**BAB I**

**PENDAHULUAN**

1. **Latar Belakang**

Dewasa ini dunia animasi semakin berkembang, baik dalam pembuatan game, film, edukasi, maupun periklanan. Pada awal mulanya animasi–animasi yang ada tersebut masih berupa gambar dua dimensi, tetapi dengan semakin berkembangnya teknologi, pembuatan animasi sudah beralih ke ranah 3 dimensi. Untuk memudahkan pembuatan animasi tersebut maka sudah banyak dikembangkan aplikasi – aplikasi dengan menggunakan komputer sebagai media utama.Untuk memudahkan merepresentasikan suatu objek 3 dimensi yang sudah dibuat ke dalam dunia nyata, dapat dilakukan pelacakan suatu tanda ( *marker* ) khusus. Selain itu, kemampuan untuk melacak suatu tanda tertentu dalam suatu kumpulan objek memberikan keuntungan yang cukup besar terhadap interaksi manusia dan komputer. Untuk pelacakan tersebut dapat digunakan media kamera sebagai input. Beberapa aplikasi *virtual reality* didesain untuk mengkombinasikan representasi virtual dengan persepsi dari dunia nyata. Representasi virtual memberikan informasi tambahan yang mana tidak didapatkan oleh user yang tidak memakai alat bantu. *Augmented reality* dapat digunakan untuk membantu memberikan gambaran atau simulasi kepada *user*. Misalnya untuk visualisasi pengenalan *concept car* atau visualisasi buku cerita 3 dimensi. Oleh karena itu sebagai tahap pertama dalam pembuatan *Augmented Reality*, dibuatlah aplikasi untuk mendeteksi papan acuan yang berupa tanda (*marker*) khusus dalam suatu gambar tertentu.

1. **Perumusan Masalah**
2. Bagaimana mendeteksi apakah suatu bagian *image* merupakan tanda
3. Bagaimana menentukan posisi papan acuan di dunia nyata pada dunia maya
4. Bagaimana cara mengenali apakah terjadi perubahan posisi ( translasi atau rotasi ) papan acuan.
5. Bagaimana merepresentasikan perubahan posisi tersebut terhadap objek yang akan diaplikasikan kedalam lingkungan virtual ( maya )
6. Bagaimana cara menerapkan teknologi *Augmented Reality*  sebagai fortofolio digital ?
7. **Tujuan**

Dengan adanya teknologi  *Augmented Reality* ini , diharapkan dapat menjadi inovasi terbaru terhadap fortofolio yang ada sehingga menjadi lebih menarik dapat dimanfaatkan oleh banyak orang.

1. **Ruang Lingkup**

Pengambilan gambar bergerak akan dilakukan melalui webcam dengan pencahayaan yang cukup.Penggunaan papan acuan berupa kertas yang datar. Pada papan acuan terdapat tanda (*marker*) berupa kotak yang diberikan kode atau identitas khusus.Bentuk *marker* minimal sebagai berikut :



Langkah yang digunakan dalam proses image preprocessing adalah camera calibration untuk memperbaiki citra kamera dari distorsi lena kamera. Kemudian dilakukan proses pencarian koordinat dari papan acuan. Pendeteksian hanya dibatasi rotasi dan translasi yang dilakukan pada meja datar. Pendeteksian dilakukan hanya 2 dimensi yaitu sumbu-x ( lebar ) dan sumbu-z ( kedalaman ) tanpa mendeteksi perpindahan sumbu-y ( tinggi ). Output yang ditampilkan *non real-time,* tidak ditampilkan secara langsung. Antara input dan output dapat delay.

1. Objek ditangkap kamera ( webcam ) dan dirubah ke bentuk simulasi 3D di layar monitor dengan background, lalu bergerak sesuai pengenalan objek.
2. Pembuatan aplikasi menggunakan *Unity.*
3. **Sistematika Penulisan**

Garis besar penulisan Penelitian ini adalah sebagai berikut:

BAB 1 : Pendahuluan

Berisi latar belakang masalah, perumusan masalah dan ruang lingkup, tujuan Tugas Akhir, dan metodologi penelitian yang dipakai.

BAB 2 : Teori Penunjang

Berisi berbagai teori yang berhubungan dengan definisi augmented reality dan algoritma yang digunakan untuk proses pendeteksian tanda ( *marker* )

BAB 3 : Metodologi Penelitian

Berisi perencanaan pembuatan keseluruhan sistem dalam aplikasi yang akan dibuat.

BAB 4 : Hasil Penelitian Dan Pembahasan

Berisi hasil pembuatan aplikasi dan segmen-segmen program yang ada dalam aplikasi yang dibuat.

BAB 5 : Kesimpulan dan Saran

Berisi kesimpulan dari hasil pengujian perangkat lunak dan beberapa saran untuk pengembangan lebih lanjut.

