BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1. Metodologi Penelitian

3.1.1. Telaah Literatur

Penulis memulai tahap penelitian dengan melakukan telaah literatur. Pada tahap ini dipelajari beberapa hal yang akan dibutuhkan dalam pengembangan program. Hal penting yang mendapat perhatian lebih pada tahap ini adalah pemahaman mengenai *Planar Homography* dan penerapannya dalam pengembangan aplikasi.

3.1.2. Observasi

Tahap berikutnya yang dilakukan dalam penelitian adalah tahap observasi. Pada tahap ini penulis melakukan observasi pada dua aplikasi yang sudah ada, yaitu Pentabulous! dan Smoothboard untuk menentukan fitur-fitur yang dibutuhkan dalam aplikasi yang akan dibangun. Hasil dari observasi ini dijadikan tolak ukur salah satu pengujian yang akan dilakukan dalam penelitian ini, yaitu perbandingan fitur antar aplikasi. Observasi yang dilakukan juga mencakup kebutuhan akan *library* tertentu dalam pengembangan program.

3.1.3. Pengembangan Program

Tahap berikutnya yang dilakukan oleh penulis adalah tahap pengembangan program. Penulis menggunakan metode *prototyping* dalam melakukan pengembangan program. Dengan metode *prototyping*, penulis merancang terlebih dahulu program yang akan dikembangkan. Kemudian, tahap *coding* dan *testing* dilakukan secara berputar terus menerus dengan tujuan ketika ditemukan masalah dalam program dapat segera diperbaiki sebelum berlanjut ke bagian program yang lain. Proses tersebut berulang sampai program sudah sesuai dengan rancangan yang telah dibuat sebelumnya.

3.1.4. Pengujian

Setelah tahap pengembangan aplikasi, penelitian berlanjut ke tahap pengujian. Dalam penelitian ini dilakukan tiga macam pengujian dengan membandingkan aplikasi yang dibuat dengan aplikasi yang sudah ada sebelumnya, yaitu Pentabulous! dan Smoothboard. Pengujian yang dilakukan adalah perbandingan fitur antar aplikasi, perbandingan keakuratan dalam kalibrasi, dan waktu yang diperlukan oleh aplikasi untuk melakukan *pairing* dan *disconnect* Wiimote.

3.2. Analisis Pengembangan Program

Berdasar teori yang telah dipaparkan sebelumnya, Wiimote memiliki beberapa spesifikasi yang mendukung sebagai perangkat untuk melakukan

interaksi antar perangkat elektronik. Komponen utama yang mendukung adalah ketersediaan koneksi *Bluetooh* sehingga Wiimote dapat berkomunikasi dengan perangkat elektronik lain yang juga memiliki koneksi *Bluetooth* untuk melakukan pertukaran data. Komponen lain dari Wiimote yang berperan dalam penggunaannya sebagai media interaksi adalah kamera inframerah. Dengan menggunakan kamera ini, Wiimote dapat menangkap cahaya inframerah yang ada di hadapannya dan memberikan koordinat dari cahaya tersebut berdasarkan perspektif kamera.

Dengan komponen tersebut, dibutuhkan sebuah piranti utilisasi yang memungkinkan tampilan layar dari komputer menjadi sebuah media yang interaktif dengan menggunakan Wiimote sebagai sensor.Wiimote terlebih dahulu di-*pairing* dengan komputer untuk melakukan pertukaran data. Wiimote tersebut lalu dihadapkan pada tampilan layar komputer, baik itu pada monitor ataupun hasil proyeksi pada layar. Kemudian, Wiimote akan menangkap cahaya inframerah dari IR Pen yang digunakan oleh user untuk berinteraksi dengan layar. Koordinat dari inframerah yang ditangkap oleh kamera Wiimote akan dikirimkan pada komputer yang terkoneksi untuk diolah lebih lanjut. Setelah mendapat koordinat tersebut, piranti utilisasi pada komputer akan memetakannya pada koordinat layar dengan menggunakan *Planar Homography* dan melakukan suatu aksi pada titik tersebut. Serangkaian aksi ini akan membuat user seolah-olah berinteraksi langsung dengan tampilan layar dari komputer.

Setelah melakukan kajian teori, ada tiga bagian utama yang menjadi inti dari proses aplikasi yang akan dibangun, yaitu bagian yang bertugas

menghubungkan Wiimote dengan aplikasi, bagian yang bertugas menghitung transformasi dengan menggunakan metode *Direct Linear Transformation*, dan bagian yang bertugas membaca perubahan *state* dan memerintahkan aksi pada komputer. Aplikasi akan dibangun berbasis .NET karena sudah tersedia WiimoteLib *library* yang dapat membantu melakukan koneksi dengan Wiimote dan melakukan pembacaan *state* dari Wiimote yang terhubung.

Bagian dari aplikasi yang bertugas menghubungkan Wiimote dengan aplikasi akan menggunakan 32feet.net *library* yang merupakan *library* untuk menghubungkan perangkat *Bluetooth* secara umum. *Library* ini dapat digunakan untuk menghubungkan perangkat elektronik yang menggunakan *Bluetooth* sebagai media komunikasi sehingga juga dapat digunakan untuk melakukan komunikasi dengan Wiimote. Aplikasi yang akan dibangun akan melakukan pencarian terhadap Wiimote lalu melakukan *pairing* dengan Wiimote yang ditemukan pertama kali.

Metode *Direct Linear Transformation* digunakan dalam proses transformasi karena memiliki keadaan syarat mudah dipenuhi dan mudah untuk diterjemahkan ke dalam bahasa pemrograman. Proses *Singular Value Decomposition* yang dibutuhkan dalam proses *Direct Linear Transformation* akan dilakukan dengan menggunakan bantuan ALGLIB *Library* yang merupakan *library* yang dapat membantu pengerjaan persamaan-persamaan matematika dalam program.

Bagian dari aplikasi yang bertugas memberikan aksi pada komputer akan menggunakan *library* milik .NET sendiri yang dapat memanipulasi kursor dari

komputer. Dengan memanipulasi kursor, pengguna dapat melakukan interaksi dengan LCIW seperti dengan menggunakan *mouse*, hanya saja pengguna langsung berinteraksi dengan tampilan dari komputer.

3.3. Alat yang digunakan

Berikut merupakan rincian *Hardware* dan *Software* yang digunakan dalam pengembangan aplikasi:

3.3.1. *Hardware*

1. Komputer dengan spesifikasi sebagai berikut:
   a. Processor : Intel® Core™ 2 Duo T5450
   b. Memori : DDR2 3GB
   c. Harddisk : 320GB
2. Perangkat *Low-Cost Interactive Whiteboard* yang terdiri dari:
   a. Wiimote
   b. Tripod sebagai penyangga
   c. IR Pen
   d. *Bluetooth dongle*

