Pemanfaatan *Accelerometer* Pada Telepon Genggam Berbasis Android Sebagai Kendali *Mobile Robot*

[1)]Christoforus Surjoputro
[1)]Program Studi S1 Sistem Komputer STIKOM Surabaya. Email: cs_sanmar@yahoo.com

Abstract

During this time, used of accelerometer sensor is very helpful in human life, generally in automotive needed.Use of this sensor as a detector for airbag in car is worthy to be removed or not in cars accident event. The other used of this sensor is to see the movement of aircraft from its starting point. The conclution of the used of accelerometer sensor in automotive field wal limited to making a decision and see the movement of object.

In mobile phone technology, this sensor is able to used as an interaction between user and applications. Generally for game application. There are several games that used this sensor to perform some actions. It can also be seen that the utilization in mobile phone technology was limited to the interaction of an application.

From both limitation, the authors attempt to use the sensor in other field with other uses. This time, the sensor is used to control a mobile robot from the axis change that are detected by sensor on mobile phone. Users are expected to get more experience in utilization of this sensor, the user can control the mobile robot as playing games in mobile phone.

This application are meneged to control the mobile robot based on the axis change that occur in mobile phone. Beside that, this application also managed to show images that captured from mobile robot so the user can control the mobile robot although the robot can't be seen. There is a problem that the mobile phone has a long respon to show an image on the display and it disturb the user.

Keyword: Mobile Robot, Robotino, Telepon Genggam, Android, *accelerometer.*

Telepon genggam saat ini juga sudah memiliki sistem operasi. Seperti halnya sistem operasi pada komputer, sistem operasi (SO) telepon genggam adalah *software* utama yang melakukan manajemen dan kontrol terhadap *hardware* secara langsung serta sebagai manajemen dan kontrol *software-software* lain sehingga *software-software* lain bekerja (Prawita, 2010).

Robot ini memiliki kemudahan dalam bergerak, karena dapat bergerak secara mudah tidak hanya ke depan dan belakang namun juga ke samping kiri dan kanan bahkan serong kiri dan kanan (Braunl, 2006).

Jika kedua teknologi di atas digabungkan, yaitu teknologi telepon genggam dengan sensor *accelerometer* dan teknologi robot omni yang dapat bergerak ke segala arah yang dapat digabungkan dengan teknik tertentu. Sehingga robot omni bergerak sesuai dengan kemiringan telepon genggam.

Pengertian Client Server

Client-Server adalah arsitektur jaringan yang memisahkan *client*(biasanya aplikasi yang menggunakan GUI) dengan *server*. Masing-masing client dapat meminta data atau informasi dari server (Anwar, dkk. 2011).

Android

Pada tahun 2005 Google mengakuisisi Android Inc yang pada saat itu dimotori oleh Andy Roubin, Rich Miner, Nick Sears, dan Chris White. Semua aplikasi yang dibuat untuk android akan memiliki akses yang setara dalam mengakses seluruh kemampuan *handset*, tanpa membedakan apakah itu merupakan aplikasi inti atau aplikasi pihak ketiga. Dalam kata lain dengan platform android

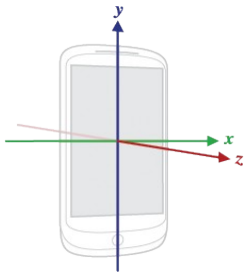
ini, *programmer* atau *developer* secara penuh akan bisa mengustomisasi perangkat androidnya (Mulyadi, 2010).

Sebuah *activity* merepresentasikan sebuah layar dengan suatu *user interface*. Sebagai contoh , sebuah aplikasi *email* mempunyai sebuah *activity* yang menampilkan list dari *email*, sebuah *activity* lain untuk membuat *email*, dan *activity* lain untuk membaca *email* (Android, 2011).

