Rancang Bangun Permainan Dribel Bola Basket *"Basketball Jam"* Untuk Remaja Menggunakan *Kinect*

Elizabeth Irenne Yuwono, S.Kom, Universitas Ciputra, Surabaya 60219
Nehemia Sugianto, S.T, Universitas Ciputra, Surabaya 60219

ABSTRAK

Olahraga bola basket adalah salah satu olahraga yang paling digemari oleh anak usia remaja baik pria maupun wanita (Statista, 2015). Olahraga ini tidak hanya memberikan manfaat kesehatan, namun juga melatih kepemimpinan dan meningkatkan konsentrasi pemainnya. Sejalan dengan perkembangan teknologi, banyak olahraga digunakan sebagai konten permainan digital untuk sebagai media hiburan dan edukasi. Penelitian ini bertujuan membuat permainan digital dribel bola basket mengikuti iringan musik dengan menggunakan *Kinect*. Pemain menggunakan bola basket sebenarnya dan melakukan gerakan dribel di depan *Kinect*. Kemudian data yang diperoleh dari *Kinect (ColorData, DepthData, SkeletonData)* digunakan untuk deteksi gerakan dribel selama permainan. Adapun dalam permainan ini, terdapat tiga jenis gerakan dribel, yaitu gerakan dribel di kiri, kanan, dan tengah. Deteksi dilakukan dalam dua tahap yaitu deteksi bola basket dan deteksi gerakan dribel. Sistem mendeteksi bola basket apabila tangan pemain memegang bola dengan asumsi bola basket sebagai objek terdekat. Sistem melakukan identifikasi bola basket melalui fitur lingkaran menggunakan *Hough Circle Detection*. Pemrosesan gambar selama tahap deteksi ini menggunakan *EmguCV*. Data posisi pemain diperoleh melalui *SkeletonData* pada *KinectSensor*. Sementara gerakan dribel dideteksi apabila tangan kanan pemain dan bola basket berada di area kiri atau kanan atau tengah. Area ditentukan berdasarkan titik tengah bingkai *ColorData (ColorFrame)*. Berdasarkan hasil implementasi, sistem mampu memproses data-data dari *Kinect*. Deteksi bola basket berhasil dilakukan namun penggunaan fitur deteksi lingkaran mengurangi performa sistem. Deteksi gerakan dribel kiri, kanan, tengah berhasil dilakukan namun beberapa kasus saat pemain menggunakan tangan kiri atau melakukan gerakan dribel *cross-over* tidak dapat dideteksi.

Kata kunci: bola basket, gerakan dribel, computer vision, pengolahan citra digital, feature extraction, EmguCV, Kinect, Skeleton Tracking, Gaussian Smoothing, Hough Circle Detection

1. Pendahuluan

Olahraga bola basket adalah salah satu olahraga yang paling sering dilakukan dengan 2-3 milyar penggemar di seluruh dunia berdasarkan data dari Sporteology (Mughal, 2015). Menurut Baumman, salah satu eksekutif FIBA (Federasi Olahraga Bola Basket Internasional); olahraga terpopuler untuk rentang usia 14 – 18 tahun pria dan wanita adalah bola basket (Statista, 2015).

Sejalan dengan perkembangan teknologi, banyak olahraga yang dikembangkan dalam bentuk permainan digital. Penelitian ini bertujuan untuk membuat permainan digital dribel bola basket dengan iringan musik menggunakan *Kinect*.

2. Olahraga Bola Basket

Olahraga bola basket adalah salah satu olahraga terpopuler di dunia dalam bentuk kompetisi antara dua tim yang terdiri dari 5 orang pemain dengan tujuan mencetak skor sebanyak mungkin dengan memasukkan bola basket ke ring setinggi 3 meter. Saat ini olahraga bola basket telah dikembangkan menjadi berbagai jenis format permainan yang kreatif dan memiliki unsur hiburan untuk membantu pelatihan dan pendidikan dasar olahraga bola basket. Olahraga bola basket memberikan berbagai jenis manfaat, antara lain: manfaat edukasi kepemimpinan dan kerjasama, kesehatan, rekreasi, dan secara psikologis meningkatkan fokus mental pemain.

