Data Rating untuk Digital Signage di Universitas Kristen Petra menggunakan Kinect

Ivan Wijaya[1], Liliana[2], Rolly Intan[3]
Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121-131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) – 8417658
Telephone number, incl. country code
E-mail: m26410045@john.petra.ac.id[1], rintan@petra.ac.id[2], lilian@petra.ac.id[3]

ABSTRAK

Seiring dengan berkembangannya teknologi, penggunakan digital signage sebagai media komunikasi sudah menjadi hal yang umum didunia digital ini. Dan sekarang Universitas Kristen Petra mulai menggunakan teknologi *digital signage* yang bernama DIVo. Tetapi untuk mengukur keefektifitas dari *digital signage* tersebut dibutuhkan suatu orang untuk menghitung berapa orang yang melewati DIVo, belum lagi masih perlu menghitung berapa orang yang membaca, dan tidak lupa untuk menghitung berapa lama orang tersebut membaca DIVo. Tetapi untuk menghitung efektifitas tersebut apabila menggunakan metode tersebut akan mendapatkan error yang sangat tinggi. Diawali dari tingkat *human error* yang bisa melakukan kesalahan dalam menghitung orang yang melintasi DIVo, belum lagi diperlukan tenaga kerja yang sangat banyak mengingat jumlah DIVo yang lebih dari 1 (satu).
Oleh karena itu dibutuhkan suatu alat yang bisa mendeteksi adanya orang lewat di depan DIVo ini tersendiri, dan bisa mendeteksi wajah dari pembaca, dan alat yang bisa menjalankankan beberapa fitur seperti diatas adalah Kinect. Kinect merupakan alat yang diciptakan oleh Microsoft dengan maksud sebagai alat input untuk gaming console bernama Xbox. Dengan fungsi Skeletal Tracking dan Face Tracking yang tersedia dalam SDK Kinect, aplikasi ini akan mendeteksi orang yang melewati dan mendeteksi berapa orang yang membaca DIVo secara bersamaan. Dan dengan menggunakan alat ini akan mengatasi human error yang terjadi apabila metode penghitungan traffic dengan pembaca secara manual.

Kata Kunci: Kinect, *Skeletal Tracking*, *Face Tracking*, Penghitungan Traffic, Penghitungan Pembaca

ABSTRACT: Along with the growth of Technology, the usage of digital signage for a communication device is common in this digital world. And now Petra Christian University starting using digital signage that named DIVo. But for measuring this device are effective or not we need someone to count every person who pass by this device, and we need to count how many person who many readers who read DIVo as well. But for getting the data manually there will be many error. For example there will be human error that can make a mistake for counting how many person that passing by DIVo twice or even more, and we need many human resource to do this work because there are more than one DIVo in Petra Christian University.
So we need some device that can detect people who passing by DIVo, and can detect thre reader face, and all of that thing can be done by using Kinect. Kinect is a device that made by Microsoft for gaming console named Xbox, this device is a input unit for Xbox so player doesn't need a controlling device at all just using their body for playing. And using skeletal tracking and Face tracking inside Kinect SDK, this application can count how many person that passing by DIVo, and count how many person that read the DIvo and count how long the reader read DIVo. And using this device we can overcome this problem if we counting traffic using manual method.

Keywords: *Kinect, Skeletal Tracking, Face Traking, Traffic Counting, Reader Counting*

1. LATAR BELAKANG

Di jaman yang maju seperti sekarang, penyampaian informasi-informasi sekarang sudah mengalami perubahan bentuk menjadi bentuk *digital*, dikarenakan penyampaian informasi yang ada akan lebih menarik dan lebih hidup. Di dalam perkuliahan, tersimpan banyak sekali informasi, dan dengan menggunakan DIVo Petra ingin memaksimalkan penyampaian informasi ke mahasiswa. Dan dengan DIVo ini sendiri akan banyak membantu mahasiswa untuk kedepannya karena dengan adanya DIVo ini sendiri dapat meminimalisir penggunakan kertas maupun biaya publikasi karena dapat mengurangi jumlah poster yang ada.
Dengan harapan dapat meningkatkan komunikasi mahasiswa mengenai informasi–informasi yang ada di dalam kampus, DIVo pun perlu diukur keefektifannya. Mulai dari berapa orang yang akan melewati DIVo, agar mengetauhi berapa orang yang melintasi DIVo yang akan menjadi calon pembaca. dan kemudian berapa orang yang melihat dari *content* tayangan dari DIVo, dan berapa lama waktu yang dihabiskan untuk membaca tayangan dari DIVo ini.
Untuk mendapatkan data–data ini sendiri, kita menggukan alat bantu Kinect dimana didalam Kinect memiliki sensor yang dapat dimanfaatkan untuk membantu sebagai alat input mendeteksi adanya orang yang melintasi di depan DIVo, mendeteksi berapa banyak orang yang sedang membaca content yang ditampilkan oleh DIVo, dan juga untuk menghitung berapa lama orang membaca content yang di tampilkan oleh DIVo
Pembuatan applikasi ini menjadi penting sebagai sebuah langkah awal untuk mengaplikasikan konsep pengukuran baik itu memadukan konvensional dan teknologi untuk mencapai tujuan kampus digital. Dimana, penyampaian pesan dari Universitas bisa dilakukan melalui DIVo dan juga mahasiswa bisa mengakses DIVo di berbagai lokasi strategis di UK Petra.