3.3.2. *Software*

1. Sistem Operasi : Windows 7 Ultimate
2. *System Type* : 32-bit Operating System
3. *Applicationand Library* :
   a. Microsoft Visual Studio 2008
   b. Microsoft Windows Driver Kit

c. WiimoteLib *Library* 1.7

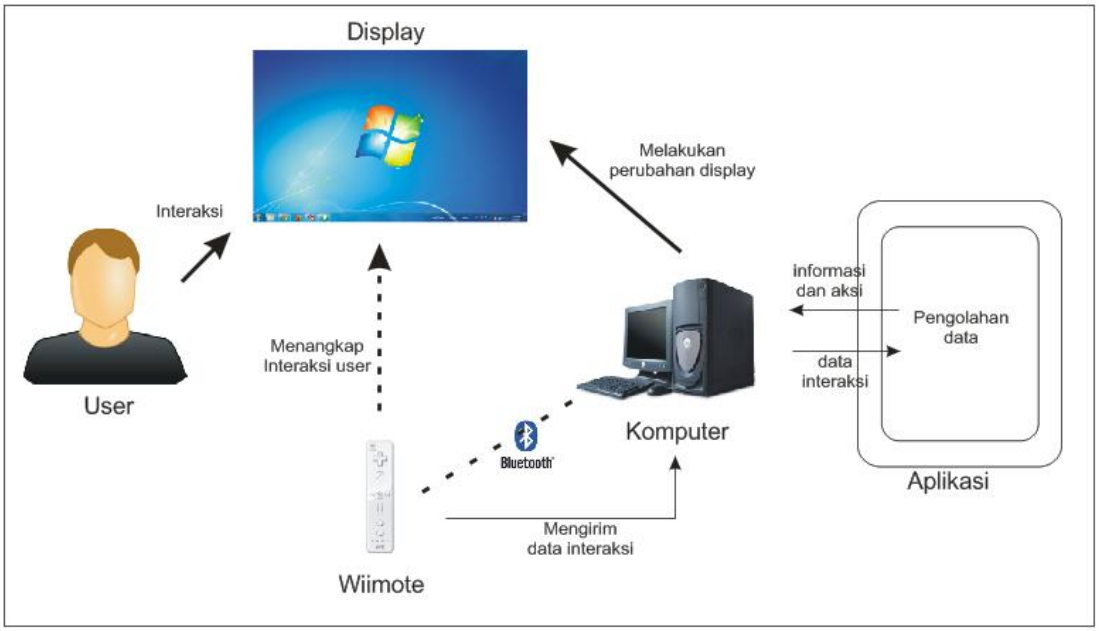
d. 32feet.NET *Library*

e. ALGLIB *Library* 3.5.0 untuk C#

3.4. Gambaran Umum Aplikasi

3.4.1 Diagram Umum *Low-Cost Interactive Whiteboard*

Gambar 3.1 menunjukkan bagaimana alur interaksi dalam *Low-Cost Interactive Whiteboard*.



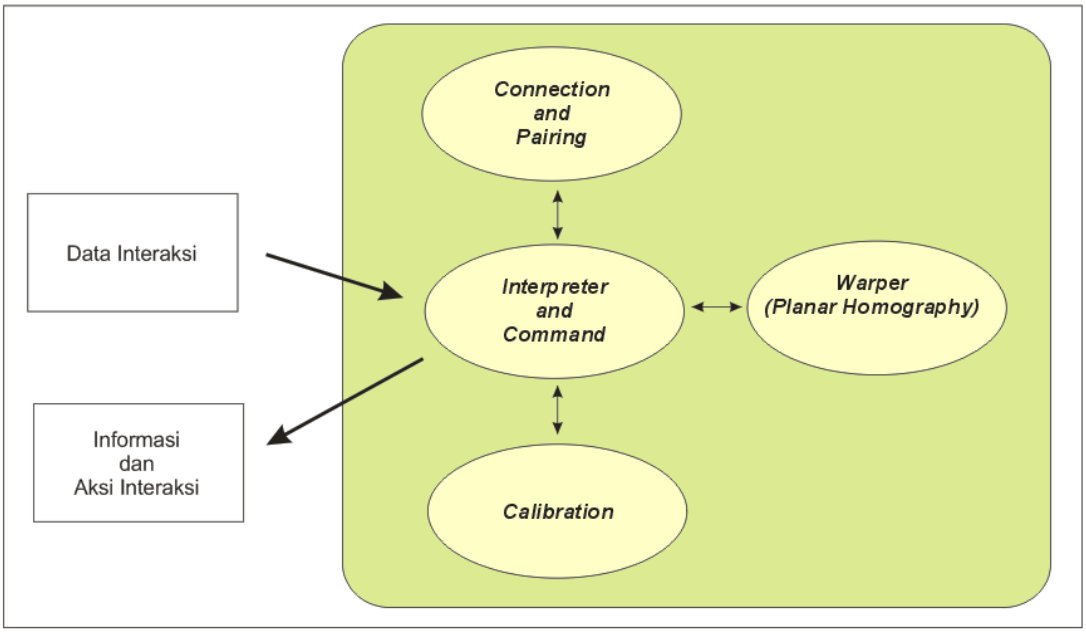
Gambar 3.1 Diagram Umum *Low-Cost Interactive Whiteboard*

Interaksi dimulai ketika user melakukan klik (menyalakan IR Pen) pada suatu titik di display layar komputer. Koordinat dari inframerah yang pada display layar ditangkap oleh Wiimote yang berfungsi sebagai sensor. Koordinat tersebut kemudian dikirimkan ke komputer melalui koneksi *Bluetooth* yang terjalin antara Wiimote dengan komputer. Aplikasi utilisasi

pada komputer yang menerima koordinat tersebut lalu memetakan koordinat perspektif kamera tersebut ke koordinat pixel sesungguhnya pada display komputer. Selain melakukan pemetaan koordinat, aplikasi juga menerjemahkan tindakan yang dilakukan oleh user untuk kemudian melakukan aksi yang sesuai pada koordinat pixel display komputer. Display komputer akan menerima perintah dari aplikasi dan menampilkan hal yang sesuai dengan perintah yang diterima. Setelah satu satu siklus ini, user akan merasakan seolah-olah berinteraksi dengan display layar seperti layar sentuh.

3.4.2 Diagram Umum Aplikasi

Gambar 3.2 menunjukkan diagram cara kerja aplikasi secara keseluruhan.



Gambar 3.2 Diagram Umum Aplikasi

Secara umum, terdapat empat proses utama yang dilakukan oleh aplikasi utilisasi LCIW, yaitu proses koneksi dan *pairing* antara perangkat Wiimote dengan komputer, proses kalibrasi untuk penghitungan *Planar Homography* yang akan digunakan, proses pengolahan dan pemetaan koordinat menggunakan *Planar Homography*, dan proses penerjemahan aksi dari user dan melakukan aksi tertentu pada koordinat hasil pemetaan.

Proses koneksi dan *pairing* antara Wiimote dengan komputer dilakukan dengan menggunakan bantuan 32feet.net *library*. Aplikasi akan mencari Wiimote yang berada dalam jangkauan *Bluetooth* yang dimiliki komputer. Aplikasi akan melakukan *pairing* pada Wiimote pertama yang ditemukan. Setalah proses *pairing*, aplikasi akan melakukan koneksi dan pertukaran data dengan Wiimote dengan bantuan WiimoteLib *library*. Dengan koneksi ini, aplikasi dapat mendeteksi perubahan-perubahan *state* yang terjadi pada Wiimote.

Proses kalibrasi dilakukan untuk mendapatkan empat koordinat yang saling memetakan antara *pixel* display layar dengan koordinat perspektif kamera Wiimote. Dengan menggunakan empat pasang koordinat tersebut maka matriks transformasi antar dua bidang dapat dihitung.

Pada proses pengolahan *Planar Homography*, aplikasi akan menghitung matriks transformasi antar dua bidang perspektif kamera dan display layar. Matriks transformasi itu akan digunakan untuk melakukan pemetaan koordinat-koordinat lainnya yang didapat dari kamera seiring dengan berlangsungnya interaksi antara user dengan layar.