* *Elizabeth Irenne.* Tel.: +62-31-7451699 Ext. 3171.
E-mail: eirene@ciputra.ac.id

Olaharaga ini menggunakan bola basket yang memiliki diameter standar sekitar 20 - 25 cm. Bola ini memiliki warna standar oranye bergaris hitam terbuat dari bahan kulit atau karet sintetik.

Terdapat beberapa gerakan dasar dalam olahraga bola basket, antara lain: dribel, lompat *(hop)*, *passing*, dan melemparkan bola ke dalam ring *(shot)*. Diantara keempat gerakan dasar ini, dribel bola merupakan salah satu gerakan yang paling sering dilakukan saat bermain bola basket. Gerakan ini adalah gerakan mengendalikan bola dengan satu tangan sehingga bola basket melambung naik turun dengan lantai di depan tubuh.

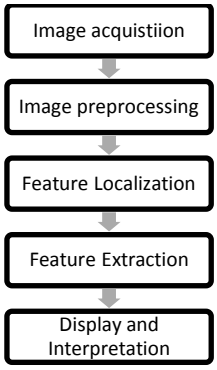
3. Computer Vision

Computer vision adalah bidang ilmu yang berkonsentrasi pada analisa, modifikasi dan memahami gambar untuk mencapai tujuan tertentu. Bidang ilmu ini digunakan untuk memperoleh informasi dari gambar yang dikirim sensor kamera untuk mengetahui apa yang terjadi di depan kamera, menciptakan sistem pintar dengan kecerdasan buatan, ataupun mengimplementasikkannya pada sistem robotik untuk memberikan respon tertentu dari kejadian yang ditangkap sensor kamera.

4. Pengolahan Citra Digital

Pengolahan citra digital adalah bagian dari bidang ilmu *computer vision* yang berkonsentrasi pada pengolahan citra dua dimensi secara digital. Pemrosesan citra meliputi; modifikasi informasi, klasifikasi, penggabungan dua bagian citra, dan pencocokan elemen dalam citra nuntuk memperoleh kualitas lebih tinggi atau mengubah penyajian citra sesuai kebutuhan sistem (Arymurthi, 1992).

Menurut Deligiannidis, terdapat beberapa fase dalam proses pengolahan citra digital, antara lain: akuisisi citra, pre-proses citra, lokalisasi fitur, ekstrasi fitur, menampilkan hasil dan intepretasinya, seperti ditunjukkan pada gambar 1.



Gambar 1. Langkah-langkah proses pengolahan citra (Deligiannidis, 2015)

Pada tahap awal, dilakukan akuisisi citra dan pre-proses untuk memperoleh informasi-informasi penting dalam citra seperti ukuran, resolusi, pixel, dan sebagainya. Selanjutnya sistem melakukan lokalisasi fitur untuk menentukan fitur-fitur apa saja yang dapat diidentifikasi, seperti fitur warna, garis, bentuk, dan lain-lain. Fitur-fitur ini kemudian diambil (ekstrasi fitur) sesuai kebutuhan sistem. Kemudian informasi fitur dan citra hasil ekstrasi ditampilkan sebagai luaran dari proses pengolahan citra digital.

5. Feature Extraction

Feature extraction adalah proses pengambilan fitur pada sebuah data digital, dalam hal ini citra digital. Terdapat dua tahap dalam ekstrasi fitur yaitu: konstruksi fitur dan seleksi fitur. Konstruksi fitur adalah tahap penentuan fitur-fitur apa saja yang diperlukan oleh sistem sebagai kondisi analisa data. Sementara seleksi fitur dilakukan untuk memilih dari fitur-fitur yang diperoleh dari konstruksi fitur. Seleksi fitur diperlukan untuk meningkatkan performa sistem. membatasi kebutuhan memori, dan memperoleh pengetahuan dari fitur-fitur tersebut.

Terdapat beberapa fitur dalam citra digital yang dapat diperoleh langsung, disebut dengan fitur dasar *(low level)*, antara lain: deteksi tepi suatu objek *(edge detection)*, deteksi sudut *(corner detection)*, deteksi objek sejenis *(blob detection)*, deteksi puncak area *(ridge detection)*, deteksi fitur lokal *(SIFT – Scale-invariant Feature Transform)*. Fitur-fitur dasar ini merupakan jenis fitur paling sederhana dan dapat digunakan untuk deteksi fitur selanjutnya seperti bentuk *(Hough Transform)*, arah gerakan, warna, dan sebagainya.