Cara menggakses sensor pada telepon genggam berbasis android yaitu menggunakan API `android.hardware`. Berikut adalah cuplikan program untuk menggunakan sensor accelerometer pada android(Android, 2011).

```
public class SensorActivity extends
Activity,implements
SensorEventListener {
private final SensorManager
mSensorManager;
private final Sensor
mAccelerometer;
public SensorActivity() {
mSensorManager =
(SensorManager)getSystemService(SEN
SOR_SERVICE);
mAccelerometer =
mSensorManager.getDefaultSensor(Sen
sor.TYPE_ACCELEROMETER);
}
protected void onResume() {
super.onResume();
mSensorManager.registerListener(thi
s, mAccelerometer,
SensorManager.SENSOR_DELAY_NORMAL);
}
protected void onPause() {
super.onPause();
mSensorManager.unregisterListener(t
his);
}
public void
onAccuracyChanged(Sensor sensor,
int accuracy) {
}
public void
onSensorChanged(SensorEvent event)
{
// alpha is calculated as t / (t +
dT)
// with t, the low-pass filter's
time-constant
// and dT, the event delivery rate
final float alpha = 0.8;
gravity[0] = alpha * gravity[0] +
(1 - alpha) * event.values[0];
gravity[1] = alpha * gravity[1] +
(1 - alpha) * event.values[1];
gravity[2] = alpha * gravity[2] +
(1 - alpha) * event.values[2];
linear_acceleration[0] =
event.values[0] - gravity[0];
linear_acceleration[1] =
event.values[1] - gravity[1];
linear_acceleration[2] =
event.values[2] - gravity[2];
}
}
```

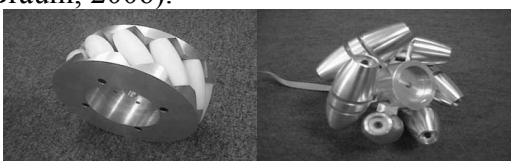
Sumbu x adalah posisi horisontal dari telepon genggam, sumbu y adalah posisi vertikal dari telepon genggam, dan sumbu z merupakan sumbu yang mengarah keluar dari layar telepon genggam. Dalam sistem ini, oordinat belakang layar memiliki nilai negatif z. Berikut adalah gambar dari posisi sumbu dari telepon genggam(Android, 2011).



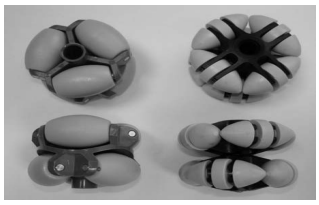
Gambar 1. Sistem Koordinat Dari Telepon Genggam

Omni-Directional Robots

Keajaiban dari pengendalian *omni-directional* adalah roda mecanum. Roda ini telah dikembangkan dan dipatenkan oleh perusahaan dari swedia Mecanum AB dengan Bengt Ilon pada tahun 1973 sehingga roda ini sudah ada cukup lama (Braunl, 2006).



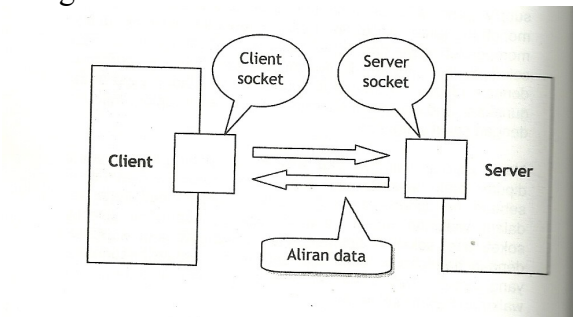
Gambar 3. Roda Mecanum dengan Putaran 45^{o}



