untuk membantu pengguna mengartikan bahasa isyarat yang dilakukan. Perangkat lunak ini dibangun dengan Microsoft Visual Studio dan dengan bantuan Kinect 2.0 untuk mengekstraksi *skeleton joint*. Data yang didapatkan dari hasil ekstraksi kemudian disimpan dan dilakukan kalkulasi untuk menjadi fitur-fitur sesuai dengan kebutuhan. Hasil dari fitur yang didapat disebut data *training*, diklasifikasi menggunakan *classifier* WEKA sehingga menghasilkan model *Decision Tree* yang akan digunakan untuk mengartikan bahasa isyarat yang diberikan

Untuk tahap *testing* data, setelah perangkat lunak mengekstraksi fitur yang didapatkan dari kalkulasi *skeleton* pengguna, fitur tersebut akan diklasifikasikan menggunakan Decision Tree yang sebelumnya sudah dibuat kemudian perangkat lunak mengartikan gerakan isyarat yang diberikan. Berikutnya perangkat lunak menampilkan arti dari gerakan isyarat yang dilakukan oleh pengguna.

### 3.1.1 Deskripsi Umum Perangkat Lunak

Tugas Akhir yang dibangun ini adalah sebuah modul pengenalan bahasa isyarat Indonesia dengan menggunakan teknologi Kinect 2.0. Gerakan isyarat yang dideteksi dapat berupa gerak isyarat statis dan gerakan isyarat dinamis.

Pengguna utama adalah semua orang yang ingin mempelajari gerakan isyarat yang mengacu pada SIBI. Pengguna dapat mempelajari isyarat yang sudah tersedia didalam perangkat lunak atau dapat memberikan isyarat baru yang mengacu pada SIBI.

### 3.1.2 Spesifikasi Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak yang akan dibuat ini melibatkan dua hal, yakni kebutuhan fungsional maupun kebutuhan non-fungsional. Dimana masing-masing berhubungan dengan keberhasilan dalam pembuatan tugas akhir ini.

#### 3.1.2.1 Kebutuhan Fungsional Perangkat Lunak

Pada perangkat lunak ini, terdapat beberapa kebutuhan fungsional yang dapat mendukung jalannya perangkat lunak ini. Berikut adalah beberapa kebutuhan fungsional :

1. Mendeteksi *skeleton* pengguna

Perangkat lunak dapat mendeteksi pengguna yang sedang berada di depan Kinect 2.0.

1. Mengekstraksi fitur statis dan dinamis *skeleton*

Perangkat lunak dapat mendeteksi posisi dari *skeleton* yang akan diekstraksi menjadi fitur-fitur yang dibutuhkan. Fitur-fitur ini yang akan dikasifikasikan melalui proses *training* dan *testing*.

1. Menerjemahkan Bahasa Isyarat

Perangkat lunak dapat menerjemahkan bahasa isyarat Indonesia yang dihasilkan melalui proses ekstraksi fitur statis dan dinamis *skeleton* pengguna.

#### 3.1.2.2 Kebutuhan Non-Fungsional

Tidak hanya kebutuhan fungsional saja, terdapat juga beberapa kebutuhan non-fungsional dalam mendukung dan menambah performa jalannya perangkat lunak. Kebutuhan non-fungsional tersebut adalah sebagai berikut:

1. Penyesuaian intensitas cahaya

Intensitas cahaya merupakan salah satu unsur yang perlu diperhatikan dalam penggunaan sensor Kinect 2.0. Jika intensitas cahaya pada saat pengambilan data kurang, maka dapat mengganggu stabilitas pengambilan data yang dapat berakibat pada hasil akhir yang ditampilkan oleh perangkat lunak. Maka dari itu, penggunaan perangkat lunak ini sebaiknya berada pada ruangan yang memiliki intensitas cahaya yang cukup.

1. Posisi Kinect dengan pengguna

Posisi peletakan Kinect 2.0 disesuaikan dengan pengguna untuk mendapatkan fitur-fitur yang maksimal, baik saat pengambilan data *traning* maupun testing. Jarak optimal Kinect 2.0 dengan pengguna adalah antara 0,6 sampai 1,8 meter.