**BAB II**

**LANDASAN TEORI**

1. **Augmented Reality**

Augmented Reality (AR) merupakan sebuah istilah untuk menggabungkan lingkungan dunia nyata dengan dunia virtual yang dibuat dengan komputer menggunakan bahasa pemrograman tertentu sehingga batas antar keduanya menjadi sangat tipis. Pada tahun 1997 Ronald Azuma mendefinisikan augmented reality sebagai sistem yang memiliki karakter sebagai berikut:

1. Menggabungkan lingkungan nyata dan virtual
2. Berjalan secara interaktif dalam waktu nyata
3. Integrasi dalam tiga dimensi (3D)
4. **Teknik Display Augmented Reality**

Sistem display AR merupakan sistem manipulasi citra yang menggunakan seperangkat optik, elektronik, dan komponen mekanik untuk membentuk citra dalam jalur optik antara mata pengamat dan objek fisik yang akan digabungkan dengan teknik AR. Bergantung kepada optik yang digunakan, citra bisa dibentuk pada sebuah benda datar atau suatu bentuk permukaan yang kompleks (tidak datar).



Gambar 1. Pembentukan citra untuk display augmented realit

Gambar 1 mengilustrasikan kemungkinan citra akan dibentuk untuk mendukung AR, peletakan display bergantung dari pandangan pengguna dan objek, dan tipe citra seperti apa yang akan dihasilkan (planar atau curved).(Ronald T. Azuma, 1997) Secara garis besarnya ada tiga teknik display AR, yaitu sebagai berikut: 1. Head-Attached Display 2. Handheld Display 3. Spatial Display.

1. **Spatial Display**

Dalam Spatial Augmented Reality (SAR), objek nyata digabungkan langsung dengan citra yang terintegrasi langsung ke lingkungan nyata. Contohnya, citra diproyeksikan ke lingkungan nyata menggunakan proyektor digital atau tergabung dengan lingkungan menggunakan panel display. Perbedaan utama pada SAR dibanding teknik display sebelumnya adalah displaynya terpisah dengan pengguna. SAR memiliki kelebihan dari HMD dan handheld, sistem ini bisa digunakan oleh banyak orang pada waktu bersamaan tanpa perlu menggunakan suatu alat.(Ronald T. Azuma, 1997) .

1. Screen- Based Video See-Through Displays.

Screen-Based AR menggabungkan citra dan lingkungan nyata yang di tampilkan ke sebuah monitor, seperti yang ditunjukan pada gambar 2.

1. Spatial Optical See-Through Displays

Sistem ini menghasilkan citra yang ditampilkan langsung ke lingkungan nyata. Komponen yang penting dalam sistem ini meliputi spatial optical combiners (planar atau curved beam combiners), layar transparan atau hologram.

1. Projection-Based Spatial Displays

Sistem ini memproyeksikan citra secara langsung pada permukaan objek fisik daripada menampilkannya pada sebuah bidang pencitraan dalam penglihatan pengguna. Sistem ini menggunakan banyak proyektor yang digunakan untuk meningkatkan wilayah tampilan serta meningkatkan kualitas citra.

1. **Pengertian Citra Digital**

Secara umum, pengolahan citra digital menunjuk pada pemrosesan gambar 2 dimensi menggunakan komputer. Dalam konteks yang lebih luas, pengolahaan citra digital mengacu pada pemrosesan setiap data 2 dimensi.



Gambar 2 Contoh Screen-Based Video See-Through Displays

Citra digital merupakan sebuah larik (array) yang berisi nilai-nilai real maupun komplek yang direpresentasikan dengan deretan bit tertentu. Suatu Citra dapat di definisikan sebagai fungsi f(x,y) berukuran M baris dan N kolom, dengan x dan y adalah koordinat spasial, dan amplitude f di titik koordinat (x,y) dinamakan intensitas atau tingkat keabuan dari citra pada titik tersebut. Apabila nilai x,y, dan amplitude f secara keseluruhan berhingga (finite) dan bernilai diskrit maka dapat dikatakan bahwa citra tersebut adalah citra digital(Darma Putra, 2004). Gambar 3 menunjukan posisi koordinat citra digital. Citra digital dapat ditulis dalam bentuk matrik sebagai berikut (Darma Putra, 2004).

Berdasarkan jenisnya, citra digital dapat dibagi menjadi 3 (Sutoyo, 2009), yaitu: 1. Citra Biner (Monokrom) Memiliki 2 buah warna, yaitu hitam dan putih. Warna hitam bernilai 1 dan warna putih bernilai 0. Untuk menyimpan kedua warna ini dibutuhkan 1 bit di memori. Contoh dari susunan piksel pada citra monokrom adalah sebagai berikut (gambar 4). 2. Citra Grayscale (skala keabuan) Citra grayscale mempunyai kemungkinan warna hitam untuk nilai minimal dan warna putih untuk nilai maksimal. Banyaknya warna tergantung pada jumlah bit yang disediakan di memori untuk menampung kebutuhan warna tersebut. Semakin besar jumlah bit warna yang disediakan di memori, maka semakin halus gradasi warna yang terbentuk (gambar 5).