2. KINECT

Kinect merupakan alat input yang dikembangkan oleh Microsoft yang dikembangkan dalam projek Natal membuat alat input yang terdiri dari beberapa sensor yang disusun menjadi suatu alat tersendiri. Sensor-sensor yang digunakan di dalam Kinect

merupakan mikrofon, infrared emitor, penerima infrared, dan kamera RGB. Alat *input* ini berfungsi menggunakan kabel *PC Standart* yaitu USB 2.0 tetapi masih membutuhkan tambahan daya power dikarenakan *power* dari *USB port* tidak bisa memenuhi kebutuhan listrik yang dibutuhkan oleh Kinect secara langsung. Dan dikarenakan tidak adanya CPU di dalam *sensor* Kinect, dan hanya madanya *Digital Signal Processor* (DSP) dimana DSP digunakan untuk memproses signal dari sensor-sensor yang ada, kemudian data *prosessing* dari Kinect akan dijalankan di dalam PC dengan menggunakan driver dari Kinect itu sendiri.[1] *Driver* yang disediakan dapat digunakan di Windows 7 atau Windows 8 yang berjalan di 32 – bit atau 64-bit prosesor. Dan minimal menggunakan 2 GB RAM dan dual-core 2.66GHz prosessor untuk minimal spesifikasi dari PC yang kita gunakan. Dan untuk sampai sekarang driver yang digunakan oleh Kinect itu sendiri masih belum bisa digunakan untuk mesin *virtual*.[3].

Microsoft juga menyediakan beberapa SDK yang berfungsi untuk menyeleksi pengguna yang berada di depan Kinect. Dan beberapa SDK yang digunakan untuk mendukung penelitian ini adalah Skeleteal tracking dan Face Detection. Dimana Skeletal tracking adalah suatu algoritma dimana Kinect dapat mengenali manusia yang berada di depan Kinect dengan menggunakan *Infrared* kamera yang terpasang didalam Kinect. Dan Kinect itu sendiri dapat mendeteksi 6 orang atau kurang dalam sekali ambil gambar. Sudut penglihatan Kinect sendiri ditentukan dari setting dari Infrared kamera itu sendiri, dimana dapat diatur dari DepthRange Enumeration, pada *mode default Kinect* dapat melihat orang berdiri antara 0,8 meter sampai 4,0 meter (untuk *Kinect for Xbox*, tetapi untuk *Kinect for PC* memiliki *near mode* dimana dari *near mode* ini bisa mendeteksi dari 0,5 meter hingga 3,0 meter) dan lebar dari 1,2 meter hingga 3,5 meter. Kinect ini sendiri dan mengikuti pergerakannya dari awal hingga akhir. Hingga sekarang *skeletal tracking* ini sendiri dapat di optimasikan untuk user yang berdiri maupun duduk di depan Kinect itu sendiri. Untuk membaca user perlu berada didepan sensor, dan memastikan sensor dapat medeteksi kepala dan setengah badan dari user tersebut.[2]