### 3.1.3 Identifikasi Pengguna

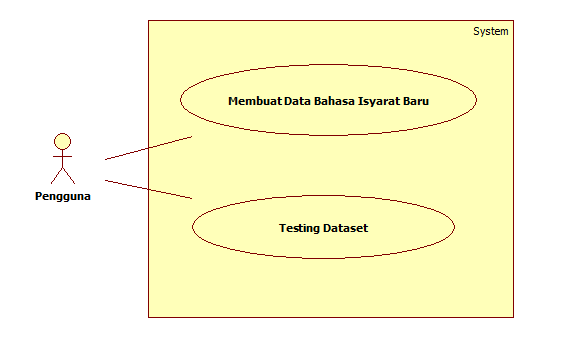
Dalam tugas akhir yang dibangun ini, pengguna yang akan terlibat dalam menjalankan perangkat lunak hanya satu orang saja, yaitu orang yang akan melakukan pengenalan bahasa isyarat Indonesia.

## 3.2 Perancangan Perangkat Lunak

Subbab ini membahas bagaimana rancangan dari tugas akhir ini. Hal yang dibahas meliputi model kasus penggunaan, definisi aktor, definisi kasus penggunaan, arsitektur umum sistem, rancangan antarmuka perangkat lunak, dan rancangan proses perangkat lunak.

### 3.2.1 Model Kasus Penggunaan

Dari hasil analisa deskripsi umum perangkat lunak dan spesifikasi kebutuhan perangkat lunak yang telah dijelaskan, maka model kasus penggunaan untuk perangkat lunak pengenalan bahasa isyarat dapat dilihat pada Gambar 3.1.



Gambar 3.1 Diagram Kasus Penggunaan Perangkat Lunak

### 3.2.2 Definisi Aktor

Aktor yang terdapat dalam sistem aplikasi ini terlihat pada Tabel 3.1.

Tabel 3.1 Definisi Kasus Penggunaan

|  |  |  |
| --- | --- | --- |
| **No** | **Nama** | **Deskripsi** |
| 1 | Pengguna | Merupakan aktor yang bertugas untuk menambahkan data *training* dan melakukan *testing* gerakan isyarat statis dan dinamis, seluruh fungsionalitas yang ada di dalam sistem dapat digunakan oleh pengguna. |

### 3.2.3 Definisi Kasus Penggunaan

Pada Gambar 3.1 telah dijelaskan bahwa aktor yang dalam hal ini disebut pengguna mempunyai dua kasus penggunaan, yakni membuat data bahasa isyarat baru dan melakukan *testing* *dataset*. Rincian mengenai kasus penggunaan tersebut dapat dilihat pada Tabel 3.2.

Tabel 3.2 Definisi Kasus Penggunaan

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Kode Kasus Penggunaan** | **Nama Kasus Penggunaan** | **Keterangan** |
| 1 | UC-01 | Membuat Data Bahasa Isyarat Baru | Pengguna membuat data bahasa isyarat yang baru |
| 2 | UC-02 | *Testing Dataset* | Pengguna melakukan *testing* *dataset* dengan melakukan gerakan bahasa isyarat Indonesia yang tersedia. |

#### 3.2.3.1 Kasus Penggunaan Membuat Data Bahasa Isyarat Baru

Spesifikasi kasus penggunaan membuat data bahasa isyarat baru dapat dilihat pada Tabel 3.3.