Gambar 2. Roda Mecanum dengan Putaran 90^{o}

SOCKET

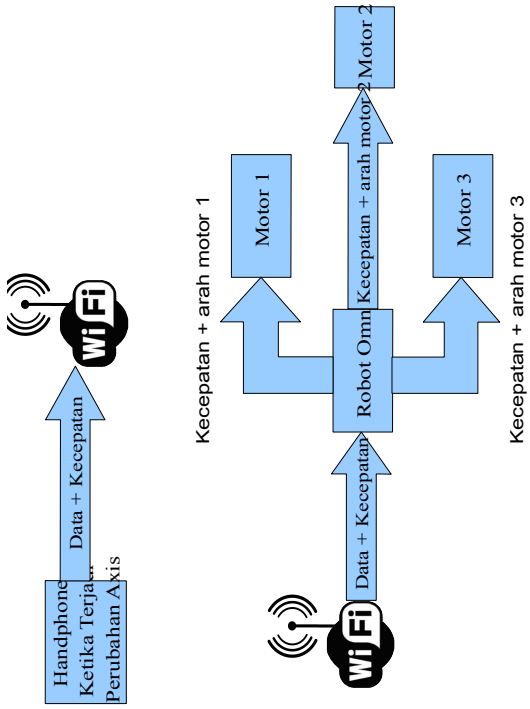
Soket adalah komponen yang bertugas sebagai penghubung antara satu peralatan dengan peralatan lain. Java menyediakan class *socket* yang merupakan class dasar untuk bisa melakukan konektivitas jaringan antar komputer. Kondisi objek soket baik di sisi *client* maupun *server* bisa digambarkan sebagai berikut:



Gambar 3. Kondisi Objek *Socket* di Sisi *Client* dan *Server*

Server adalah pihak yang selalu menunggu request dari client. Dengan demikian client adalah program yang memulai koneksi dalam suatu jaringan (Purnama, 2005).

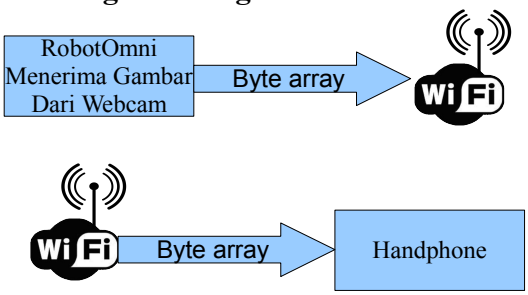
Blok Diagram Pengiriman Data Kontrol Robot



Gambar 4. Blok Diagram Pertama, Pengiriman Kecepatan

Dari blok diagram pada gambar 4 terlihat terdapat dua bagian utama, yaitu bagian pengirim dan bagian penerima. Bagian pengirim dari blok di atas yaitu telepon genggam, sedangkan bagian penerima yaitu robot omni. Cara kerja dari blok diagram ini, ketika terjadi perubahan axis pada telepon genggam, telepon genggam tersebut akan memproses axis tersebut untuk dirubah ke nilai bilangan asli baik positif maupun negatif. Setelah berubah menjadi satuan bilangan asli, data tersebut dikirimkan ke robot omni melalui *socket* dengan perantara jaringan *wireless*. Setelah robot menerima data tersebut, robot akan mengolah bilangan asli positif atau negatif tersebut menjadi kecepatan masing-masing motor dari robot omni.