Gambar 3 Koordinat citra digital

Setiap piksel pada citra warna mewakili warna yang merupakan kombinasi tiga warna dasar, yaitu merah, hijau, dan biru (RGB = Red, Green, Blue). Setiap warna dasar menggunakan penyimpanan 8 bit = 1 byte (nilai maksimum 255 warna), jadi satu piksel pada citra warna diwakili oleh 3 byte.(gambar 6)

1. **Deteksi Tepi (Edge Detection)**

Tepian dari suatu citra mengandung informasi penting dari citra bersangkutan. Tepian citra dapat merepresentasikan objek – objek yang terkandung dalam citra tersebut, bentuk, dan ukurannya serta terkadang juga informasi tentang teksturnya. Tepian citra adalah posisi dimana intensitas pixel dari citra berubah dari nilai rendah ke nilai tinggi atau sebaliknya. Deteksi tepi umumnya adalah langkah awal melakukan segmentasi citra (Darma Putra, 2004). Deteksi tepi (edge detection) adalah operasi yang dijalankan untuk mendeteksi garis tepi (edges) yang membatasi dua wilayah citra homogeny yang memiliki tingkat kecerahan yang berbeda (Pitas 1993). Deteksi tepi pada suatu citra adalah suatu proses yang menghasilkan tepi-tepi dari obyek-obyek citra, tujuannya adalah:

1. Untuk menandai bagian yang menjadi detail citra.
2. Untuk memperbaiki detail dari citra yang kabur, yang terjadi karena error atau adanya efek dari proses akuisisi citra.
3. Serta untuk mengubah citra 2D menjadi bentuk kurva suatu titik (x,y) dikatakan sebagai tepi (edge) dari suatu citra bila titik tersebut mempunyai perbedaan yang tinggi dengan tetangganya. Gambar 7 berikut ini menggambarkan bagaimana tepi suatu gambar di peroleh.

Ada dua metode untuk dapat mendeteksi tepi yaitu (wasista, 2009):

1. Metode First-Order Derivative Edge Detection
2. Metode Second-Order Derivative Edge Detection.
3. **Operator Canny**

Deteksi tepi Canny dapat mendeteksi tepian yang sebenarnya dengan tingkat kesalahan minimum. Dengan kata lain, operator Canny di desain untuk menghasilkan citra tepian yang optimal. Berikut adalah langkah-langkah dalam melakukan deteksi tepi Canny (Darma Putra, 2004).

1. Menghilangkan derau yang ada pada citra dengan mengimplementasikan tapis Gaussian. Proses ini akan menghasilkan citra yang tampak sedikit buram. Hal ini dimaksudkan untuk mendapatkan tepian citra yang sebenarnya. Bila tidak dilakukan maka garis-garis halus juga akan di deteksi sebagai tepian. Berikut ini adalah salah satu contoh tapis Gaussian dengan σ = 1,4.
2. Melakukan deteksi tepi dengan salah satu deteksi tepi turunan pertama dengan melakukan pencarian secara horizontal (Gx) dan vertikal (Gy).
3. Menentukan arah tepian yang ditemukan dengan menggunakan rumus sebagai berikut. Selanjutnya membagi ke dalam 4 warna sehingga garis arah berbeda dan memiliki warna berbeda. Pembagiannya adalah 0 – 22,5 dan 157,5 – 180 derajat berwarna kuning, 22,5 – 67,5 berwarna hijau, dan derajat 67,5 – 157,5 berwarna merah.
4. Memperkecil garis tepi yang muncul dengan menerapkan nonmaximum suppression sehingga menghasilkan garis tepian yang lebih ramping.
5. Langkah terakhir adalah berinisiasi dengan menerapkan dua buah tresholding.
6. **Adobe Flash**

Adobe Flash merupakan sebuah program yang ditujukan kepada para desainer atau programer yang bertujuan merancang animasi untuk pembuatan sebuah halaman web, pembuatan game interaktif, presentasi untuk tujuan bisnis, proses pembelajaran, pembuatan film kartun, dan dapat digunakan untuk membangun sebuah aplikasi yang bernilai tinggi serta tujuan – tujuan lain yang lebih spesifik lagi. Teknologi flash menjadi solusi bagi penyebar informasi atau pembangunan aplikasi untuk disebarkan ke khalayak ramai sehingga menjadi teknologi yang popular dan berkembang dengan pesat. Flash dapat dilihat dari dua aspek, yaitu:

1. Flash sebagai software.

Adobe Flash sebagai software pembuat atau pembangun aplikasi, system informasi, dan pembuat animasi.

1. Flash sebagai teknologi.

Sekarang ini hampir semua browser serta sebagian peralatan elektronik sudah terinstal Flash Player untuk dapat menjalankan animasi. Flash adalah program animasi yang berbasis vector yang dapat menghasilkan file yang berukuran kecil sehingga mudah di akses, Flash dilengkapi dengan tool – tool untuk membuat gambar yang kemudian akan dibuat animasi atau dijalankan dengan script-nya (ActionScript).