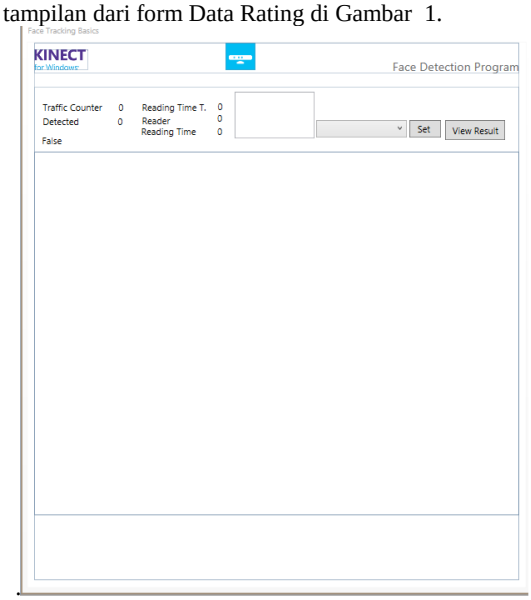
Dan dengan menggunakan fungsi SDK yang sudah disediakan kita dapat menghitung untuk berapa banyak skeleton yang terdeteksi oleh Kinect dengan menggunakan *GetTotalSkeleton*(e). Dengan fungsi tersebut kita bisa menghitung berapa orang yang terdeteksi pada saat itu. Dengan melihat perubahan yang terjadi pada *GetTotalSkeleton*(e), kita bisa memantau untuk penambahan.[4]. Dan dengan kemampuan tersebut dikiranya dapat membantu untuk penelitian ini dimana dapat mendeteksi beberapa orang yang melintasi DIVo.

Kemudian SDK yang digunakan selain *Skeletal Tracking* adalah *Face Detection*. SDK tersebut memperbolehkan user untuk membuat applikasi yang ada mendeteksi wajah manusia secara real time. Algoritma yang digunakan dalam Kinect ini sendiri dibagi 2 tahap besar. Pertama, Kinect akan mengambil 9 bagian khusus dari wajah yaitu hidung, mulut, mata, dan seterusnya. Kemudian gambar akan diolah untuk menghilangkan varisasi pencahayaan dan kemudian mengambil hasil-hasil olahan tersebut. Dan kemudian sistem akan menentukan ekpresi yang ada. Dan alat pengenalan wajah ini juga menentukan dimana wajah muncul dalam sudut padang tertentu dan menormalkan ukuran wajah untuk mengkopensasi apakan user berada derange baca Kinect atau tidak. Dalam kondisi yang paling baik pun Kinect dapat mencapai tingkat keberhasilan hingga 85 persen.[5]

Dan dengan SDK face detection dikiranya dapat membantu untuk penelitian tersebut untuk mendeteksi orang yang sedang membaca DIVo.

3. DESAIN SISTEM

Penelitian ini merupajan Program Data Rating dimana setelah masuk ke Data Rating, maka Program akan membuka suatu form dimana form tersebut akan menyalakan fungsi mengambil gambar yang diterima oleh Kinect sebagai alat input. Kemudian pada saat yang sama akan menjalankan fungsi *skeleton detection*, *face detection* dan *reading time* yang berjalan secara berututan seperti di. Setelah *input video* jalan, fungsi *skeleton detection* akan menseleksi ada berapa orang yang berada di depan Kinect pada saat itu, setelah itu *program* akan menghitung berapa orang yang melewati Kinect itu sendiri dengan cara menambahkan setiap orang yang melintasi atau berada di depan Kinect, dan jika ada penambahan orang pada Kinect maka *counter* orang akan bertambah. Dan jika ada perubahan angka berupa pengurangan orang maka jumlah counter akan tetap. Sewaktu ada orang yang terdeteksi di depan Kinect maka fungsi *face detection* akan berjalan, fungsi *face detection* akan mencari orang yang berada di depan Kinect dan menoleh kearah Kinect, kemudian dari wajah yang terdeteksi tersebut kemudian *counter reader* akan bertambah 1 dan reading timer akan jalan pada saat yang sama. Apabila ada 2 atau lebih wajah yang terdeteksi maka *counter* yang berjalan akan berjalan beberapa kali lebih cepat sesuai dengan jumlah wajah yang terdeteksi di depan Kinect. Berikut adalah tampilan dari form Data Rating di Gambar 1.



Gambar 1 Form Data Rating

Kemudian tiap jamnya hasil dari yang sudah akan disimpan pada temp untuk menyimpan hasil yang sudah diperoleh selama ini,. Kemudian dari semua hasil yang diterima tiap jamnya akan di simpan secara langsung ke dalam file .txt sesuai dengan tanggal dimana aplikasi itu berjalan. Dan apabila aplikasi ini dimatikan, maka data yang sudah tersedia akan ditambahkan ke dalam file simpanan beserta durasi berapa lama aplikasi itu berjalan.