Terdapat dua tahap deteksi dalam sistem *"Basketball Jam"*, yaitu deteksi bola basket dan gerakan pemain. Untuk tahap deteksi bola basket, citra digital yang diperoleh dari *Kinect* diproses dan dilakukan pengambilan fitur bentuk lingkaran menggunakan *Hough Circle Detection.* Sementara dalam tahap deteksi gerakan pemain, fitur posisi tangan dan kaki pemain digunakan.

5.1. Image Smoothing

Image smoothing adalah proses pengolahan citra digital untuk mengurangi *noise* pada tingkat warna abu-abu. Menurut Goshtasby (2005), *noise* pada citra berkontribusi pada tingginya frekuensi spasial sebuah citra, operasi *smoothing* dapat mengurangi besarnya frekuensi spasial. Semakin tinggi frekuensi spasial maka semakin tinggi tingkat kontras citra. Semakin rendah frekuensi spasial, semakin halus sebuah citra. Operasi *smoothing* dapat dilakukan menggunakan konvolusi atau dengan *filtering.* Terdapat dua jenis operasi *smoothing* yang menggunakan konvolusi, yaitu: *linear smoothing* dan *nonlinear smoothing.* Menurut Joshi (2006), *linear smoothing* melakukan konvolusi sehingga diperoleh efek fokus halus pada citra, dengan matriks 3x3 yang sama pada setiap pixel citra. Matriks untuk konvolusi pada *linear smoothing* ditunjukkan pada persamaan 1.

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \quad (1)$$

Nonlinear smoothing mengatasi kekurangan pada *linear smoothing* yang menganggap setiap sinyal dalam citra adalah sama dengan memperlakukan setiap picel dengan berbeda. Persamaan 2 menunjukkan matriks konvolusi untuk *nonlinear smoothing.*

$$\frac{1}{8}\begin{bmatrix}1 & 1 & 1\\ 1 & 0 & 1\\ 1 & 1 & 1\end{bmatrix} \quad (2)$$

Untuk mengurangi *zero-mean* atau *white noise* digunakan *Gaussian filtering.* Dalam *Gaussian filtering*, ukuran *filter* menentukan besar *smoothing* yang ingin diimplementasikan pada citra. Semakin besar ukuran *filter* maka hasil pengolahan citra akan semakin kabur *(blur)*, sebaliknya semakin kecil ukuran *filter* hasil pengolahan citra tidak akan terlalu kabur namun *noise* yang tersisa lebih banyak. (Saphiro, 2000) Persamaan 3 menunjukkan perhitungan *Gaussian filter* pada pengolahan citra. $g(x, y)$ menunjukkan pixel hasil operasi *Gaussian filter, d* adalah jarak antara pixel $[x, y]$ dengan pixel pusat $[x_c, y_c]$.

$$g(x,y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{d^2}{2\sigma^2}} \quad (3)$$

where $d = \sqrt{(x - x_c)^2 + (y - y_c)^2}$

5.2. Hough Circle Detection

Dalam pengolahan citra terdapat beberapa metode ekstrasi fitur yang dapat digunakan untuk pengenalan bentuk seperti garis, dan lingkaran. Metode yang paling sering digunakan adalah *Hough Transform. Hough Transform* digunakan untuk menentukan parameter lingkaran ketika sejumlah titik berada pada garis keliling lingkaran. Menurut Seul, parameter yang digunakan dalam *Hough Circle Detection* adalah parameter 3 dimensi *(a, b, r)* dimana *a* dan *b* adalah koordinat titik pusat dan *r* adalah radius lingkaran (Seul, 2000). Persamaan 4 menunjukkan persamaan untuk deteksi bentuk lingkaran menggunakan *Hough Transform.*

$$(x - a)^2 + (y - b)^2 = r^2 \quad (4)$$

Berdasarkan persamaan tersebut, setiap koordinat titik dalam domain spasial dipetakan ke koordinat-koordinat yang dapat digunakan dalam area parameter.