Aplikasi juga melakukan proses penerjemahan terhadap perilaku interaksi yang dilakukan oleh user kemudian diteruskan sebagai perintah aksi untuk komputer pada koordinat pixel tertentu. Pada proses ini aplikasi menerjemahkan apakah jenis interaksi user berupa klik kiri, klik kanan, ataupun *drag click* seperti saat menggunakan perangkat tetikus.Aksi ini kemudian diteruskan kepada komputer sebagai perintah untuk melakukan aksi yang sesuai pada koodinat pixel yang diinginkan oleh user.

3.4.3 Fungsionalitas Aplikasi

Fungsionalitas yang dimiliki oleh aplikasi antara lain :

1. Mencari dan melakukan *pairing* terhadap Wiimote pertama dalam jangkauan *Bluetooth* komputer.
2. Melakukan kalibrasi antara koordinat perspektif kamera dengan koordinat *pixel* pada tampilan komputer.
3. Menghitung *Homography Matrix* berdasarkan koordinat-koordinat yang diketahui.
4. Memetakan koordinat perspektif kamera menjadi koordinat *pixel* sebenarnya.
5. Membaca perubahan *state* yang terjadi pada Wiimote.
6. Menerjemahkan interaksi dari pengguna dan memberikan perintah interaksi pada komputer.

3.5. Perancangan Aplikasi

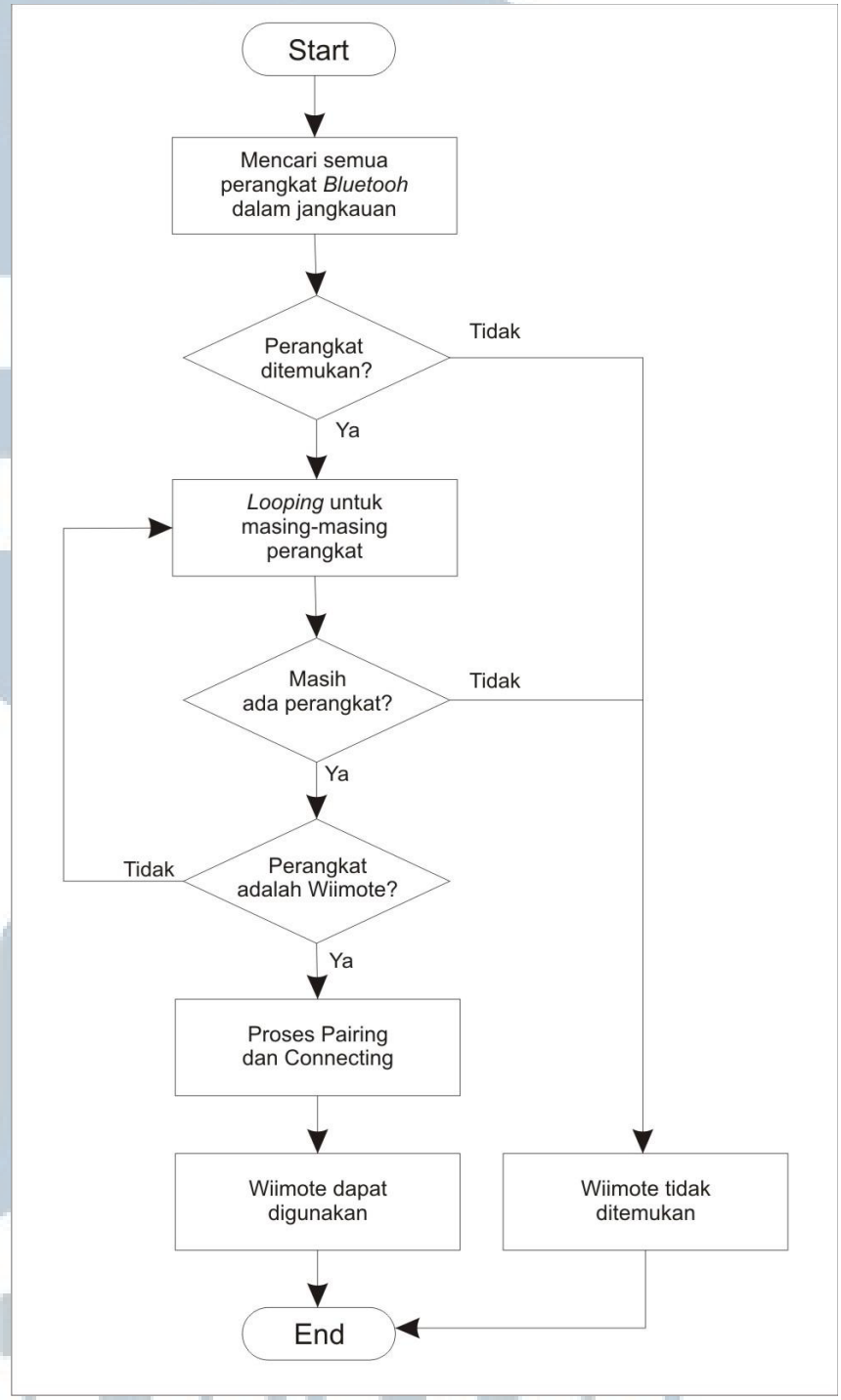
3.1.1. Perancangan Modul Aplikasi

Seperti yang sudah disebutkan sebelumnya, aplikasi utilisasi *LCIW* yang dibangun memiliki empat modul utama, yaitu :

1. Modul *Connection and Pairing*
2. Modul *Calibration*
3. Modul *Warper* (*Planar Homography*)
4. Modul *Interpreter and Command*.

Berikut akan dijelaskan secara mendetail mengenai fungsi, alur kerja, dan spesifikasi dari masing-masing modul.