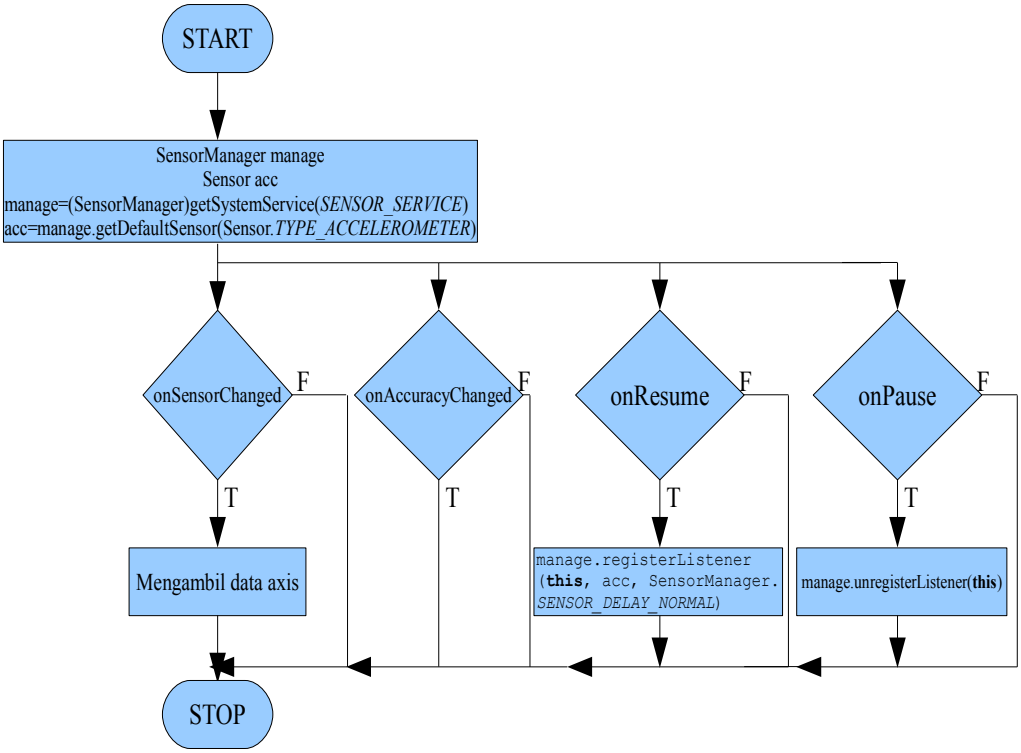
Blok Diagram Pengiriman Data Gambar



Gambar 5. Blok Diagram Pengiriman Gambar

Dari blok diagram pada gambar 5, juga terdapat 2 buah bagian yaitu pengirim dan penerima. Bagian pengirim adalah robot omni, sedangkan penerimanya adalah telepon genggam. Cara kerja dari pengiriman video ini adalah robot menerima sebuah frame dari *webcam*. Setelah robot menerima frame tersebut, robot akan mengirimkan frame yang berupa gambar tersebut melalui *socket* dengan perantara jaringan *wireless*. Kemudian frame tersebut diterima oleh telepon genggam dan ditampilkan di layar telepon genggam tersebut.

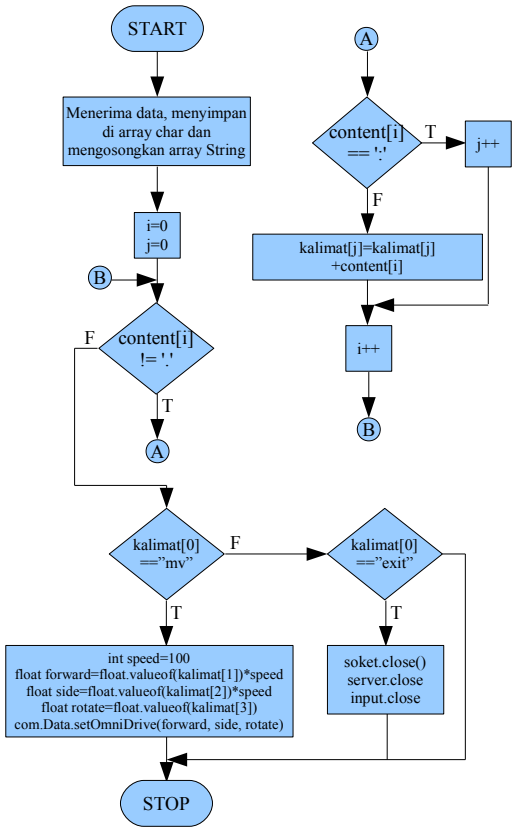
Proses Pendeteksian Perubahan Axis



Gambar 6. Blok Diagram Pendeteksian Axis

Untuk melakukan pendeteksian perubahan axis dengan sensor *accelerometer*, diperlukan tiga komponen yaitu SensorManager, Sensor, dan SensorEventListener. Untuk menggunakan suatu sensor, objek SensorManager harus dihubungkan terlebih dahulu dengan sistem android yang mengatur bagian sensor. Selain itu, sensor juga harus ditentukan ingin menggunakan sensor *accelerometer*. Untuk bisa mendeteksi sensor, suatu class harus didaftarkan terlebih dahulu. Setelah didaftarkan, setiap ada perubahan axis, android akan memanggil fungsi onSensorChanged. Di dalam fungsi itulah, proses pengambilan nilai axis dari posisi telepon genggam dilakukan. Ketika sudah tidak menggunakan sensor lagi, terjadi proses penghapusan class yang bersangkutan dari daftar. OnAccuracy changed tidak digunakan karena tidak melakukan perubahan *accuracy* pada sensor *accelerometer*.