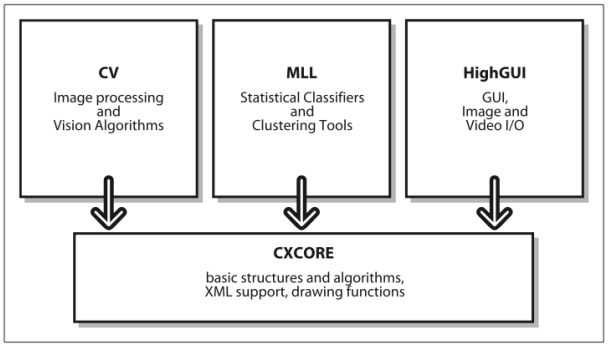
6. OpenCV Library

OpenCV Library (*Open Source Computer Vision Library*) adalah *library* untuk efisiensi komputasi untuk permasalahan *computer vision* pada aplikasi *real-tim* dan bersifat *open-source. OpenCV library* menyediakan antarmuka untuk bahasa pemrograman C++, C, Python, dan Java. *Library* ini dapat dijalankan pada sistem operasi Windows, Mac OS, iOS, Linux, dan Android.

OpenCV Library menyediakan berbagai jenis modul untuk membantu komputasi bidang ilmu *computer vision*, antara lain:

- **core**, modul untuk penggunaan struktur data dan fungsi dasar seperti array untuk penyimpanan informasi citra.
- **imgproc**, modul pengolahan citra untuk *filtering* linear dan non-linear, transformasi citra secara geometrik, konversi area warna, histogram, dan lain-lain.
- **video**, modul untuk analisis informasi pada video termasuk deteksi objek pada video digital.
- **calib3d**, modul untuk implementasi algoritma geometrik, rekonstruksi 3D, estimasi objek, dan sejenisnya.
- **features2d**, modul untuk *feature extraction*, termasuk deteksi dan pencocokkan fitur
- **objdetect**, modul untuk deteksi objek pada citra
- **highgui,** modul antarmuka untuk pemrosesan video, termasuk analisa *codec* citra dan video.
- **gpu**, modul untuk implementasi algoritma-algoritma *GPU-accelerated* dari modul *OpenCV* lain.

Berdasarkan fungsional setiap modulnya, *OpenCV Library* dibagi menjadi empat komponen utama yaitu *CV*, *MLL*, *HighGUI*, dan *CXCORE* (Bradski, Gary dan Kaehler, Adrian, 2008). Komponen *CV* meliputi pemrosesan citra dan implementasi algoritma *computer vision.* Sementara komponen *MLL* merupakan modul-modul untuk klasifikasi dan pengelompokkan data *(clustering)* secara statistik. Komponen *HighGUI* meliputi antar muka dan proses pengolahan informasi dari citra dan video. Komponen *CXCORE* adalah modul-modul yang mengatur struktur-struktur data, fungsi, dan algoritma dasar yang digunakan oleh modul-modul pada komponen *CV*, *MLL*, dan *HighGUI*.



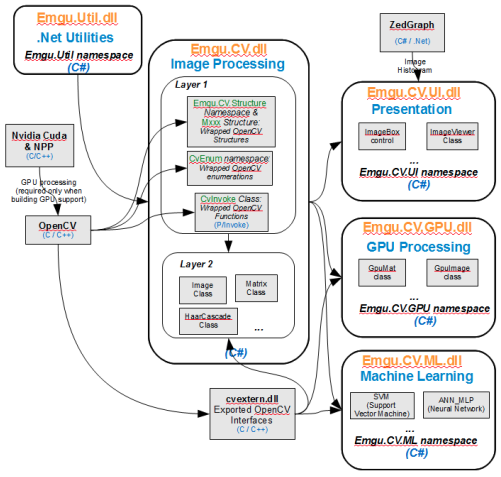
Gambar 2. Struktur Dasar *OpenCV* (Bradski, Gary dan Kaehler, Adrian, 2008)

7. EmguCV Library

EmguCV library adalah *.Net wrapper* untuk *OpenCV Library* pada platform berbeda. *EmguCV Library* dapat memanggil fungsi-fungsi *OpenCV* melalui bahasa

pemrograman yang sesuai dengan *.NET* seperti *C#, VB, VC++, IronPython,* dan lain-lain. Tersedia beberapa *wrapper* untuk *EmguCV,* antara lain *Visual Studio, Xamarin Studio,* dan *Unity.*

EmguCV library memberikan beberapa manfaat dalam kemudahan komputasi *computer vision,* antara lain: *cross platform, cross languages, image class with generic color and depth, generic operations on image pixels,* dan pilihan untuk akses *image class* atau *direct invoke functions* dari *OpenCV.*