**BAB III**

**METODOLOGI PENELITIAN**

1. **Tempat dan Waktu Penelitian**

Dalam pelaksanaan tugas besar ini penulis mengambil tempat penelitian di Lab-Praktikum Teknik Informatika Uin Sunan Gunung Djati Bandung

1. **Bahan dan Peralatan**

Dalam pembuatan tugas besar ini, penulis menggunakan peralatan dan program yang disesuaikan dengan kebutuhan dalam pembuatan aplikasi dengan augmented reality. Secara lebih spesifik, peralatan dan program yang digunakan dirinci sebagai berikut:

1. Spesifikasi Laptop Yang Digunakan Yaitu:
2. Sistem operasi Windows XP 32-bit.
3. Processor Intel Core i3-380
4. RAM DDR2 2 GB
5. Harddisk 320 GB.
6. Webcam eksternal Logitech 720HD
7. Webcam internal 320 DPI
8. Perangkat Lunak.
9. Windows 7, berfungsi sebagai sistem operasi
10. Adobe Flash CS5, berfungsi sebagai IDE
11. Adobe Flash Builder, berfungsi sebagai script editor
12. Flash player 10-11, berfungsi sebagai debugger
13. Mozilla Firefox, berfungsi sebagai web browser
14. **Prosedur Penelitian**

Prosedur yang dilakukan dalam membuat perancangan kartu nama dengan augmented reality adalah sebagai berikut:

1. Melakukan studi literatur dan mencari referensi yang berhubungan dengan augmented reality, actionscript dan flash, dan pengolahan citra digital.
2. Melakukan pengambilan data berupa biodata dan portofolio dari pelukis.
3. Melakukan pembuatan animasi.
4. Melakukan pembuatan deteksi marker.
5. Melakukan perancangan marker.
6. Melakukan pembuatan motion detect dengan actionscript.
7. Merancang interface aplikasi augmented reality.
8. Melakukan pengujian dari aplikasi augmented reality.
9. Mengimplementasikan aplikasi yang telah dibuat.
10. **Perancangan Program**

Perancangan program aplikasi ini dilakukan dengan menerapkan pendeteksian warna RGB (Red-Green-Blue) untuk mendeteksi marker dengan pola warna tersebut yang akan di jadikan sebagai input untuk aplikasi augmented reality. Pada aplikasi augmented reality, objek animasi akan di render jika marker telah terdeteksi lalu akan diterapkan deteksi tepi dengan operasi canny untuk deteksi motion untuk interaksi antara pengguna dan objek animasi. Secara terperinci, perancangan dan aplikasi akan dibagi menjadi beberapa bagian, dengan flowchart seperti pada gambar 8.

1. **Mendeteksi Marker**

Proses ini dimulai dengan mengakses kamera pada komputer (plug-and-play webcam, embedded webcam, dan sebagainya), kemudian melakukan kalibrasi singkat ukuran video dan kualitasnya (fps), dan menangkap inputan berupa gambar live yang terhampar di hadapan kamera.



Pada tahap akses ini, user akan menghadapkan marker pada kamera.

Kamera dan program akan menerima gambar dari video stream kamera. Hasil gambar dari input video stream ini akan dikalkulasikan setiap pixelnya. Kemudian,program akan mencari pola warna pada marker. Setelah menemukan pola warna RGB pada marker. Program akan mengkalkulasi pola warna berdasarkan posisi dari ketiga warna (merah di atas,biru di samping, hijau di samping) tersebut dan membandingkan jumlah pixel dari ketiga warna tersebut.

Marker akan terbaca jika orientasi warna merah berada di atas, jika orientasi tidak sesuai maka marker tidak akan terbaca, jika marker telah terbaca maka proses pendeteksian akan dihentikan untuk meminimalisir penggunaan memori komputer. Berikut adalah logika algoritmanya.

P = position, PTop = top position, PL = left position, PR = right position For each pixel (Px) in the image

If (P = 1)

PT.RED > PL.RED and PR.RED

Else If (P = 0)

PL.GREEN > PT.GREEN and PR.GREEN

PR.BLUE > PT.BLUE and PR.BLUE

Calculate the score of the pixel

End if

End for

1. **Render Animasi**

Proses ini dimulai ketika aplikasi telah mendeteksi marker yaitu aplikasi akan mengambil gambar dari lokasi sumber gambar dan menyimpannya dalam array. Gambar yang telah tersimpan dalam array tersebut lalu akan ditempatkan pada movieclip dengan meng-iterasi jumlah movieclip sesuai dengan jumlah gambar. Jika jumlah movieclip sama dengan jumlah gambar maka iterasi akan di hentikan lalu akan diposisikan dengan bentuk lingkaran 3 dimensi dimana movieclip pertama akan berada paling depan, dan yang terakhir paling belakang. Ketika proses render animasi dipanggil, secara bersamaan pula akan dilakukan proses pendeteksian gerakan.