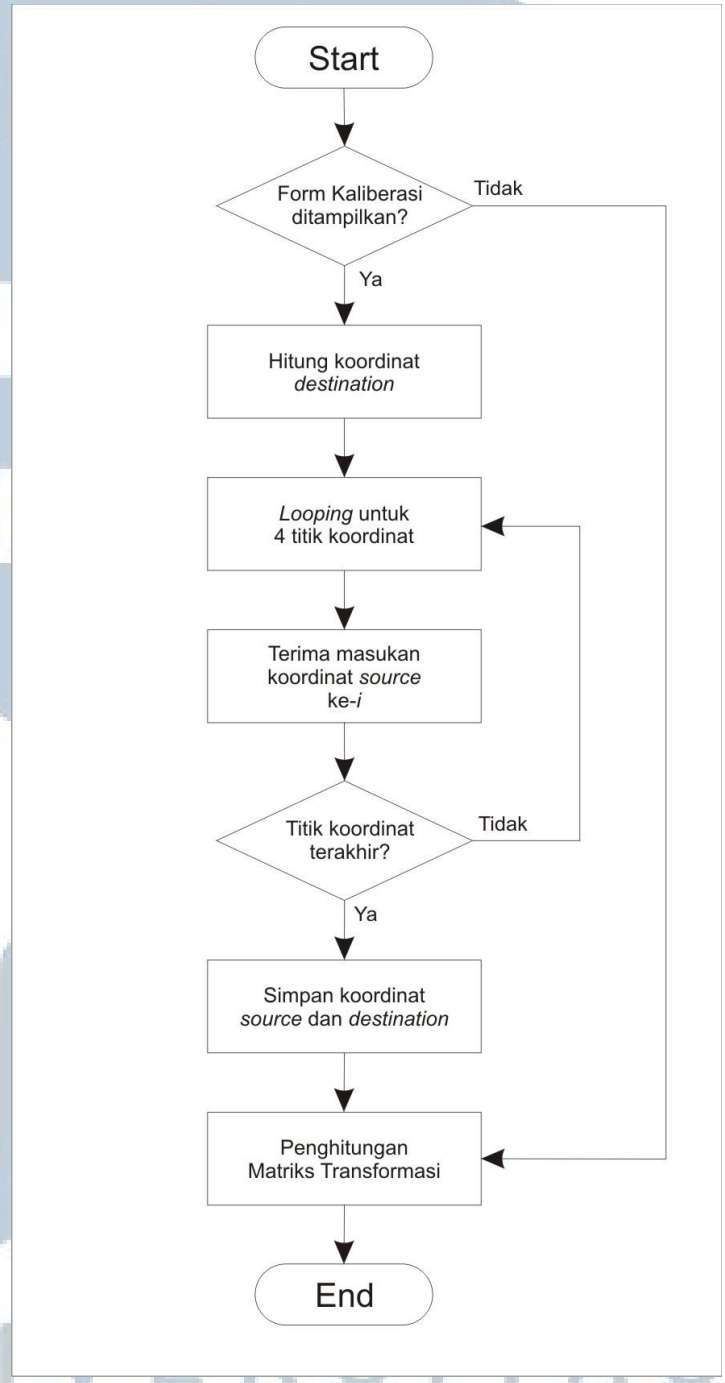
3.5.1.1 Modul*Connection and Pairing*



Gambar 3.3 *Flowchart* Modul *Connection and Pairing*

Gambar 3.3 menunjukkan alur proses yang dilakukan oleh modul *Connection and Pairing*. Proses dimulai dengan mencari semua perangkat *Bluetooth* yang terdeteksi dalam jangkauan *Bluetooth* milik komputer. Karena proses ini menggunakan bantuan 32feet.net *library* yang merupakan *library* untuk membantu koneksi *Bluetooth* secara umum, maka pencarian perangkat *Bluetooth* dilakukan untuk semua perangkat dan tidak terbatas pada Wiimote saja. Setalah proses pencarian selesai, masing-masing perangkat akan diperiksa apakah perangkat tersebut adalah Wiimote atau bukan. Jika perangkat yang terdeteksi merupakan Wiimote, proses akan dilanjutkan ke proses *pairing* agar komunikasi antara Wiimote dan komputer dapat terjalin. Koneksi ini kemudian akan diteruskan ke modul *Interpreter and Command* yang bertugas melakukan membaca perubahan *state* yang terjadi pada Wiimote. Jika perangkat Wiimote tidak ditemukan, proses tidak dilanjutkan dan mengembalikan status bahwa Wiimote tidak ditemukan. Apabila terdapat lebih dari satu Wiimote yang terdeteksi oleh modul, aplikasi hanya akan melakukan *pairing* dengan Wiimote pertama yang ditemukan.

3.5.1.2 Modul *Calibration*



Gambar 3.4 *Flowchart* Modul *Calibration*

Seperti yang sudah disebutkan sebelumnya, tugas dari modul *Calibration* adalah mendapatkan koordinat-koordinat yang akan

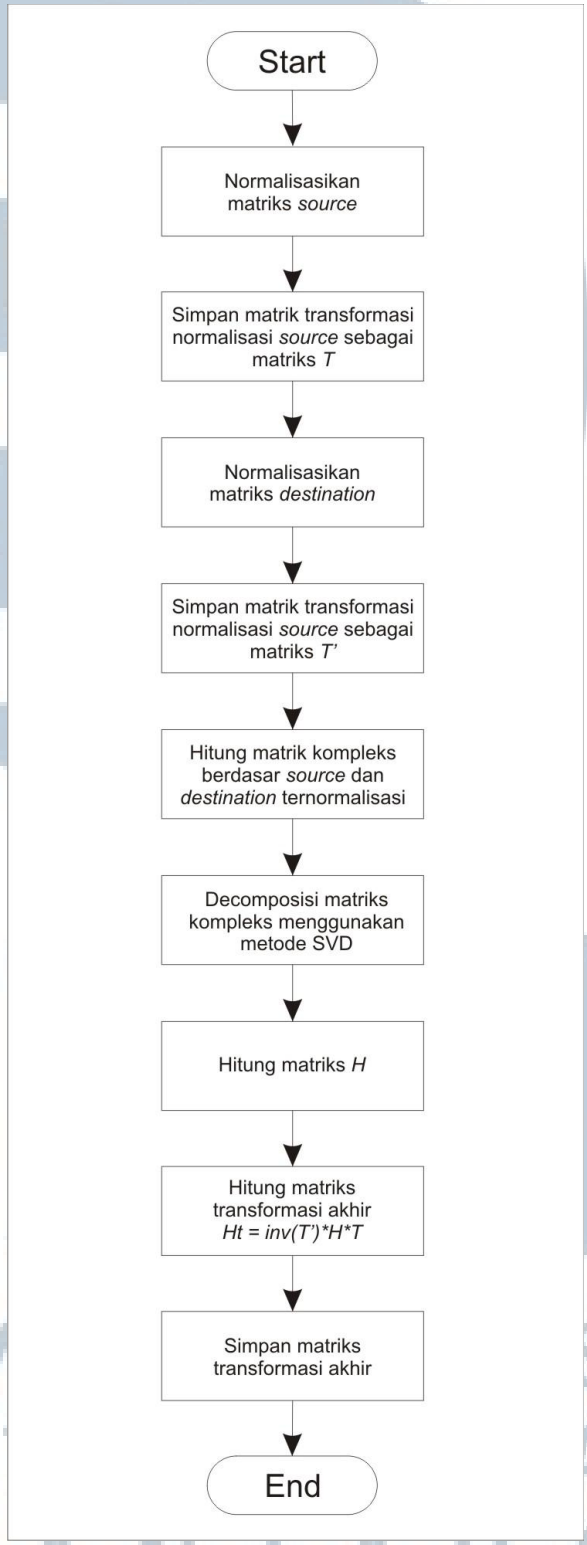
digunakan oleh modul *Warper* pada proses *Direct Linear Transformation*. Pada gambar 3.4 dapat dilihat bahwa modul ini hanya berjalan ketika *Form* Kalibrasi ditampilkan. Proses pertama yang dilakukan oleh modul ini adalah penghitungan koordinat sebenarnya pada layar (*pixel*). Modul ini akan menghitung empat titik di empat sudut layar dengan *margin* 10% dari ukuran *pixel* layar. Dengan perhitungan persentase ini, aplikasi dapat digunakan untuk berbagai ukuran layar.Keempat titik ini kemudian disebut sebagai titik *destination.*

Titik-titik *destination* ini kemudian ditampilkan pada layar agar pengguna dapat memberikan koordinat berdasar perspektif kamera Wiimote yang berhubungan dengan masing-masing titik *destination*.Titik-titik dari perspektif kamera ini kemudian disebut sebagai titik *source*. Keempat pasang titik *source* dan *destination* ini kemudian disimpan dan akan digunakan untuk perhitungan *Direct Linear Transformation* sebagai fungsi pemetaan.

3.5.1.3 Modul *Warper*(*Planar Homography*)

Modul *Warper* melakukan dua tugas utama dalam aplikasi, yaitu menghitung matriks transformasi menggunakan metode *Planar Homography* dan memetakan koordinat dari perspektif kamera (*source*) ke koordinat di tampilan layar.