Tabel 3.3 Spesifikasi Kasus Penggunaan Membuat Data Bahasa Isyarat Baru

|  |  |
| --- | --- |
| **Nama Kasus Penggunaan** | **Membuat Data Bahasa Isyarat Baru** |
| Nomor | UC-01 |
| Deskripsi | Kasus penggunaan aktor untuk membuat data bahasa isyarat baru |
| Aktor | Pengguna |
| Kondisi Awal | Pengguna sudah menjalankan perangkat lunak dan perangkat Kinect 2.0 telah tersambung |
| Alur Normal | 1. Pengguna memasukan nama bahasa isyarat yang akan dibuat di dalam *textbox* perangkat lunak  2. Pengguna menekan tombol “Create File” di dalam perangkat lunak  3. Perangkat lunak menerima inputan dari Kinect 2.0 dan ketika *skeleton* pengguna ditemukan, perangkat lunak memberikan waktu 5 detik untuk pengguna mempersiapkan gerakan  A1. Kinect 2.0 tidak menemukan *skeleton* pengguna  4. Pengguna melakukan gerakan bahasa isyarat yang akan dibuat  5. Perangkat lunak mengekstrak data *skeleton* pengguna sebanyak 40 data/*frame* untuk dikalkulasi  6. Perangkat lunak menyimpan hasil ekstraksi ke dalam sebuah berkas berekstensi .csv |
| Alur Alternatif | A1. Kinect 2.0 tidak menemukan *skeleton* pengguna  1. Sistem memberikan notifikasi bahwa *skeleton* pengguna tidak ditemukan |
| Kondisi Akhir | Perangkat lunak membuat dataset bahasa isyarat yang baru |

#### 3.2.3.2 Kasus Penggunaan *Testing* Data

Spesifikasi kasus penggunaan *Testing* Data dapat dilihat pada Tabel 3.4.

Tabel 3.4 Spesifikasi Kasus Penggunaan *Training Dataset*

|  |  |
| --- | --- |
| **Nama Kasus Penggunaan** | ***Testing Dataset*** |
| Nomor | UC-02 |
| Deskripsi | Kasus penggunaan aktor untuk melakukan *testing dataset* isyarat bahasa Indonesia |
| Aktor | Pengguna |
| Kondisi Awal | Pengguna dalam keadaan menjalankan perangkat lunak dan sudah ada model hasil *classifier* data isyarat bahasa Indonesia di dalam perangkat lunak |
| Alur Normal | 1. Pengguna menekan tombol “Start Testing” di dalam perangkat lunak  2. Perangkat lunak menerima inputan dari Kinect 2.0 dan ketika *skeleton* pengguna ditemukan, perangkat lunak memberikan waktu 3 detik untuk pengguna mempersiapkan gerakan  A1. Kinect 2.0 tidak menemukan *skeleton* pengguna  3. Pengguna melakukan gerakan bahasa isyarat  4. Perangkat lunak mengekstrak data *skeleton* pengguna sebanyak 40 data/*frame* untuk dikalkulasi  5. Hasil ekstraksi data diklasifikasi menggunakan *Decision Tree* yang sudah dibuat di dalam perangkat lunak  6. Perangkat lunak menampilkan isyarat bahasa Indonesia hasil klasifikasi |
| Alur Alternatif | A1. Kinect 2.0 tidak menemukan *skeleton* pengguna  1. Sistem memberikan notifikasi bahwa *skeleton* pengguna tidak ditemukan |
| Kondisi Akhir | Perangkat lunak memberikan keluaran berupa bahasa isyarat yang dimaksud oleh pengguna |

### 3.2.4 Arsitektur Umum Sistem

Arsitektur umum pada perangkat lunak ini memiliki perangkat tambahan Kinect 2.0 sebagai perangkat masukan. Implementasi aplikasi dibuat menggunakan Microsoft Visual Studio. Arsitektur umum perangkat lunak yang akan dibuat dapat dilihat pada Gambar 3.2.