1. **Mendeteksi Gerakan.**

Setelah marker terdeteksi maka proses ini akan dimulai bersamaan dengan proses render animasi. Tujuan dari proses mendeteksi gerakan adalah untuk sarana interaksi objek animasi dengan user seperti menggerakan animasi ke kiri dan kanan, dan menampilkan detail dari gambar lukisan. Pada proses deteksi gerakan ini program akan mengambil input dari video stream yang di tangkap kamera dan menyimpannya dalam variable dengan tipe bitmapdata, mengkonversinya ke dalam resolusi yang lebih kecil dan tidak akan ditampilkan ke layar komputer, ini dilakukan karena proses terjadi hampir bersamaan secara live dengan tampilan dari video stream dan untuk mengkalkulasi setiap pixel dibutuhkan resource memori makin besar jika ukuran dari video stream makin besar pula. Pertama – tama akan dilakukan deteksi tepi dengan operator canny. Mengapa menggunakan sebenarnya dengan tingkat kesalahan minimum (Darma Putra,2010). Dengan kata lain, operator canny di desain untuk menghasilkan citra tepian yang optimal. Berikut adalah langkah-langkah dalam melakukan deteksi tepi canny yang digunakan dalam program.

1. Menghilangkan derau yang ada pada citra dengan mengimplementasikan tapis Gaussian. Proses ini akan menghasilkan citra yang tampak sedikit buram. Hal ini dimaksudkan untuk mendapatkan tepian citra yang sebenarnya. Bila tidak dilakukan maka garis-garis halus juga akan di deteksi sebagai tepian. Berikut ini adalah tapis Gaussian yang digunakan dengan σ = 1,4 dengan matriks pada rumus (2).
2. Melakukan deteksi tepi dengan operator sobel, yaitu dengan mengkalkulasi dan men-iterasi nilai input pixel dari setiap frame dengan matriks operator sobel.
3. Menentukan arah tepian yang ditemukan dengan menggunakan rumus (2) dan (3).
4. Memperkecil garis tepi yang muncul dengan menerapkan nonmaximum suppression sehingga menghasilkan garis tepian yang lebih ramping.
5. Langkah terakhir adalah menyimpan nilai pixel yang telah dikalkulasikan dengan variable baru. Dan menerapkan tresholding antara nilai input dan nilai hasil kalkulasi.

Untuk mendapatkan motion detect menggunakan metode difference blend mode, yang pada dasarnya adalah membandingkan warna merah, hijau, dan biru dari masing-masing pixel per-frame (frame sebelum, frame sesudah) maka didapatkan nilai selisih dari kedua frame tersebut (blend frame). Untuk mendapatkan warna hitam yang lebih akurat maka frame yang di proses adalah output dari deteksi tepi.

**BAB IV**

**HASIL PENELITIAN DAN PEMBAHASAN**

1. **Pengujian Program**

Aplikasi Pendeteksi Gerak untuk *Augmented reality* merupakan aplikasi perangkat lunak utama dalam Tugas Akhir ini. Tampilan dapat dilihat pada Gambar 4.30. Pada saat aplikasi ini berjalan, prosedur pendeteksian *marker* akan berjalan secara otomatis. Apabila suatu objek sudah teridentifikasi sebagai *marker*, maka pada layar akan muncul suatu objek. Objek yang ditampilkan dapatdiubah-ubah sesuai dengan keinginan user melalui penekanan tombol keyboard.



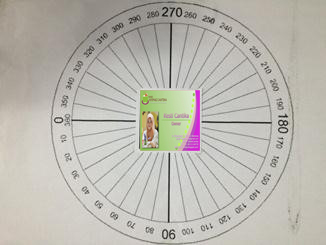
Gambar 4.30. Tampilan aplikasi utama

1. **Pengujian Tingkat Akurasi Rotasi Marker Pada Sumbu Pusat**

Proses pengujian tingkat akurasi rotasi *marker* pada sumbu pusat dilakukan dengan kondisi jarak dari camera ke *marker* 31 cm dengan ketinggian 37 cm. Dengan rumus phytagoras, maka ditemukan teta sudut kamera adalah 48.27o kearah bawah. Untuk menjaga agar *marker* selalu tepat ditengah ketika percobaan ini dilakukan, maka titik pusat *marker* ditancapkan ke titik pusat kertas derajat dengan menggunakan jarum.