1. Menghitung Matriks Transformasi



Gambar 3.5 *Flowchart* Penghitungan Matriks Transformasi

Langkah pertama dari proses penghitungan matriks transformasi adalah melakukan normalisasi pada koordinat-koordinat *source* dan *destination* yang diketahui. Normalisasi dilakukan dengan melalui berbagai tahap.Tahap pertama adalah menghitung titik pusat dari keempat koordinat dengan mencari rata-rata dari nilai *x* dan *y* yang dimiliki masing-masing koordinat.Kemudian koordinat asal dikurangi dengan koordinat pusatnya untuk mendapat koordinat baru.Koordinat baru hasil pengurangan tersebut kembali dicari faktor skalanya agar panjang vector dari koordinat masing-masing ke titik pusat adalah $\sqrt{2}$. Matriks transformasi translasi dan skalasi yang didapat kemudian dikalikan untuk mendapatkan matriks transformasi normalisasi.Pada koordinat *source*, matriks transformasi normalisasi ini disimpan sebagai matriks *T*,sedangkan pada koordinat *destination*, matriks transformasi normalisasi ini disimpan sebagai matriks *T'*. Kedua matriks ini akan digunakan kembali pada saat menghitung matriks transformasi pemetaan akhir dari metode *Direct Linear Transformation*.

Langkah berikutnya adalah membuat matriks kompleks yang disusun dari pasangan koordinat *source* dan *destination* yang sudah ternormalisasi. Menurut teori *Direct Linear Transformation*, dengan 4 pasang titik yang diketahui, maka akan dihasilkan sebuah matriks kompleks dengan dimensi 8x9 seperti berikut:

$$\begin{pmatrix} 0 & 0 & 0 & -x_1 & -y_1 & -1 & y'_1x_1 & y'_1y_1 & y'_1 \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 & -x'_1 \\ 0 & 0 & 0 & -x_2 & -y_2 & -1 & y'_2x_2 & y'_2y_2 & y'_2 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x'_2x_2 & -x'_2y_2 & -x'_2 \\ 0 & 0 & 0 & -x_3 & -y_3 & -1 & y'_3x_3 & y'_3y_3 & y'_3 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x'_3x_3 & -x'_3y_3 & -x'_3 \\ 0 & 0 & 0 & -x_4 & -y_4 & -1 & y'_4x_4 & y'_4y_4 & y'_4 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x'_4x_4 & -x'_4y_4 & -x'_4 \end{pmatrix}$$

di mana x_i dan $y_i$adalah koordinat *source* sedangkan x'_i dan y'_iadalah koordinat *destination*. Matriks ini lalu akan didekomposisi menggunakan metode *Singular Value Decomposition* dan menghasilkan tiga buah matriks yaitu matriks U, S, dan V. Dari ketiga matriks tersebut, yang akan digunakan untuk langkah selanjutnya hanya matriks V.

Dari matriks V yang memiliki dimensi 9x9, nilai yang ada pada kolom terakhir diambil dan diubah menjadi matriks transformasi *H*dengan format sebagai berikut:

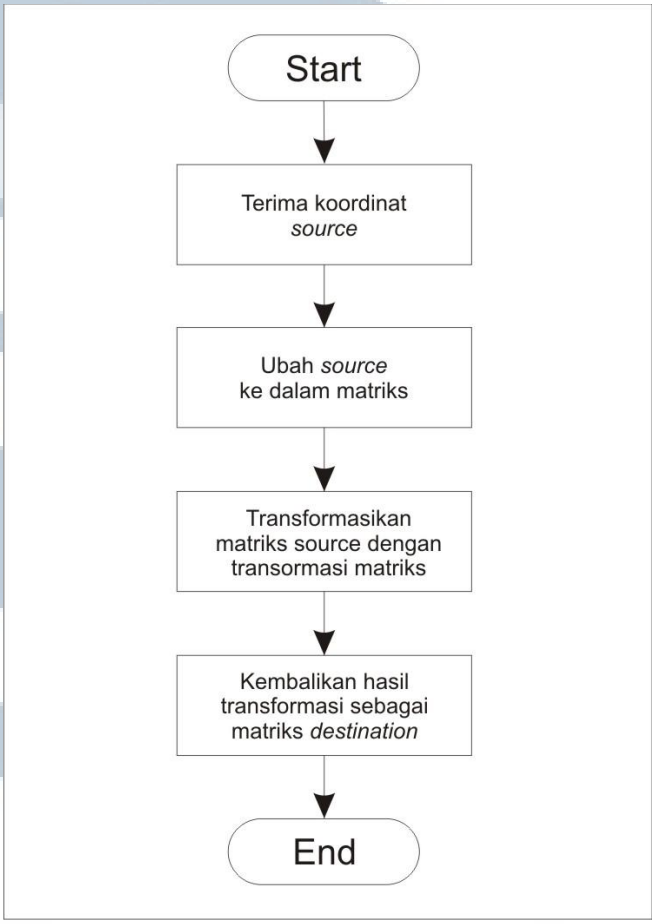
$$H = \begin{bmatrix} V_{19} & V_{29} & V_{39} \\ V_{49} & V_{59} & V_{69} \\ V_{79} & V_{89} & V_{99} \end{bmatrix}$$

Setelah mengetahui matriks H, maka matriks transformasi akhir dapat dihitung dengan rumus

$$H_t = inv(T')(H)(T)$$

Matriks $H_t$inilah yang kemudian disimpan sebagai matriks transformasi dan yang akan digunakan pada proses pemetaan titik *source* ke *destination.*

2. Memetakan Koordinat *Source* ke *Destination*



Gambar 3.6 *Flowchart* Pemetaan Koordinat

Pada proses pemetaan ini, koordinat *source* berdasar perspektif kamera Wiimote ditransformasikan dengan menggunakan matriks transformasi yang sudah didapat pada proses sebelumnya. Langkah pertama, koordinat *source* diubah menjadi bentuk matriks $\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$. Dengan bentuk matriks seperti itu, maka proses transformasi cukup dilakukan dengan cara mengalikan matriks *source* dengan matriks transformasi.

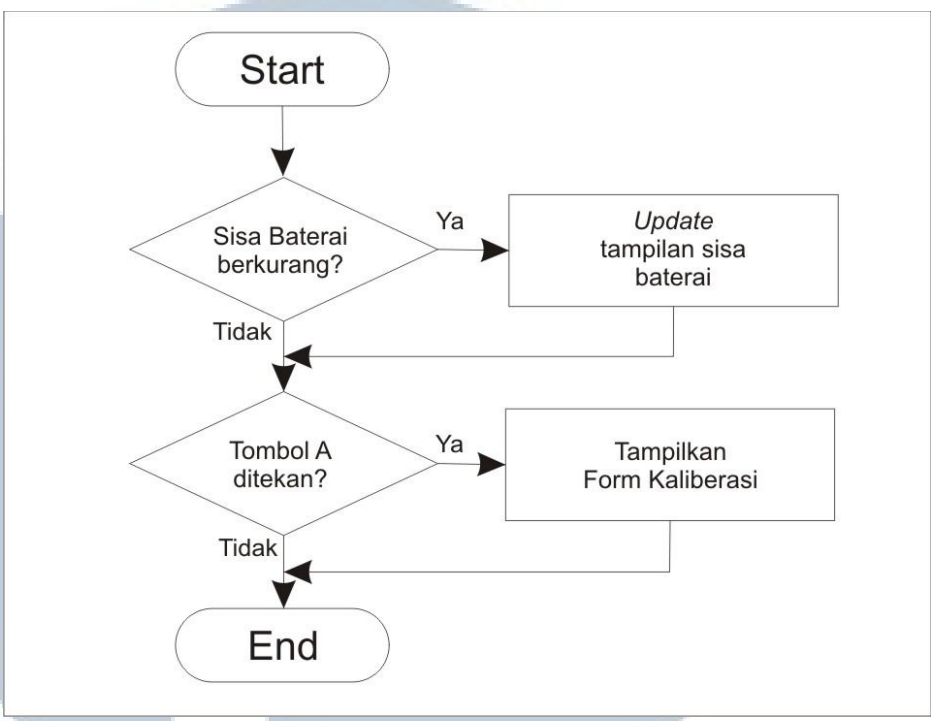
$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Hasil dari perkalian matriks tersebut lalu disederhanakan agar kembali dalam bentuk $\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix}$.Maka kita mendapatkan koordinat yang sesungguhnya pada *pixel* layar yaitu x' dany'.Koordinat tersebut kemudian disimpan sebagai koordinat tujuan titik interaksi di mana interaksi dilakukan oleh pengguna. Koordinat ini akan digunakan oleh modul *Interpreted and Command* untuk menentukan di titik mana interaksi dilakukan pada tampilan layar.