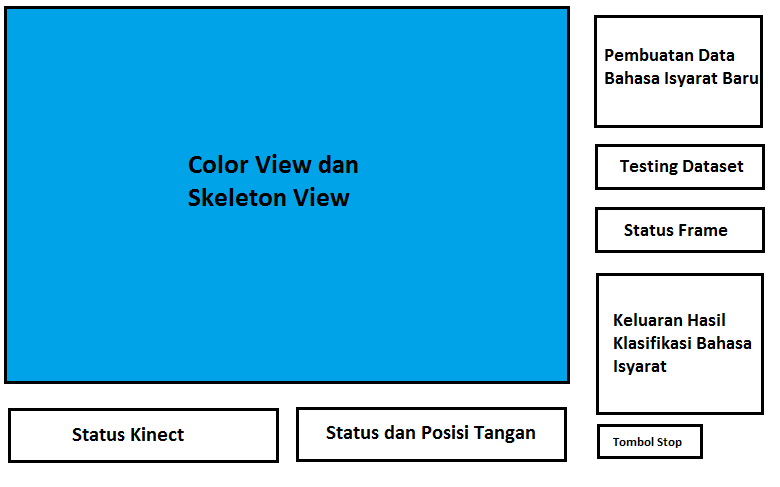


Gambar 3.2 Arsitektur Umum Sistem

### 3.2.5 Rancangan Antarmuka Perangkat Lunak

Rancangan antarmuka perangkat lunak diperlukan untuk memberikan gambaran umum kepada pengguna bagaimana sistem yang ada dalam perangkat lunak ini berinteraksi dengan pengguna. Selain itu, rancangan ini juga memberikan gambaran bagi pengguna tentang tampilan yang sudah disediakan didalam perangkat lunak, sehingga akan muncul kesan pengalaman pengguna yang baik dan mudah.

Rancangan antarmuka perangkat lunak ini hanya memiliki satu Windows dengan beberapa bagian seperti *color view*, *skeleton view* dan memiliki beberapa kontrol yang sekiranya dapat dipahami oleh pengguna. Rancangan antarmuka perangkat lunak dapat dilihat pada Gambar 3.3.



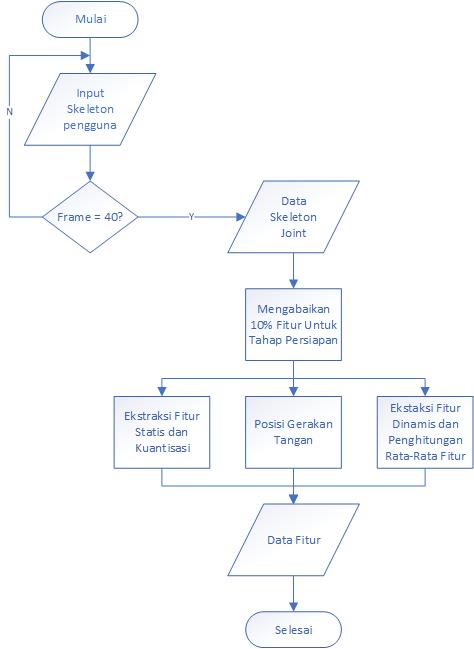
Gambar 3.3 Rancangan Antarmuka Perangkat Lunak

### 3.2.6 Rancangan Proses Perangkat Lunak

Pada rancangan proses perangkat lunak akan dijelaskan mengenai proses yang terjadi dalam sistem untuk memenuhi fungsionalitas yang ada pada perangkat lunak. Proses ini penting agar perangkat lunak dapat berjalan secara baik dan benar.

#### 3.2.6.1 Rancangan Proses Ekstraksi Fitur

Proses ekstraksi fitur ini sangat dibutuhkan bagi pengguna untuk melakukan *training* dan *testing*. Rancangan proses pengambilan fitur dapat dilihat pada Gambar 3.4.



Gambar 3.4 Diagram Alir Proses Ekstraksi Fitur

1. *Input* *skeleton joint* pengguna oleh Kinect 2.0

Saat perekaman *skeleton joint* oleh Kinect 2.0 yang kemudian diolah menjadi fitur data, perangkat lunak akan memberikan waktu terlebih dahulu kepada pengguna selama 5 detik untuk mempersiapkan gerakan. Setelah itu, perangkat lunak mengambil total 40 data untuk dikalkulasi dan dijadikan fitur.

Jumlah data tersebut ditentukan berdasarkan durasi gerakan bahasa isyarat yang paling lama dilakukan diantara seluruh *sample* gerakan yang sudah ditentukan sebelumnya. Apabila gerakan pengguna terlalu cepat atau terlalu lambat, maka pengambilan data harus diulang sekali kembali guna mendapatkan hasil yang maksimal.