Setelah itu dari data yang telah diperoleh tersebut, kita mendapatkan *data traffic*, *data reader*, dan berapa lama *reader*

menghabiskan waktu untuk membaca DIVo. Dan dari ketiga data tersebut *data rating* untuk DIVo bisa dicari dengan memasukan dalam formula Data Rating untuk digital signage.
Formula yang digunakan untuk mendapatkan data rating untuk *digital signage* yaitu dengan menggunakan data *Traffic* yaitu berapa orang yang melewati DIVo dikalikan dengan berapa orang yang sudah membaca DIVo dikalikan dengan berapa waktu yang telah dihabiskan untuk membaca dari tiap readernya dibagi dengan waktu rotasi yang dihabiskan dari perpindahan poster satu ke poster selanjutnya. Dan hasil dari proses diatas mendapatkan

4. Implementasi Sistem

Berikut adalah beberapa fungsi yang penting untuk mendeteksi dan menggambar *skeletal* pada program ini.
Fungsi *GetTotalSkeleton* merupakan fungsi dimana mengecek berapa orang yang terdeteksi oleh kinect

```
private int
GetTotalSkeleton(SkeletonFrameReadyEventArgs
e)
{
using (SkeletonFrame skeletonFrameData =
e.OpenSkeletonFrame())
{
if (skeletonFrameData == null)
{
return 0;
}

skeletonFrameData.CopySkeletonDataTo(allSkele
tons);
//People not tracked
 int noTrac = (from s in allSkeletons
where s.TrackingState ==
SkeletonTrackingState.PositionOnly
select s).Count();

//People tracked
int comEsq = (from s in allSkeletons
 where s.TrackingState ==
SkeletonTrackingState.Tracked
select s).Count();
 int numberOfSkeletons = noTrac + comEsq;
return numberOfSkeletons;
}
}
```

Fungsi SensorSkeletonFrameReady merupakan fungsi yang berjalan pada waktu aplikasi berjalan. Dan di dalam fungsi ini tersedia semua perintah yang menghitung dan mendeteksi *skeleton* yang ada.

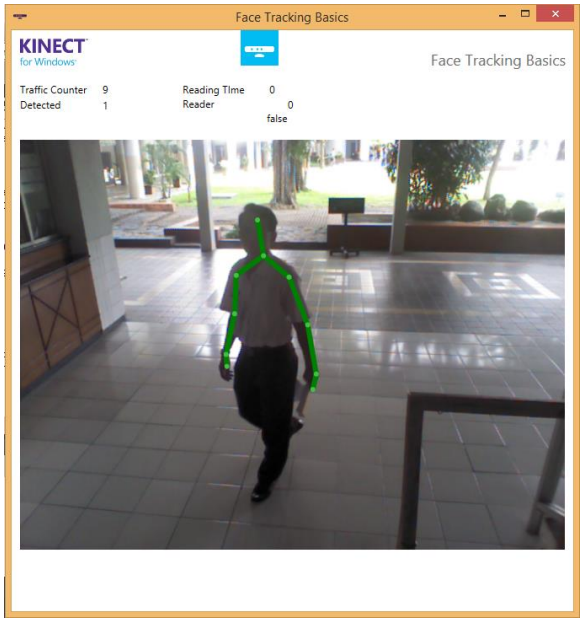
```
private void SensorSkeletonFrameReady(object
sender, SkeletonFrameReadyEventArgs e)
{
TriggerTimer.Start();
if (TriggerTime >= 3600)
{
string tanggal = (DateTime.Now.ToString("d-M-
y")) + ".txt";
using (StreamWriter writer =
File.AppendText(tanggal))
{
writer.WriteLine("Time : " +
DateTime.Now.ToString());
writer.WriteLine("Duration : " +
(TriggerTime/60) + "Menit");
writer.WriteLine("Traffic Counter : " +
Traffic_Counter);
writer.WriteLine("Reader Counter : " +
Count_reader);
writer.WriteLine("Reading Time : " +
Count_Time_Read);
writer.WriteLine("");
writer.WriteLine("==========================
==============================");
Traffic_Counter = 0;
Count_reader = 0;
Count_Time_Read = 0;
TriggerTime = 0;
}
}
if (Reading == true)
{dispatcherTimer.Start();}
    else
{dispatcherTimer.Stop();}
if
(faceTrackingViewer.GetFaceTrackingIsEnabled(
) && Reading == false)
{
Reading = true;
Count_reader++;
label5.Content = "true";
}
else if
(faceTrackingViewer.GetFaceTrackingIsEnabled(
) && Reading == true)
{
Reading = true;
label5.Content = "true";
}
else
{
Reading = false;
label5.Content = "false";
}
AllSkeleton_tracked_detected =
GetTotalSkeleton(e);
Count_Detected = Getnotrac(e);
Count_Tracked = GetTrack(e);
Skeleton[] skeletons = new Skeleton[0];
if (Count_Detected > Temp_Detect && Change ==
false)
{
Traffic_Counter++;
Temp_Detect = Count_Detected;
Change = true;
}
if ((Count_Tracked + Count_Detected)==
Temp_Detect)
{
Change=false;
Temp_Detect = Count_Detected;
}
label1.Content = Traffic_Counter;
label2.Content =
AllSkeleton_tracked_detected;
label3.Content = Count_Time_Read;
label4.Content = Count_reader;
if
(faceTrackingViewer.GetFaceTrackingIsEnabled(
))
label5.Content = "true";
else
label5.Content = "false";
using (SkeletonFrame skeletonFrame =
e.OpenSkeletonFrame())
{
if (skeletonFrame != null)
{
```

```
skeletons = new
Skeleton[skeletonFrame.SkeletonArrayLength];
skeletonFrame.CopySkeletonDataTo(skeletons);
}
}
using (DrawingContext dc =
this.drawingGroup.Open())
{
// Draw a transparent background to set the
render size
dc.DrawRectangle(Brushes.Transparent, null,
new Rect(0.0, 0.0, RenderWidth,
RenderHeight));

if (skeletons.Length != 0)
{
foreach (Skeleton skel in skeletons)
{
RenderClippedEdges(skel, dc);
if (skel.TrackingState ==
SkeletonTrackingState.Tracked)
{
this.DrawBonesAndJoints(skel, dc);
}
else if (skel.TrackingState ==
SkeletonTrackingState.PositionOnly)
{
dc.DrawEllipse(
this.centerPointBrush,
null,
this.SkeletonPointToScreen(skel.Position),
BodyCenterThickness,
BodyCenterThickness);
}
}
}
// prevent drawing outside of our render area
this.drawingGroup.ClipGeometry = new
RectangleGeometry(new Rect(0.0, 0.0,
RenderWidth, RenderHeight));
}
}
```