3.5.1.4 Modul *Interpreter and Command*

Modul *Interpreter and Command* memiliki tugas utama untuk melakukan komunikasi dengan Wiimote dan menerima bentuk perubahan *state* yang terjadi pada Wiimote. Pada modul ini terdapat dua buah sub-modul yang berjalan secara *asynchronous* dengan aplikasi agar perubahan state dapat diterima secara dinamis dan terus menerus.

Sub-modul pertama bertugas untuk menerima perubahan state berupa penekanan tombol pada Wiimote dan perubahan sisa baterai yang dimiliki oleh Wiimote. Alur dari sub-modul tersebut sebagai berikut :

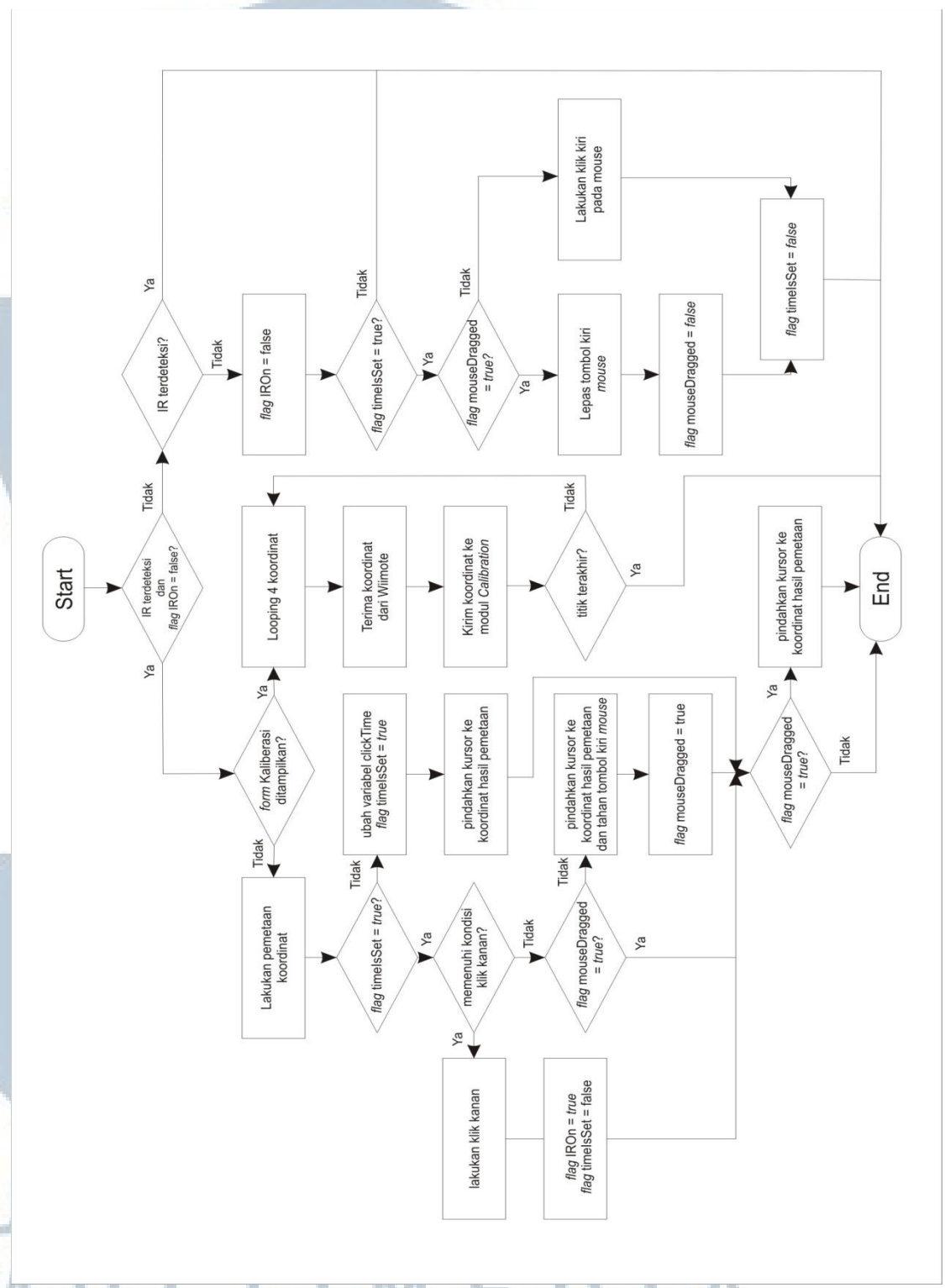


Gambar 3.7 *Flowchart* Sub-Modul *Button and Battery Listener*

Sub-modul ini akan terus menerus melakukan *polling* terhadap Wiimote dan menerima perubahan *state* khusus untuk perubahan sisa baterai dan penekanan tombol A pada Wiimote. Pertama modul akan bertanya apakah daya baterai yang menjadi sumber listrik bagi Wiimote sudah berkurang. Jika ya, maka modul akan melakukan *update* pada tampilan antarmuka aplikasi yang menampilkan sisa daya baterai.Selanjutnya, modul juga menanyakan apakah tombol 'A' pada Wiimote ditekan atau tidak. Jika ya, modul akan memerintahkan untuk menampilkan *Form* Kalibrasi dan modul *Calibrartion* akan mulai bekerja.

Sub-modul berikutnya bertugas untuk menangkap dan menginterpretasikan interaksi yang dilakukan oleh pengguna melalui

cahaya inframerah.Aplikasi ini memungkinkan tiga interaksi yang dilakukan oleh pengguna, yaitu klik kiri, klik kanan, dan *drag click*. Klik kiri dilakukan pengguna dengan menekan tombol IR Pen dan langsung melepasnya, klik kanan dilakukan dengan cara menekan dan menahan tombol IR Pen di satu koordinat selama satu detik, dan *drag click* dilakukan dengan cara menekan dan menahan tombol IR Pen lalu memindah koordinat IR Pen sesuai keinginan. Gambar 3.8 menunjukkan alur kerja dari sub-modul ini.