Komunikasi Untuk Menggerakkan Robot Omni



Gambar 7. Blok Diagram Pengolahan Data Pada Robot Omni

Dari blok diagram di atas, tampak jelas bahwa telepon genggam hanya mengirimkan dua kriteria, yaitu mv dan exit. Mv ini berfungsi untuk memberikan perintah pada robot omni untuk menggerakkan robot dengan arah dan kecepatan yang sudah dikirimkan bersamaan dengan perintah tersebut. Untuk pengiriman data kecepatan dari telepon genggam yaitu "mv:(kec.maju):(kec.samping):(kec.putar)". Penerimaan di robot omni memisahkan setiap bagian dengan pendeteksian tanda ':'. Jadi isi dari kalimat[0] adalah "mv", kalimat[1] adalah kecepatan maju berupa bilangan asli, kalimat[2] adalah kecepatan bergerak ke samping berupa bilangan asli, dan kalimat[3] adalah kecepatan putar dalam derajat/detik. Kecepatan maju dan ke samping hanya berupa transmisi dalam bentuk bilangan asli baik positif maupun negatif, sehingga perlu dikalikan dengan speed supaya menjadi kecepatan yang sebenarnya, yaitu dalam bentuk mm/detik. Untuk berhenti berkomunikasi, telepon genggam hanya mengirimkan "exit.". Jadi *array* kalimat[] yang terisi hanya kalimat[0] dengan isi "exit". Untuk mengakhiri pembacaan *array char* yang berisi data dari *socket*, setiap pengiriman selalu diakhiri dengan tanda titik('.').

Pengiriman yang dilakukan telepon genggam menggunakan perintah seperti di bawah ini.

```
DataOutputStream delivered = new
DataOutputStream(soket.getOutputStr
eam());
delivered.writeBytes(data);
```

Pengiriman data menggunakan objek `DataOutputStream` yang dihubungkan dengan *outputstream* milik koneksi *socket*. Metode pengiriman yang

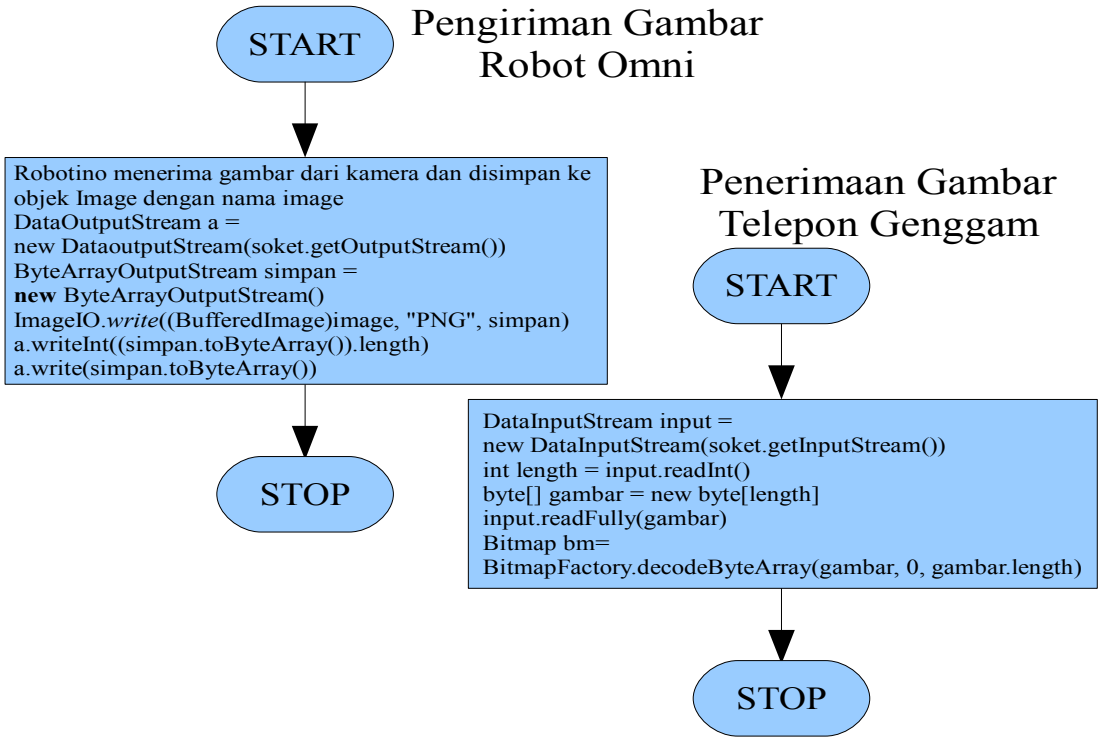
digunakan yaitu writeBytes dengan data yang dikirim berupa *string*. Untuk penerimaan data berikut adalah potongan program yang digunakan.

```
BufferedReader input = new
BufferedReader(new
InputStreamReader(soket.getInputStr
eam()));
while(!input.ready());
input.read(content);
```