Untuk melakukan ekstraksi *skeleton* pengguna dalam melakukan *training* maupung *testing*, penulis mengambil sebanyak 11 *skeleton joints* yang diketahui oleh Kinect 2.0. Sebelas *skeleton joints* tersebut adalah sebagai berikut:

1. Leher (Vektor N)
2. Bahu tulang belakang (Vektor SS)
3. Tulang belakang tengah (Vektor SM)
4. Bahu tangan kanan (Vektor SR)
5. Siku tangan kanan (Vektor ER)
6. Pergelangan tangan kanan (Vektor WR)
7. Telapak Tangan kanan (Vektor HR)
8. Bahu tangan kiri (Vektor SL)
9. Siku tangan kiri (Vektor EL)
10. Pergelangan tangan kiri (Vektor WL)
11. Telapak tangan kiri (Vektor HL)

2. Ekstraksi fitur statis

Dalam proses ekstraksi fitur statis, didapatkan fitur yang berjumlah 21 buah[ ]. Dua puluh satu fitur yang dimaksud dapat dilihat pada Tabel 3.5

Tabel 3.5 Fitur Statis

|  |  |  |
| --- | --- | --- |
| Vektor2 (x,y) | Sudut (float) | Jarak (float) |
| SR -> ER | ∠ SS – SR – ER | HR – HL |
| ER -> WR | ∠ SR – ER – WR |
| WR -> HR | ∠ ER – WR – HR |
| SL -> EL | ∠ SS – SL – EL |
| EL -> WL | ∠ SL – EL – WL |
| WL -> HL | ∠ EL – WL – HL |
| HR -> HL |  |

3. Ekstraksi fitur dinamis dan kuantisasi

Dalam proses ekstraksi fitur dinamis, berdasarkan persamaan (1.0) dan (1.1), perangkat lunak terlebih dahulu menentukan dan  yang didapatkan dari hasil selisih masing-masing koordinat X dan Y antar data ke-*n* dan *n+1 skeleton joints* HR dan HL. Nilai dan kemudian digunakan dalam menentukan orientasi sudut mutlak () berdasarkan persamaan (1.2) yang dapat dijelaskan sebagai berikut:

1. Jika dan lebih besar atau sama dengan daripada nol, maka nilai adalah arctan hasil pembagian dan kemudian dikalikan dengan hasil pembagian 180 dengan PI ().
2. Jika lebih kecil daripada nol, maka nilai – adalah arctan hasil pembagian dan dikalikan dengan hasil pembagian 180 dengan kemudian ditambahkan 180.
3. Jika lebih kecil daripada nol, maka nilai adalah arctan hasil pembagian dan dikalikan dengan hasil pembagian 180 dengan kemudian ditambah 360.

Hasil dari tersebut kemudian diterjemahkan menjadi data kuantisasi hasil ekstraksi fitur dinamis (berdasarkan Gambar 2.3) yang dapat dijelaskan sebagai berikut:

1. Jika sama dengan nol, maka kuantisasinya adalah 0.
2. Jika lebih besar daripada 314, maka kuantisasinya adalah 8.
3. Jika lebih besar daripada 269, maka kuantisasinya adalah 7.
4. Jika lebih besar daripada 224, maka kuantisasinya adalah 6.
5. Jika lebih besar daripada 179, maka kuantisasinya adalah 5.
6. Jika lebih besar daripada 134, maka kuantisasinya adalah 4.
7. Jika lebih besar daripada 89, maka kuantisasinya adalah 3.
8. Jika lebih besar daripada 44, maka kuantisasinya adalah 2.
9. Jika lebih kecil daripada 44, maka kuantisasinya adalah 1.

Data kuantisasi hasil ekstraksi fitur dinamis gerakan tangan yang didapatkan kemudian dijadikan fitur data. Terdapat 80 fitur dinamis yang diekstraksi. Fitur ke 1 s.d. 40 menginterprestasikan data kuantisasi 40 data yang dihasilkan oleh tangan kiri. Sedangkan fitur ke 41 s.d. 80 menginterprestasikan data kuantisasi 40 data yang dihasilkan oleh tangan kanan.

Setelah dikurangi dari 10% data yang diabaikan dalam tahap persiapan, perangkat lunak mengambil hanya data genap saja untuk mendapatkan data yang lebih akurat [ ]. Maka didapatlah 18 fitur dinamis untuk tangan kanan dan 18 fitur dinamis untuk tangan kiri.