Gambar 3.8 *Flowchart* Sub-Modul *IR Listener*

Sub-modul ini akan menanyakan apakah Wiimote mendeteksi cahaya inframerah atau tidak secara terus-menerus selama aplikasi berjalan. Dalam melakukan prosesnya, sub-modul ini akan memiliki beberapa *flag*, yaitu IROn, timeIsSet, dan mouseDragged. *Flag* IROn digunakan mendeteksi apakah inframerah sudah terlihat sebelumnya, *flag* timeIsSet digunakan untuk mendeteksi apakah pengguna menekan tombol IR Pen dalam waktu lama, dan *flag* mouseDragged digunakan untuk mendeteksi apakah pengguna melakukan *drag click* atau tidak.*Flag* IROn bernilai benar apabila cahaya inframerah sudah terlihat pada proses sebelumnya dan bernilai salah jika cahaya inframerah tidak terlihat dip roses sebelumnya. *Flag* timeIsSet bernilai benar selama user tengah melakukan interaksi. Sementara *flag* mouseDragged bernilai benar selama pengguna melakukan interaksi *drag click.*

Pada langkah pertama, aplikasi akan menanyakan apakah terdapat inframerah terdeteksi dan *flag* IROn dalam kondisi salah. Jika benar, aplikasi akan memeriksa apakah *Form* Kalibrasi sedang tampil di layar. Jika ya, maka aplikasi akan melakukan proses kalibrasi sampai selesai dan sub-modul ini bertugas mengirimkan koordinat kalibrasi pada modul *Calibration*. Jika tidak, artinya pengguna sedang melakukan interaksi normal dengan tampilan. Koordinat yang didapat pada interaksi normal ini dikirim ke modul *Warper* untuk dipetakan dan mendapat koordinat *pixel* tempat interaksi dilakukan.

Setelah mendapat koordinat tempat interaksi dilakukan, sub-modul akan mengartikan jenis interaksi yang sedang dilakukan user. Aplikasi akan memeriksa *flag* timeIsSet bernilai benar atau salah. Jika salah, artinya pengguna baru saja memulai interaksi dan jenis interaksi akan ditentukan pada proses berikutnya. Jika benar, artinya pengguna tengah melakukan interaksi klik kanan ataupun *drag click*. Jika interaksi memenuhi kondisi klik kanan, yaitu menahan tombol IR Pen selama satu detik di koordinat yang sama, aplikasi akan menentukan bahwa jenis interaksi adalah klik kanan dan mengirim perintah klik kanan pada koordinat yang dimaksud. Akan tetapi, jika pengguna memindah koordinat selama menahan tombol IR, aplikasi akan menentukan bahwa jenis interaksi adalah *dragclick* dan membuat nilai *flag* mouseDragged menjadi benar.

Kembali pada pertanyaan pertama, apakah terdapat inframerah terdeteksi dan *flag* IROn dalam kondisi salah. Jika pertanyaan keadaan tidak sesuai dengan pertanyaan, aplikasi akanmemeriksa apakah ada cahaya inframerah yang terdeteksi. Jika tidak ada, maka *flag* IROn akan dibuat bernilai salah dan aplikasi memeriksa apakah pengguna tengah melakukan interaksi sebelumnya.

Jika *flag* timeIsSet masih bernilai benar, artiya pengguna baru saja mengakhiri interaksinya. Jika interaksi berupa *drag click*, maka *dragclick* akan diakhiri, dan jika bukan *drag click*, pengguna hanya melakukan klik kiri dan perintah dikirimkan ke komputer.

3.5.2. Perancangan Tatap-muka Aplikasi

Aplikasi yang dibuat akan memiliki tiga tampilan antar-muka, yaitu tampilan antar-muka utama, tampilan antar-muka kalibrasi, dan tampilan antar-muka pencarian Wiimote. Selanjutnya akan dijelaskan lebih lanjut mengenai komponen masing-masing antar-muka dan interaksinya.

3.5.2.1 Tampilan Antar-Muka Utama

Gambar 3.9 Tampilan Antar-Muka Utama

Tampilan antar-muka utama merupakan tampilan utama yang dimiliki oleh aplikasi. Seperti yang digambarkan pada gambar 3.9,

tampilan antar-muka utama memiliki banyak komponen yang memberikan informasi dan kemampuan interaksi pada pengguna.

Komponen paling atas dari tampilan ini adalah *menu toolbar*. Seperti aplikasi pada umumnya, *menu toolbar* memiliki menu "*File*" dan "*Help*". *Menu* "*File*" sendiri diturunkan menjadi dua pilihan *menu*, yaitu "*Connect Wiimote*" dan "*Exit*". *Menu*"*Connect Wiimote*" berfungsi untuk memulai pencarian Wiimote. Apabila pengguna melakukan klik pada *menu* ini, aplikasi akan menampilkan tampilan pencarian Wiimote dan memulai pencarian Wiimote. Jika Wiimote sudah terhubung dengan aplikasi, *menu* "*Connect Wiimote*" menjadi tidak aktif.Sementara *menu* "*Exit*" memiliki fungsi menutup aplikasi.*Menu* "*Help*" sendiri hanya memiliki satu *menu* turunan, yaitu menu "*About*" yang berfungsi menampilkan informasi tentang aplikasi.

Komponen berikutnya adalah sebuah *progress bar*yang menampilkan informasi mengenai sisa daya baterai yang dimiliki oleh Wiimote yang terhubung.*Progress bar* ini akan menunjukkan sisa daya baterai dalam bentuk persentase. Semakin banyak sisa daya baterai, progress bar akan semakin penuh terisi dan sebaliknya.

Komponen selanjutnya dari tampilan antar-muka ini adalah sebuah *track bar* yang menunjukkan *smoothing level* yang digunakan oleh aplikasi. Dengan menggunakan komponen ini, pengguna dapat mengubah nilai *smoothing level* yang digunakan aplikasi dalam jangkauan nilai 0 sampai 10. Semakin besar nilai *smoothing* yang

digunakan, maka semakin halus juga aplikasi mengakomodasi *drag click* yang dilakukan oleh pengguna dan sebaliknya.

Tampilan utama ini juga memiliki sebuah tombol yang berfungsi untuk memulai proses kalibrasi. Apabila pengguna menekan tombol ini, aplikasi akan menampilkan tampilan antar-muka kalibrasi dan memulai proses kalibrasi. Tombol ini memiliki fungsi yang sama dengan tombol A yang ada pada Wiimote.

Komponen terakhir adalah sebuah *panel* yang berfungsi menampilkan *tracking* yang didapatkan oleh Wiimote setelah proses kalibrasi. Yang dimaksud dengan *tracking* di sini adalah posisi dan tampilan komputer berdasar sudut pandang kamera.*Panel* ini akan memiliki warna hitam sebagai dasar dan pada bagian di mana tampilan komputer berada akan berwarna putih. Persentase dari *tracking* ini didapat dari luas tampilan komputer di hadapan kamera Wiimote dibagi dengan luas panel.

Komponen terakhir adalah *status bar* yang berada di bagian paling bawah tampilan.Komponen ini berupa tulisan yang menunjukkan informasi mengenai hubungan antara aplikasi dengan Wiimote. Jika tidak ada Wiimote yang terhubung, komponen akan menampilkan tulisan "*No* Wiimote *connected*", sebaliknya, jika Wiimote terhubung dengan aplikasi, komponen akan menampilkan tulisan "*Wiimote connected*".