Gambar 4.31. Sudut 0o gambar dari webcam (kiri) dengan gambar tampak atas (kanan)





4.2. Tabel pengujian tingkat akurasi rotasi marker pada sumbu pusat

|  |  |  |
| --- | --- | --- |
| **Sudut Real** | **Sudut Estimasi** | **Selisih** |
| 0 o | 0 o | 0 o |
| 45 o | 36.38 o | 8.62 o |
| 60 o | 69.23 o | -9.23 o |
| 90 o | 89.05 o | 0.95 o |
| 145 o | 140.31 o | 4.69 |
| 225 o | 217.63 o | 7.37 |

Dari Tabel 4.2 dapat dilihat bahwa pada sudut real 0o dan 90o, Ketika *marker* selaras dengan axis horizontal maupun vertikal, perhitungan estimasi rotasi yang dihasilkan relative lebih akurat. Namun pada waktu *marker* diputar diagonal, perhitungan rotasi menjadi tidak akurat. Nilai yang dihasilkan memiliki perbedaan maksimal 8.62 o dan minimal -9.23 o.

1. **Pengujian Jarak Optimal Pendeteksian Marker**

Pada sub bab ini dilakukan pengujian dengan variabel jarak dan variabel rotasi. Pengujian dengan variabel jarak yaitu, marker akan di uji coba pada jarak yang berbeda-beda dimulai dari 20 cm dari kamera, 30 cm, kemudian kelipatan 3 cm hingga 3.6 meter. Sedangkan variabel rotasi adalah marker diuji dengan sudut rotasi 0o, 45o ,90o ,135o ,180o.

Tabel 4.3. Tabel hasil pengujian jarak dan rotasi marker

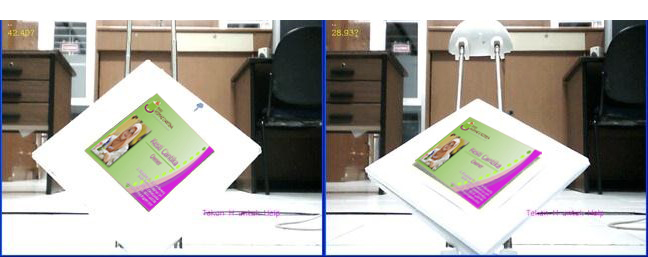
**Jarak (cm)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **rotasi marker** | | | | |
| **0** o | **45** o | **90** o | **135** o | **180** o |
| **20**  **30**  **60**  **90**  **120**  **150**  **180**  **210**  **240**  **270**  **300**  **330**  **360** | 359.26 | 43.64 | 91.45 | 133.42 | 179.68 |
| 359.5 | 42.06 | 91.53 | 133.95 | 180 |
| 357.09 | 42.14 | 88.06 | 132.93 | 177.09 |
| 355.71 | 42.88 | 86.99 | 135 | 175.6 |
| 356.05 | 42.14 | 87.95 | 130.82 | 176.19 |
| 3.01 | 38.66 | 92.86 | 131.42 | 174.56 |
| 356.63 | 39.81 | 96.71 | 133.03 | 183.37 |
| 6.34 | 39.81 | 96.71 | 132.51 | 173.29 |
| 3.81 | 39.81 | 7.59 | 42.27 | 172.41 |
| 4.09 | 45 | 12.09 | 39.29 | 4.09 |
| 4.76 | 45 | 5.19 | 34.99 | 4.76 |
| 5.71 | 36.87 | 5.71 | 37.87 | 5.71 |
| - | - | - | - | - |

Tabel 4.3 merupakan tabel hasil pengujian jarak dan rotasi marker dengan variabel sumbu-x (horizontal) yaitu rotasi marker dalam derajat, variabel sumbu-y (vertical) yaitu jarak marker dari kamera dalam satuan cm. sedangkan nilai yang diberikan adalah estimasi sudut yang dihitung oleh aplikasi perangkat lunak. Sedangkan cell yang berwarna merah merupakan indicator bahwa estimasi rotasi tidak valid, karena perbedaannya terpaut sangat jauh. apabila cell yang berwarna merah diabaikan, maka dapat ditemukan nilai maximal perbedaan estimasi rotasi oleh perangkat lunak adalah 6.34 o dan nilai minimum nya adalah -8.13 o. Dari data Tabel 4.3 dapat dilihat pada jarak marker 20 cm hingga 210 cm dari kamera, marker dapat terdeteksi dan dikenali dengan cukup baik. Baik pada rotasi 0 o, 45 o, 90 o,135 o, dan 180 o. Nilai perbedaan maksimum yang dihasilkan adalah 6.71 o, sedangkan nilai perbedaan minimumnya adalah -6.71 o. Sedangkan pada jarak 240 cm dengan sudut 90 o dan 135 o, terlihat tanda- tanda estimasi rotasi marker mulai tidak akurat. Dapat dilihat pada Error! eference source not found. kotak kecil yang dipakai sebagai acuan dalam menentukan titik utama mulai tidak terlihat di kamera. Sehingga mengakibatkan proses pengenalan menjadi kacau. Pada jarak marker 270 cm dari kamera, marker masih dapat terdeteksi, namun untuk penentuan titik-titik kotak tidak valid. Tingkat error semakin meningkat. Pada sudut 90 o dan 180 o dianggap mendekati dengan sudut 0 o . sedangkan sudut 135 o dianggap mendekati dengan 45 o.