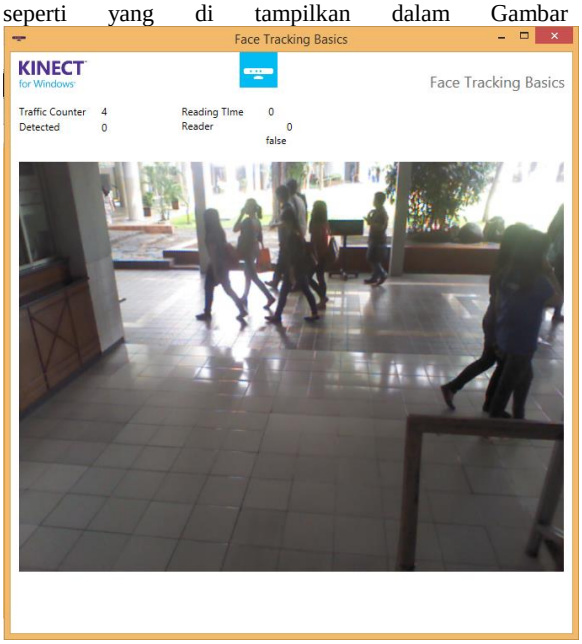
5. PENGUJIAN

Pada proses pengujian, data yang didapat dari berapa orang yang melewati DIVo akan dihitung dengan *kinect* seperti yang ditampilkan pada Gambar 3.



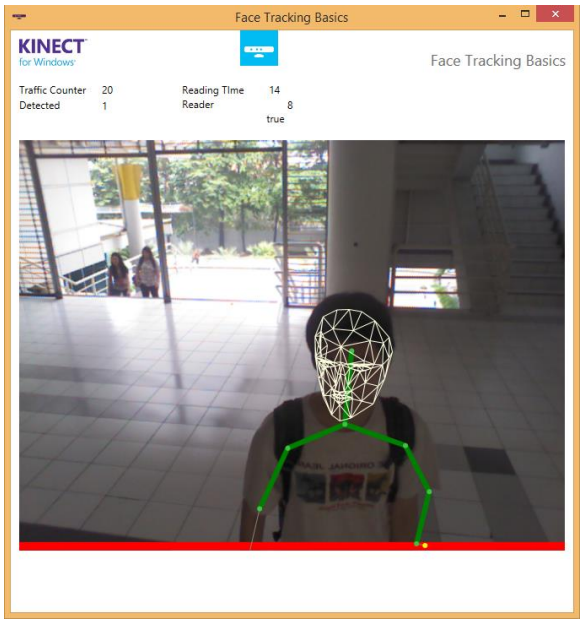
Gambar 2 Contoh Kinect mendeteksi orang yang melewati DIVo

Tetapi ada juga beberapa kendala dimana kinect tidak dapat mendeteksi orang yang melintasi DIVo dikarenakan adanya kendala dimana jarak orang yang melewati DIVo itu terlalu jauh, seperti yang di tampilkan dalam Gambar 4.