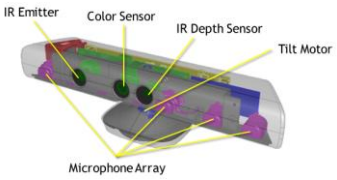


Gambar 3. Arsitektur *OpenCV* (www.*emgu.com*)

8. Kinect

Kinect adalah sebuah perangkat keras yang memiliki sensor-sensor untuk pengambilan citra digital dengan kualitas lebih baik dibandingkan kamera biasa atau sensor sejenisnya, dengan konsentrasi pada sensor warna RGB, kedalaman *(depth)* melalui infra merah, dan deteksi rangka tubuh manusia. *Kinect* terdiri dari beberapa kamera, sebuah *microphone array* dan sebuah *accelerometer*. Berikut adalah komponen-komponen penting dari *Kinect* versi 1:

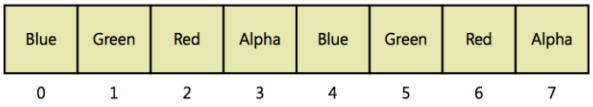
- Sebuah *RGB camera* yang mampu menangkap gambar dan menghasilkan sebuah citra berwarna (*RGB data*) dalam tiga *channel* dengan resolusi maksimal hingga 1280 x 960 *pixel.*
- Sebuah *infrared (IR) emitter* dan *infrared (IR) depth sensor*. *Infrared (IR) emitter* memancarkan cahaya infra merah. *Infrared (IR) depth sensor* menangkap pantulan dari cahaya infra merah tersebut. Pantulan dari cahaya infra merah tersebut dikonversi menjadi informasi kedalaman (*depth data*) yang berisi jarak antara obyek dengan sensor.
- Sebuah *multi-array microphone* yang terdiri dari empat mikrofon untuk menangkap data suara yang untuk merekam suara, mengetahui lokasi sumber dan arah gelombang suara.
- Sebuah *3-axis accelerometer* yang dikonfigurasi untuk jangkauan 2G dimana G merupakan akselerasi terhadap gravitasi bumi. *Accelerometer* ini dapat digunakan untuk menentukan orientasi *kinect* saat ini.



Gambar 4. *Kinect* Versi 1 (www.*microsoft.com*)

8.1. Color Data

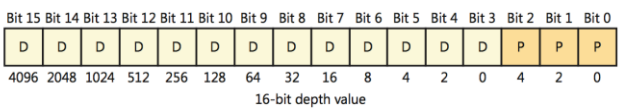
Citra berwarna yang diperoleh dari kamera RGB pada *Kinect* disebut dengan *color data. Color data* menyimpan kumpulan *pixel*, setiap *pixel* berisi empat nilai warna dasar; merah, hijau, dan biru serta tingkat transparansi *(alpha).* Setiap warna dasar memiliki nilai batas 0 hingga 255, semakin tinggi sebuah nilai warna dasar maka intensitas tampilan warna tersebut akan semakin kuat. *Color data* disimpan dalam sebuah array satu dimensi bertipe *byte.* Setiap warna dasar direpresentasikan oleh 1 *byte*, sehingga setiap *pixel* memiliki 4 *byte.* Gambar 5 menunjukkan representasi penyimpanan *color data.*



Gambar 5. Representasi penyimpanan citra bertipe *color data* (Miles, 2012)

8.2. Depth Data

Data yang diperoleh dari *IR emitter* dan *IR depth sensor* pada *Kinect* disimpan dalam *depth data. Depth data* menyimpan informasi jarak antara kamera dengan titik obyek dari citra yang tertangkap oleh *Kinect. Depth data* menyimpan jarak antara kamera dengan objek dalam satuan milimeter, dengan kapasitas deteksi kedalaman antara 0,8 hingga 4 meter. Pada sistem, *depth data* disimpan dalam bentuk *pixel* 16 bit. 3 bit pertama menyimpan informasi nomor urut *player* yang terdeteksi oleh *Kinect* sedangkan 13 bit selanjutnya menyimpan informasi jarak dari kamera menuju titik objek. Gambar 6 menunjukkan representasi *pixel* pada *depth data*.