1. **Pengujian Tingkat Kemiringan *Marker* Pada Jarak Dan Rotasi Tertentu**

Pengujian ini merupakan tahap pengujian lebih lanjut dari pengujian sebelumnya. Pada pengujian kali ini faktor rotasi marker juga akan diperhitungkan. Marker akan di miringkan pada sudut 30 o, 60 o, 120 o, 150 o pada jarak 30 cm hingga 210 cm dan akan dirotasi sebanyak 45 o



Gambar 4.32. Gambar jarak marker 30 cm dari kamera dengan kemiringan 120o dan 150o.

1. **Pengujian Dengan Menggunakan Webcam Yang Berbeda**

Pada pengujian ini, akan dilakukan uji coba seperti dengan pengujian 4.2 tetapi dengan menggunakan 3 webcam yang tipe dan resolusinya berbeda. Tujuan dari pengujian ini adalah untuk mengetahui efek dari webcam yang berbeda.



Gambar 4.33. Gambar kamera yang dipakai

Spesifikasi webcam yang digunakan berurutan dari kiri adalah :

1. Webcam intel CS110 dengan resolusi maksimal 320x240
2. logitect quickcam Webcam P/N 861080 dengan resolusi maksimal 320x240
3. Webcam logitech Quickcam Communicate STX P/N 861223-0000. (Webcam yang dipakai pada pengujian-pengujian sebelumnya).

Tabel 4.4. Tabel hasil pengujian dengan 3 kamera

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | **Orientasi** | | | | | | |
|  |  | 0 o | 45 o | 60 o | 90 o | 145 o | 225 o | 270 o |
| **kamera** | 1 | 0 | 41.19 | 70.35 | 84.09 | 139.97 | 214.16 | 267.95 |
| 2 | 358.21 | 40.82 | 69.44 | 85.43 | 137.82/34.82 | 36.47 | 358.21 |
| 3 | 0 | 36.38 | 69.23 | 89.05 | 140.31 | 217.63 | 269.31 |

Hasil pengujian pada tabel diatas, orientasi 0-90 derajat, dapat dilihat bahwa terdapat perbedaan estimasi rotasi marker. Namun tidak terlalu signifikan. Akan tetapi pada orientasi 145 o hingga 270 o, pada kamera-2 estimasi mulai terlihat kacau. Hal ini dikarenakan spesifikasi kamera yang kurang memadai. Dari hasil pengamatan pada kamera-2, hasil image yang didapat lebih terdapat noise, flicker dan tidak fokus. Hal itu menyebabkan estimasi rotasi pada kamera-2 tidak stabil / gampang berubah.

Gambar 4.34. Perbandingan hasil dari kamera-1 dan kamera-2



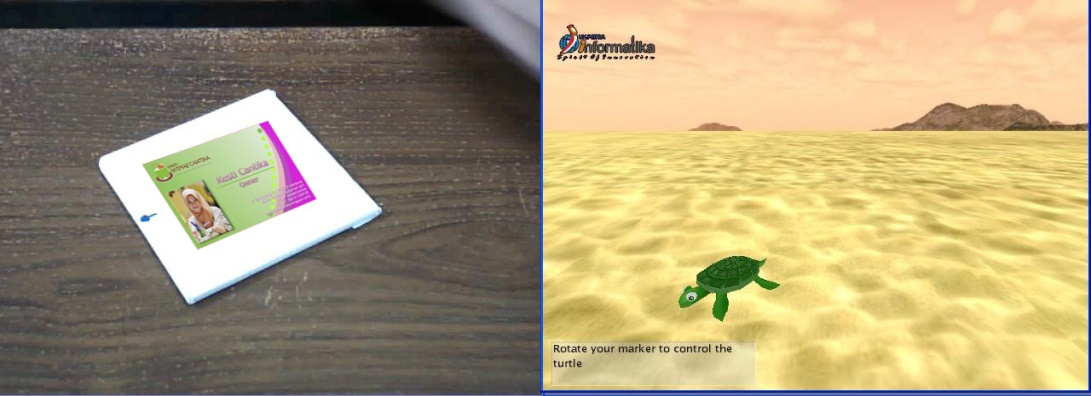


1. **Pengujian Prototyping Game**

Pada pengujian ini, core aplikasi marke detection perangkat lunak Tugas Akhir ini akan digabung dengan game engine Irrlicht. Marker digunakan untuk mengontrol objek kura-kura untuk pada suatu prototype game sederhana. Input yang digunakan berasal dari kamera webcam untuk menangkap image. Kemudian akan dilakukan pendeteksian marker dan estimasi rotasi yang terjadi. Nilai estimasi tersebut nantinya akan digunakan untuk mengontrol arah kura-kura.



Gambar 4.35. Gambar dari *webcam (kiri)* dan gambar objek kura-kura yang mengarah ke bawah(kanan).