4. Posisi gerakan tangan

Pada datake-20, perangkat lunak mengidentifikasi posisi gerakan tangan yang dilakukan oleh pengguna. Seperti yang dapat dilihat pada Gambar 3.5, posisi gerakan tangan dibagi menjadi tiga daerah, yaitu Area 1 (kepala), Area 2 (dada), dan Area 3 (perut). Daerah posisi gerakan tangan tersebut ditentukan oleh *skeleton joints* N dan SM yang dapat dijelaskan sebagai berikut:

1. Jika koordinat *skeleton joints* HR dan atau HLberada diatas (lebih besar daripada) koordinat *skeleton joints* N, maka posisi diidentifikasikan berada pada Area 1.
2. Jika koordinat *skeleton joints* HR dan atau HL berada dibawah (lebih kecil daripada) koordinat *skeleton joints* N dan diatas (lebih besar daripada) koordinat *skeleton joints* SM, maka posisi diidentifikasi berada pada Area 2.
3. Jika koordinat *skeleton joints* HR dan atau HLberada dibawah (lebih kecil daripada) koordinat *skeleton joints* SM, maka posisi diidentifikasikan berada pada Area 3.

Posisi gerakan tangan yang sudah teridentifikasi kemudian dijadikan fitur data. Terdapat dua fitur data yang dihasilkan oleh penentuan posisi gerakan tangan. Masing-masing fitur mewakili tangan pengguna yaitu tangan kanan dan tangan kiri.

5. Fitur data akhir

Setelah seluruh fitur yang diperlukan sudah didapat. Fitur-fitur tersebut disatukan dan disimpan. Susunan dari fitur-fitur tersebut dapat dilihat pada Tabel 3.6.

*Skeleton* Neck

*Skeleton*

SpinMid



Gambar 3.5 Posisi Gerakan Tangan pada Tubuh Mengacu pada Leher (Vektor N) dan Tulang Belakang Tengah (Vektor SM)

Tabel 3.6 Fitur Dinamis

|  |  |
| --- | --- |
| **Fitur** | **Indeks Fitur** |
| Kuantisasi Tangan Kiri | Fitur ke 1 s.d. 18 |
| Kuantisasi Tangan Kanan | Fitur ke 19 s.d. 36 |
| Posisi Tangan Kiri | Fitur ke 37 |
| Posisi Tangan Kanan | Fitur ke 38 |
| Statis | Fitur ke 39 s.d. 59 |

#### 3.2.6.2 Rancangan Proses *Training Dataset*



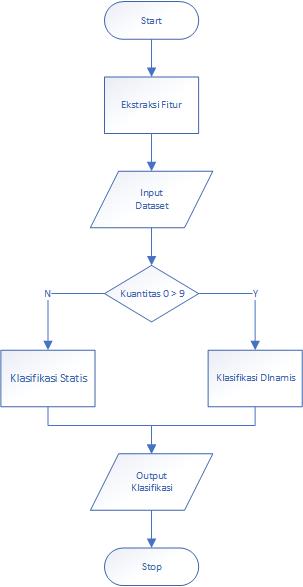
Gambar 3.6 Rancangan Proses *Training* *Dataset*

Seperti yang dapat dilihat pada Gambar 3.6, proses *training dataset* dimulai dengan menyimpan fitur data yang didapatkan sebelumnya terlebih dahulu ke dalam sebuah berkas berekstensi .csv. Data *training* tersebut kemudian diolah pada perangkat lunak WEKA menggunakan algoritma Random Tree. Keluaran yang didapatkan dari hasil *training* data tersebut berupa model Decision Tree dan digunakan sebagai *classifier* ketika pengguna melakukan *testing* data. *Classifier* berupa Decision Tree dapat dilihat pada Gambar 3.8 dan Gambar 3.9.

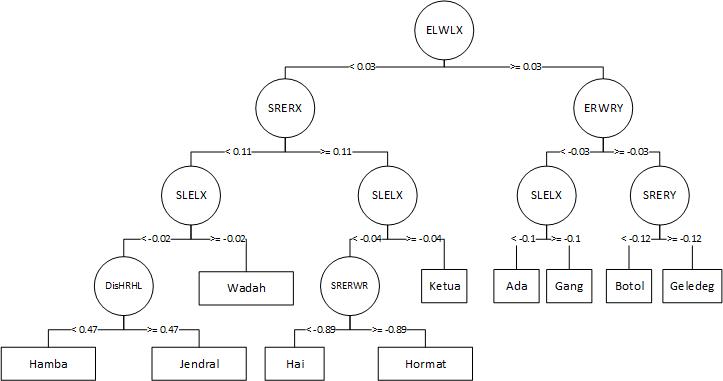
Proses ini dilakukan baik pada fitur statis dan fitur dinamis. Dikarenakan fitur-fitur yang dibutuhkan untuk mendapatkan hasil klasifikasi data yang maksimal, maka dipisahkan fitur-fitur sesuai dengan kebutuhan klasifikasi statis ataupun dinamis.