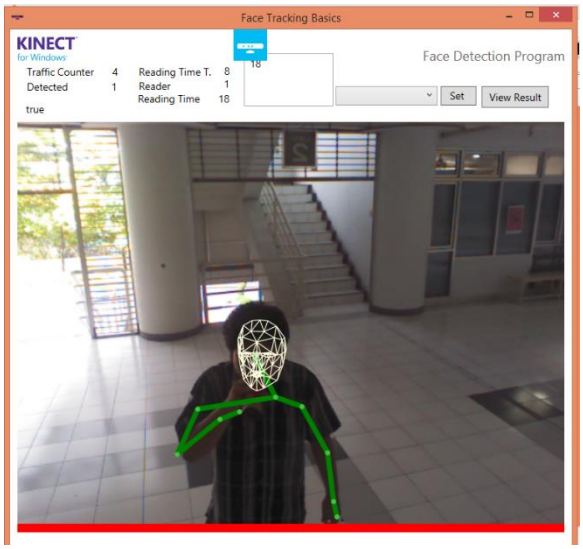
Gambar 3 Contoh Kinect tidak mendeteksi Skeletal

Kemudian setelah berhasil mendapatkan skeletal yang melewati depan DIVo. Program akan menjalankan fungsi dimana akan membedakan orang yang membaca atau melintasi DIVo, dengan cara mendeteksi wajah yang sedang membaca DIVo tersebut. Dan pada Gambar 5 merupakan salah satu contoh dimana program berhasil mengambil wajah pembaca yang sedang membaca DIVo.



Gambar 4 Contoh Kinect Mendeteksi Wajah Orang yang sedang membaca DIVo

Tetapi dari ada juga kendala dimana Kinect terganggu dengan adanya objek yang berada di depan wajah pembaca seperti kacamata, botol minum dan lain-lain. Kendala yang terjadi berupa kecepatan menangkap wajah pembaca tidak secepat seperti orang yang tidak menggunakan kacamata atau orang yang tidak meminum pada saat membaca DIVo seperti yang di Gambar 6.



Gambar 5 Contoh Kinect Mendeteksi Wajah yang membaca DIVo tetapi Orang yang membaca itu sedang minum

6. KESIMPULAN

Kesimpulan yang dapat diambil dari pembuatan Tugas Akhir ini adalah:

- Sensor Kinect dapat digunakan untuk mengambil data *Traffic* dari orang yang melintasi depan Kinect dengan menggunakan fungsi skeletal traking
- Sensor Kinect dapat digunakan untuk mengambil wajah dari pembaca sehingga jumlah dari pembaca bisa diambil
- Sensor Kinect memiliki batasan tersendiri untuk jarak baca mendeteksi orang yang melewati di depan sensor Kinect
- Sensor Kinect memiliki kekurangan dimana sensor tersebut tidak terlalu cepat untuk mendeteksi wajah pembaca apabila adanya aksesoris seperti kaca mata atau tertutup rambut
- Sensor Kinect memiliki kekurangan dimana sensor tersebut tidak dapat menangkap wajah dengan jelas apabila adanya cahaya yang kuat dari belakang
- Sensor Kinect memiliki kekurangan dimana sensor tersebut tidak dapat menangkap *skeletal* yang berdempetan

7. REFERENSI

[1] Cathuhe, David (2012). *Programming with the Kinect for Windows Software Development Kit.* United State of America

[2] Fitzpatrick, Matthew dan Matthiopoulos, Nicolaus (2013). Real *Time Person Tracking and Identification using the Kinect Sensor.* United State of America

[3] Kar, Abhishek (2013). *Skeletal Tracking Using Microsoft Kinect.* Kanpur

[4] Webb, Jarrett. dan Ashley, James (2012). *Beginning Kinect Programming with the Microsoft Kinect SDK* . New York

[5] Wyremblelski, Adam (2014). *Detection of the Selected, Basic Emotion Based on Face Expression Using Kinect.* Moszczak