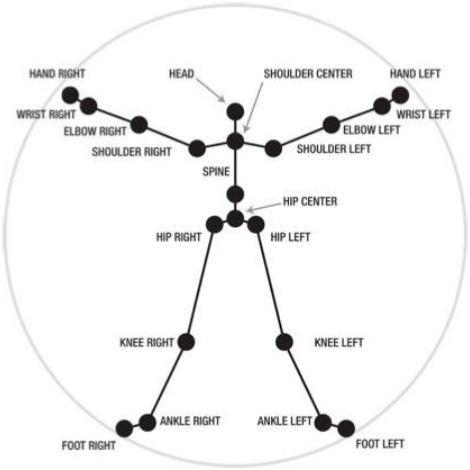


Gambar 6. Representasi sebuah *pixel* pada citra bertipe *depth data* (Miles, 2012)

8.3. Skeleton Data

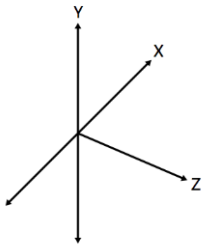
Kinect mampu mendeteksi posisi sendi-sendi pada rangka tubuh manusia. Manusia terdiri dari 206 tulang (*bones*) yang saling terhubung bersama untuk menjadi

sebuah kerangka (*skeleton*) yang dapat menyokong tubuh manusia (*body*) (Miles, 2012). *Kinect* merepresentasi tulang-tulang tersebut menjadi 19 tulang yang dihubungkan oleh 20 sendi seperti yang ditunjukkan oleh gambar 7. Posisi rangka tubuh manusia ini diperoleh dari perpaduan antara *color data* dan *depth data*. *Kinect* versi 1 mampu mendeteksi rangka tubuh pemain hingga enam pemain, namun *Kinect* hanya mampu mendeteksi rangka tubuh secara lengkap pada dua pemain pertama saja.



Gambar 7. Ilustrasi Sendi-Sendi Tubuh Manusia Yang Ditangkap Oleh *Kinect* (Webb, 2012)

Skeleton data yang diperoleh dari *Kinect* menyimpan koordinat 3D dari posisi sendi-sendi tubuh pemain. Sistem koordinat *skeleton data* menggunakan sumbu X, sumbu Y dan sumbu Z yang berpotongan (0,0,0). *Kinect* menghadap sumbu Z. Jika pemain bergerak ke kanan maka nilai X pada tiap sendi semakin tinggi, begitu pula sebaliknya. Jika pemain berjinjit maka nilai Y pada tiap sendi semakin tinggi, begitu pula sebaliknya. Jika pemain bergerak menjauhi *Kinect* maka nilai Z pada tiap sendi semakin tinggi, begitu pula sebaliknya. *Skeleton data* dapat diproses sesuai kebutuhan sistem untuk pengolahan pada koordinat 2D, terdapat dua *method* untuk melakukan transformasi dari koordinat 2D menjadi koordinat 3D yaitu *MapSkeletonPointToColorPoint* (transformasi koordinat sesuai *color data)* dan *MapSkeletonPointToDepthPoint* (transformasi koordinat sesuai *depth data)*.



Gambar 8. Sistem Koordinat Ruang Kerangka Manusia Pada *Kinect* (Webb, 2012)

Hasil pelacakan tubuh manusia bisa dibagi menjadi tiga macam kondisi yaitu *Tracked*, *Untracked* dan *Position Only*. *Tracked* berarti hasil pelacakan tubuh manusia tersebut berhasil didapatkan. *Untracked* berarti hasil pelacakan tubuh manusia tersebut tidak berhasil didapatkan. *Position Only* berarti hasil pelacakan tubuh manusia tersebut tidak berhasil didapatkan secara sempurna karena tidak berhasil mendapatkan informasi hingga tiap sendinya tetapi masih bisa melakukan pelacakan tubuh manusia.