Gambar 4.36. Gambar dari *webcam (kiri)* dan gambar objek kura-kura yang mengarah diagonal bawah kiri (kanan)

**BAB V**

**PENUTUP**

1. **Kesimpulan**

Berdasarkan sistem yang telah dikembangkan dan hasil pengujian yang telah ditentukan, dapat disimpulkan beberapa hal sebagai berikut:

1. Estimasi rotasi pada pengujian rotasi *marker* pada sumbu pusat terdapat perbedaan sudut dengan sudut asli maksimal 8.62 o dan minimal -9.23 o.
2. Estimasi rotasi pada pengujian rotasi *marker* pada titik tertentu, terdapat perbadaan sudut dengan sudut asli maksimal 9.36 o dan minimal -6.04 o.
3. Estimasi rotasi dengan cara menghitung 2 garis terkadang memberikan hasil yang relative akurat dengan perbedaan kurang lebih +- 10 derajat. Hal ini dikarenakan *parallax* atau perbedaan posisi sudut pengamatan yang menyebabkan terjadi perbedaan posisi apa yang dilihat kamera dengan yang sebenarnya.
4. Faktor external seperti cahaya yang terlalu terang atau bayangan yang menutupi *marker* dapat mempengaruhi kinerja pendeteksian.
5. Sistem pendeteksian *marker* tergolong memiliki performa yang cukup bagus.
6. *Marker* dapat dideteksi hingga pada jarak 3.3 meter(330 cm). Namun jarak yang efektif untuk pengenalan rotasi adalah 20 cm hingga 2.1 meter(210 cm) dari kamera.
7. Derajat kemiringan *marker* memiliki pengaruh dalam kinerja pendeteksian *marker*. Karakteristik jarak efektif maksimal *marker* dari kamera adalah :
8. Pada derajat kemiringan *marker* 30 o adalah 210 cm. o
9. Pada derajat kemiringan *marker* 60 o adalah 240 cm. o
10. Pada derajat kemiringan *marker* 90 o adalah 270 cm. o
11. Pada derajat kemiringan *marker* 120 o adalah 180 cm o
12. Pada derajat kemiringan *marker* 150 o adalah 150 cm o
13. Tingkat kemiringan marker memiliki pengaruh terhadap estimasi rotasi
14. System pendeteksian marker dapat berjalan pada webcam dengan spesifikasi resolusi rendah, namun tingkat akurasi estimasi rotasi semakin menurun.
15. Sistem inti pendeteksian dan estimasi rotasi marker ini dapat dikombinasikan dengan platform yang lain. Misalnya sebagai controller suatu objek pada suatu game tertentu.
16. **Saran**

Terdapat beberapa saran yang diharapkan dapat mendukung pengembangan lebih lanjut, yaitu :

1. Pengembangan metode estimasi rotasi yang lebih akurat agar lebih mendekati dengan rotasi yang terjadi pada dunia nyata.
2. Pengenalan marker lebih dari 1 jenis
3. Implementasi augmented reality pada project lain. Misalnya: advertising

**DAFTAR PUSTAKA**

Andrew, Greensted(2010)*. Otsu Thresholding.* Retrieved Februari 23, 2012, from: <http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html> *An introduction to contours (n.d).* Retrieved 24 February 2012, from: <http://www.aishack.in/2010/01/an-introduction-to-contours/>

Bowo Dwi Ariyanto, M. hariadi. (2010). simluasi perilaku pergerakan objek 3d media augmented reality berbasis logika fuzzy, from<http://digilib.its.ac.id/ITS-Master-3100012045760/18002> *Camera Calibration(n.d.).* Retrieved October 11, 2011, from: <http://www.umiacs.umd.edu/~ramani/cmsc828d/lecture9.pdf>

*Documentation: Image Segmentation-Dynamic Thresholding*. Retrieved Februari 22, 2012, from: <http://www.dandiggins.co.uk/arlib-3.html>

*Documentation: Geometric Image Transformations*. Retrieved Februari 22, 2012, from: [http://opencv.willowgarage.com/documentation](http://opencv.willowgarage.com/documentation%20/geometric_image_transformations.html)

*Documentation: Camera Calibration and 3D Reconstruction*. Retrieved Februari 22, 2012, from: <http://opencv.willowgarage.com/documentation/>

Gady, Adam (2006). *Introduction to programming with OpenCV*. Retrieved October 11, 2011, from: [http://www.cs.iit.edu/~agam/cs512/lect- notes/opencv-intro/opencv-intro.html](http://www.cs.iit.edu/~agam/cs512/lect-notes/opencv-intro/opencv-intro.html)

Gopall, D.J, Jayanthi,S(2008). A simple Scheme for Contour Detection. Retrieved 24 February 2012, from: <http://cvit.iiit.ac.in/papers/>

Hongzhi, John. *generalizing Edge Detection To Contour Detection for Image*