Semua komponen pada tampilan antar-muka utama ini hanya aktif jika ada Wiimote yang terhubung dengan aplikasi, kecuali komponen *menu toolbar*.Jika tidak ada Wiimote yang terhubung, user tidak dapat melakukan interaksi pada komponen-komponen yang ada.

3.5.2.2 Tampilan Antar-Muka Pencarian Wiimote

Tampilan kedua yang dimiliki aplikasi adalah tampilan antar-muka pencarian Wiimote.Tampilan ini muncul secara otomatis saat aplikasi dibuka apabila belum ada Wiimote yang terhubung dengan komputer. Jika sudah ada Wiimote yang terhubung, aplikasi akan langsung menampilkan tampilan antar-muka utama. Tampilan antar-muka pencarian Wiimote ini juga ditampilkan ketika pengguna menekan *menu* "*Connect* Wiimote" dari tampilan antar-muka utama.

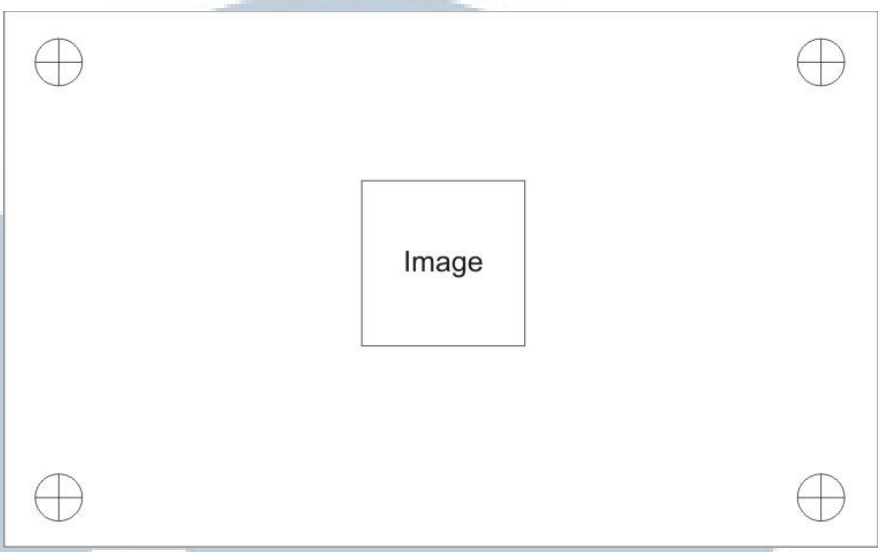
Tampilan antar-muka ini digambarkan dalam gambar 3.10.

Gambar 3.10 Tampilan Antar-Muka Pencarian Wiimote

Tampilan antar-muka pencarian memiliki sebuah komponen *image*, *progress bar*, dan *status bar*.Komponen *image* seperti pada gambar 3.10 berisi logo UMN sebagai instansi pendukung pembuatan aplikasi ini. Komponen *progress bar* hanya berfungsi sebagai pemanis yang menunjukkan bahwa proses pencarian Wiimote dan *pairing* sedang dilakukan oleh aplikasi. Sementara komponen *status bar* berfungsi menampilkan status tahap pencarian yang dilakukan aplikasi.*Status bar*akan menampilkan tulisan "*Searching* Wiimote", "Wiimote *found*", "*Installing* Wiimote", dan "Wiimote *installed*" sesuai dengan tahap yang tengah dilakukan oleh aplikasi. Apabila aplikasi gagal menemukan Wiimote, *status bar*akan menampilkan tulisan "*Failed to find* Wiimote".

3.5.2.3 Tampilan Antar-Muka Kalibrasi

Tampilan antar-menu kalibrasi merupakan tampilan dari aplikasi yang ditampilkan secara *fullscreen* dengan tujuan mendapatkan pasangan koordinat dari tampilan komputer dan koordinat berdasar perspektif kamera.Tampilan antar-muka kalibrasi ditunjukkan pada gambar 3.11.



Gambar 3.11 Tampilan Antar-Muka Kalibrasi

Seperti yang ditunjukkan gambar 3.11, tampilan antar-muka ini terdiri dari sebuah komponen *image* dan empat buah komponen yang menyerupai sasaran tembak pada keempat sudut tampilan. Sama seperti tampilan antar-muka pencarian Wiimote, komponen *image* juga diisi dengan logo UMN. Sementara untuk komponen sasaran tembak akan muncul secara bergantian dengan urutan sebagai berikut: kiri atas, kanan atas, kiri bawah, dan kanan bawah.

Fungsi dari komponen sasaran tembak ini adalah agar pengguna dapat menggunakan IR Pen tepat pada titik tersebut sehingga koordinat cahaya inframerah dapat ditangkap oleh kamera Wiimote dan diterima oleh aplikasi.Interaksi ini menghasilkan pasangan koordinat pada *pixel* tampilan komputer dan koordinat berdasar perspektif kamera. Komponen sasaran tembak akan muncul dan menghilang secara bergantian sesuai dengan urutan yang telah disebutkan sebelumnya.

Komponen akan hilang setelah Wiimote menemukan koordinat cahaya inframerah dan komponen sasaran tembak berikutnya akan muncul. Setelah keempat komponen muncul dan menghilang, tampilan antar-muka kalibrasi akan tertutup dan aplikasi memulai melakukan proses penghitungan kalibrasi.

3.5.3. Fitur-fitur Aplikasi

Aplikasi yang akan dibangun akan memiliki beberapa fitur selain dari fungsi utama dari aplikasi itu sendiri. Fitur-fitur tersebut antara lain:\

1. Fitur *Save Calibration*

Fitur ini memungkinkan aplikasi untuk menyimpan perhitungan kalibrasi dan transformasi yang terakhir digunakan oleh aplikasi.Perhitungan ini akan disimpan dalam sebuah *text file* dengan format tertentu.Fitur simpan ini dijalankan secara otomatis saat aplikasi ditutup.

2. Fitur *Load Calibration*

Fitur ini membaca *text file* hasil penyimpanan kalibrasi yang dibuat oleh aplikasi. Data pada *text file* tersebut kemudian digunakan langsung oleh aplikasi untuk proses pemetaan koordinat. Fitur ini dijalankan secara otomatis ketika aplikasi mulai berjalan. Dengan adanya fitur *save* dan *load*, maka apabila posisi Wiimote dan tampilan komputer tidak berubah, proses kalibrasi tidak perlu dilakukan kembali.

3. Fitur klik kanan

Meskipun klik kanan termasuk dalam interaksi utama yang dilakukan aplikasi, klik kanan pada aplikasi ini dapat dikategorikan sebagai fitur karena agak berbeda dengan klik kanan pada aplikasi sebelumnya. Pada aplikasi yang dibangun, klik kanan dilakukan dengan menahan tombol IR Pen di titik tertentu sehingga cahaya inframerah muncul di koordinat yang sama dalam waktu tertentu. Metode ini lebih praktis bila dibandingkan dengan fitur yang ada pada aplikasi Pentabulous!, di mana user harus menekan tombol klik kanan pada antar-muka aplikasi terlebih dahulu lalu menekan di koordinat yang diinginkan.

4. Fitur *Smoothing*

Fitur ini bekerja saat pengguna melakukan interaksi *drag*. Fitur smoothing memungkinkan pengguna mendapatkan perubahan titik yang lebih halus.Fitur ini menghitung rata-rata dari perubahan koordinat yang terjadi sesuai dengan nilai *smoothing* yang ditentukan lalu mengembalikannya pada modul *Interpreter and Command* sebagai koordinat interaksi.