Untuk penerimaan data pada sisi robot, menggunakan objek `BufferedReader`. Objek ini memerlukan `reader` sebagai sumber penerimaan data, maka tidak dapat digunakan secara langsung metode `getInputStream` milik *socket*. Oleh karena itu, digunakanlah objek `InputStreamReader` sebagai perantara agar sumber dapat berasal dari InputStream milik *socket*. Keuntungan menggunakan BufferedReader ini yaitu bisa dideteksi apakah sudah terdapat data yang siap diambil atau belum. Setelah dideteksi ada data, maka data tersebut diambil dengan perintah `input.read(content)`. Dengan metode penerimaan ini, data yang diterima berupa *array of char.*

Telepon genggam memberikan instruksi kepada robot untuk bergerak dengan perintah mv ketika terjadi perubahan transmisi kecepatan yang dihitung di bagian class mulai. Telepon genggam memberikan perintah berhenti melakukan koneksi kepada robot omni dengan mengirimkan perintah exit ketika class dari telepon genggam yang melakukan koneksi dengan robot ditutup.

Komunikasi Pengiriman Gambar Dari Robot Omni



Gambar 8. Blok Diagram Penerimaan dan Pengiriman Gambar

Pengiriman gambar pada gambar 8 jelas menggunakan *socket* yang sudah terhubung antara robot omni dengan telepon genggam. Gambar yang sudah diterima tersebut, diubah menjadi objek *ByteArrayOutputStream* dengan tipe gambar PNG dengan menggunakan objek *ImageIO*. Setelah dirubah ke *ByteArray*, yang pertama kali dikirmkan oleh robot omni adalah panjang dari *array* yang berisi gambar tersebut. Hal ini diperlukan agar telepon genggam membuat *array of byte* dengan ukuran yang sesuai. Berikutnya adalah pengiriman data *byte* yang berisi gambar dikirimkan.

Pengiriman dilakukan menggunakan metode yang berbeda dengan pengiriman data. Berikut adalah potongan program untuk mengirimkan data.

```
DataOutputStream a = new
DataOutputStream(soket.getOutputStr
eam());
ByteArrayOutputStream simpan = new
ByteArrayOutputStream();
a.writeInt(integer);
a.write(byte_array);
```

Objek yang digunakan untuk pengiriman data tetap sama, yaitu menggunakan objek `DataOutputStream`. Perbedaannya yaitu metode yang dipakai untuk mengirimkan data. Terdapat dua metode untuk mengirimkan data pada class ini, yaitu metode `a.writeInt(integer)` dan `a.write(byte_array)`. `a.writeInt(integer)` berguna untuk mengirimkan data dalam format *integer*, sedangkan `a.write(byte_array)` mengirimkan data dalam bentuk *array of byte*.

Pada telepon genggam, yang pertama diterima adalah panjang dari *byte* yang harus dibuat. Setelah itu, gambar diterima dan disimpan ke *array byte* dengan ukuran yang sesuai. Setelah data gambar diterima, telepon genggam mengubahnya menjadi *bitmap*. Setelah itu, gambar ditampilkan di layar telepon genggam.