#### 3.2.6.3 Rancangan Proses *Testing* Data

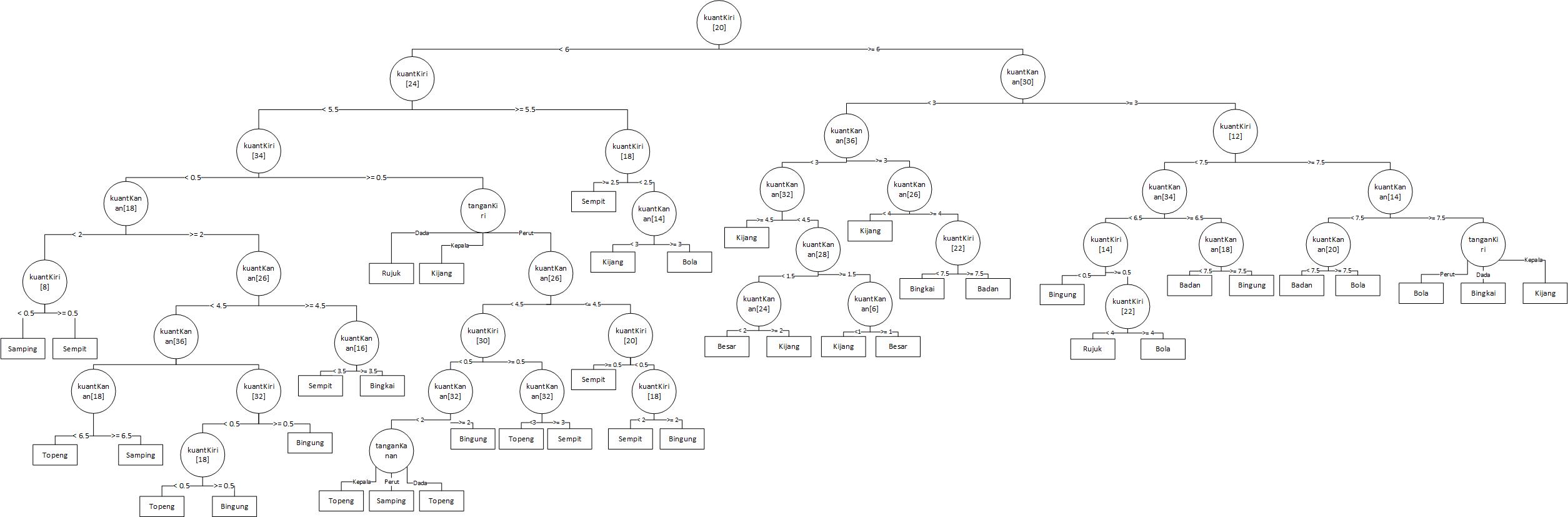
Seperti yang dapat dilihat pada Gambar 3.7, proses *testing* data dilakukan dengan mengenali gerakan isyarat yang dilakukan merupakan gerakan isyarat statis atau dinamis. Setelah hasil data fitur baru ditentukan, data tersebut dimasukan kedalam *classifier* yang didapat yang sudah didapatkan pada proses *training* data di Subbab 3.2.6.2. Keluaran hasil klasifikasi yang dilakukan tersebut berupa prediksi gerakan bahasa isyarat yang dimaksud oleh pengguna dalam bentuk gambar dan juga tulisan bahasa isyarat.



Gambar 3.7 Rancangan Proses *Testing* Dataset



Gambar 3.8 Decision Tree Statis



Gambar 3.9 Decision Tree Dinamis

*[Halaman ini sengaja dikosongkan]*