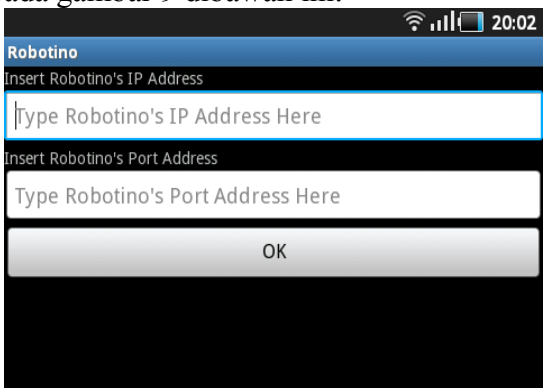
Penerimaan data pada telepon genggam ini menggunakan metode yang berbeda dari penerimaan data pada robot omni. Berikut adalah potongan program untuk penerimaan data yang digunakan.

```
DataInputStream input = new
DataInputStream(soket.getInputStrea
m());
int length = input.readInt();
input.readFully(byte_array);
```

Objek penerimaan data pada penerimaan gambar ini menggunakan DataInputStream. Hal ini dikarenakan `BufferedReader` hanya dapat menerima data berupa *arrray of char* dan *string*, sedangkan yang diperlukan yaitu menerima data dalam bentuk *integer* dan *array of byte*. Untuk membaca data *integer* yang sudah dikirim yaitu menggunakan metode perintah `readInt`, sedangkan untuk membaca data *array of byte* yang sudah dikirim robot menggunakan perintah `readFully(byte_array)`. Penggunaaan objek ini tidak mendukung pendeteksian adanya data atau tidak, tetapi otomatis berhenti menunggu data pada metode ketika dipanggil.

HASIL DAN PEMBAHASAN

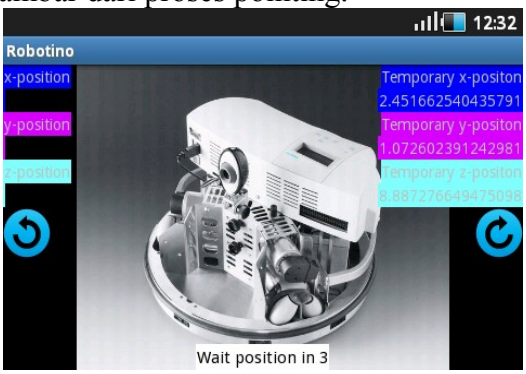
Hasil ketika aplikasi pada telepon genggam dijalankan dapat dilihat seperti pada gambar 9 dibawah ini.



Gambar 9. Halaman pertama dari Aplikasi

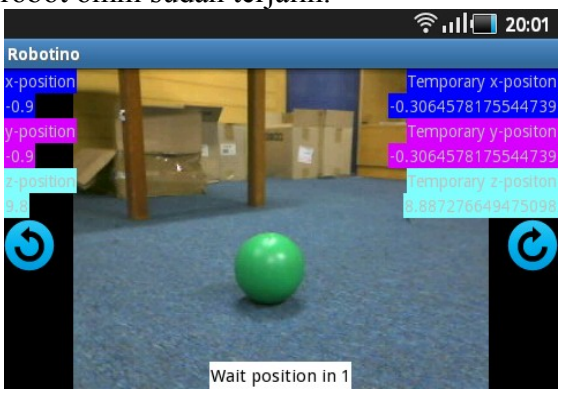
Gambar 9 di atas merupakan form awal yang tampil dari aplikasi yang meminta pengguna untuk memasukka IP dan *port* dari robot omni.

Setelah *user* memasukkan IP dan *port* dari robot omni, maka aplikasi akan memasuki form berikutnya. Hal pertama yang dilakukan adalah *pointing* yang bertujuan agar posisi telepon genggam sejajar dengan daratan. Berikut adalah gambar dari proses pointing.



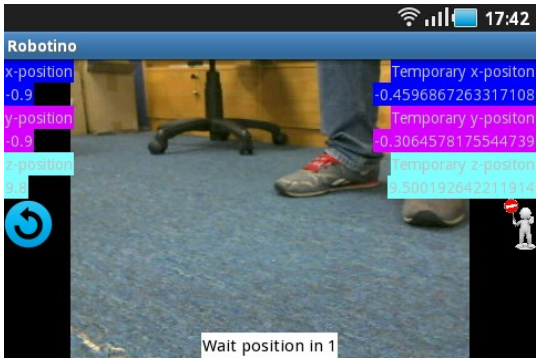
Gambar 10. Keadaan Ketika Proses Pointing

Setelah proses *pointing*, dilanjutkan dengan koneksi ke robot yang dilakukan oleh aplikasi dan mulai menjalankan program utama yaitu proses pertukaran data. Berikut adalah gambar ketika komunikasi antara telepon genggam dan robot omni sudah terjalin.

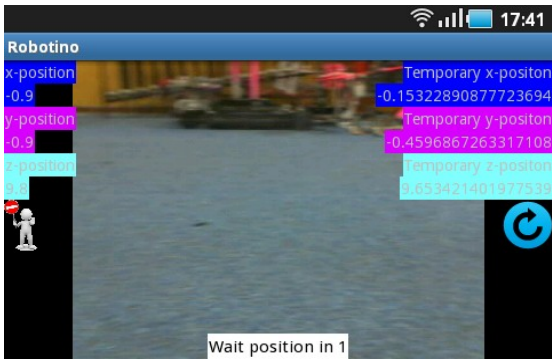


Gambar 11. Keadaan Ketika Program Sedang Berjalan

Pada aplikasi di bagian telepon genggam terdapat tombol untuk memutar robot ke kiri dan ke kanan. Berikut adalah kedua gambar ketika robot diberi perintah oleh pengguna untuk berputar ke kiri dan ke kanan .



Gambar 12. Keadaan Ketika Robot Berputar Ke Kiri



Gambar 13. Keadaan Ketika Robot Berputar Ke Kanan

Dari Percobaan yang dilakukan, sensor dari telepon genggam ini tidak presisi, karena tidak bisa memiliki nilai yang sama meskipun diletakkan pada posisi yang sama. Oleh karena itu, perubahan kecepatan dari robot omni dilakukan dengan memberikan batasan nilai. Jika perubahan axis melebihi batas tersebut, maka kecepatan robot akan berubah dan batas yang sudah ditentukan tersebut dinaikkan lagi, sehingga ketika perubahan axis melebihi batas tersebut, kecepatan robot akan bertambah.

Telepon genggam dan robotino dapat berkomunikasi dengan baik. Perubahan axis dari telepon genggam dapat mempengaruhi pergerakan dari robotino dengan adanya kendala seperti yang sudah dijelaskan sebelumnya. Robotino juga dapat berputar ketika tombol rotasi pada aplikasi telepon genggam ditekan. Telepon genggam dapat menerima gambar yang ditangkap dari kamera robotino dengan adanya kendala. Kendala ini yaitu pemrosesan yang lama dari telepon genggam untuk menampilkan gambar yang diterima. Disimpulkan demikian karena ketika menggunakan *client* dari PC seperti pada pengujian sebelumnya, proses penerimaan gambar lancar tanpa kendala.

DAFTAR PUSTAKA

Android. 2011. *Sensor Event*. (online). (http://developer.android.com/reference/android/hardware/SensorEvent.html, diakses tanggal 26 Agustus 2011).

Anwar, Gunadi and P., Joshua R.T. and P., Yudha W. and Permana, Wim and Prasetyo and Rusdi, Faris and Zuliardi, Agus. 2006. *Jaringan Client Server.* (online). (http://www.google.co.id/url?sa=t&source=web&cd=2&sqi=2&ved=0CBwQFjAB&url=http%3A%2F%2Fwww.wimpermana.web.ugm.ac.id%2Fbudi_s%2Fwp-content%2Fclient_server.pdf&rct=j&q=client%20server%20adalah&ei=XxMITufCG4OurAeHvZ2aDA&usg=AFQjCNFS2SOnPLkATcrU8CcrZcucDLlQmw&cad=rja, diakses 27 Juni 2011).

Braunl, Thomas. 2006. *Embedded Robotics*. Germany: Springer.

Mulyadi. 2010. *Membuat Aplikasi Untuk Android*. Yogyakarta: Multimedia Center Publishing.

Prawita, Ida Bagus Dony. 2010. *Jenis-Jenis Sistem Operasi*. (online). (http://www.docstoc.com/docs/62675496/JENIS—JENIS-SISTEM-OPERASI, diakses tanggal 26 Agustus 2011).