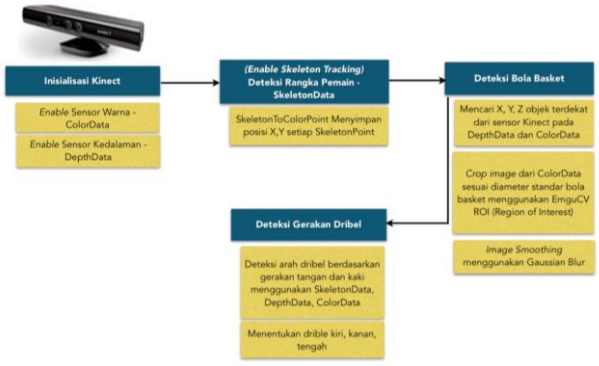
Nilai posisi sebuah sendi pada pelacakan tubuh manusia bisa dibagi menjadi tiga macam kondisi yaitu *Tracked*, *Inferred* dan *Not Tracked*. *Tracked* berarti nilai posisi sendi tersebut pada pelacakan tubuh manusia tersebut berhasil didapatkan dengan baik. *Inferred* berarti *Kinect* tidak dapat melacak sendi tersebut tetapi masih bisa mendapatkan nilai posisinya berdasarkan posisi dari sendi-sendi yang lain pada pelacakan tubuh manusia tersebut. *Not Tracked* berarti nilai posisi sendi tersebut pada pelacakan tubuh manusia tersebut tidak berhasil didapatkan.

9. Desain Sistem *Basketball Jam*

Basketball Jam menggunakan Kinect sebagai media permainan dan sensor warna *(color)*, kedalaman *(depth)*, dan kerangka pemain *(skeleton)*. Gambar 8 menunjukkan desain sistem *Basketball Jam.*

9.1. Inisialisasi Kinect

Pada tahap awal dilakukan inisialisasi pada *Kinect*, seluruh sensor menyala dan melakukan pembaharuan terus-menerus pada *ColorData* dan *DepthData*. Fitur *SkeletonTracking* juga digunakan untuk memperoleh *SkeletonData*.



Gambar 9. Desain Sistem *Basketball Jam*

9.2. Deteksi Rangka Pemain

SkeletonData digunakan untuk mengetahui keberadaan pemain. Jika *SkeletonData* tidak null maka sensor mendeteksi pemain dan akan menyimpan posisi sendi-sendi tubuh pemain dalam satuan *SkeletonPoint*. Posisi X,Y diperoleh menggunakan metode *SkeletonToColorPoint*, sementara posisi Z diperoleh menggunakan *SkeletonToDepthPoint.* Posisi X,Y,Z setiap sendi digunakan sebagai acuan selama proses

deteksi bola basket dan gerakan dribel.

9.3. Deteksi Bola Basket

Pada tahap ini diasumsikan bola basket merupakan objek terdekat dengan *Kinect*. Sistem menyimpan titik tengah objek terdekat menggunakan *DepthToColorPoint* pada *DepthData* dan *ColorData*. Kemudian sistem menghitung diameter bola pada *ColorData* menggunakan rasio antara diameter standard bola basket dengan Z titik tengah objek. Citra bola di-*crop* menggunakan *EmguCV* ROI *(Region of Interest)* dengan diameter hasil perhitungan dan titik tengah dari objek terdekat. *Noise* pada citra dikurangi menggunakan *Gaussian Blur*.

Terdapat dua kondisi untuk tahap deteksi ini, yaitu: ada lingkaran dengan diameter kurang lebih sama dengan diameter bola dan adanya tangan pemain. Deteksi lingkaran dilakukan menggunakan *Hough Circle Detection*. Sementara posisi tangan pemain diperoleh dari *SkeletonData*. Apabila kedua kondisi terpenuhi maka sistem mendeteksi bahwa pemain sedang memegang bola basket.

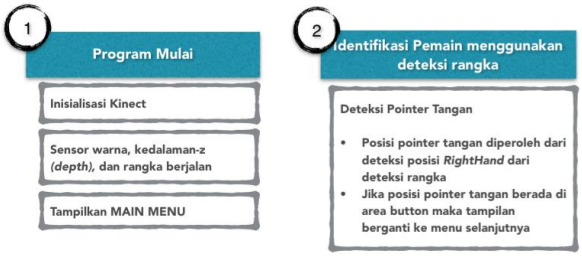
9.4. Deteksi Gerakan Dribel

Sistem dapat mendeteksi tiga jenis gerakan dribel; dribel bola di kiri (< titik tengah *ColorFrame*), kanan (> titik tengah *ColorFrame*), dan tengah (= titik tengah *ColorFrame*). Gerakan dribel dideteksi apabila:

- Tangan kanan berada di area kiri / kanan / tengah
- Bola basket berada di area bawah kiri / kanan / tengah. Area bawah adalah area antara lutut hingga lantai.

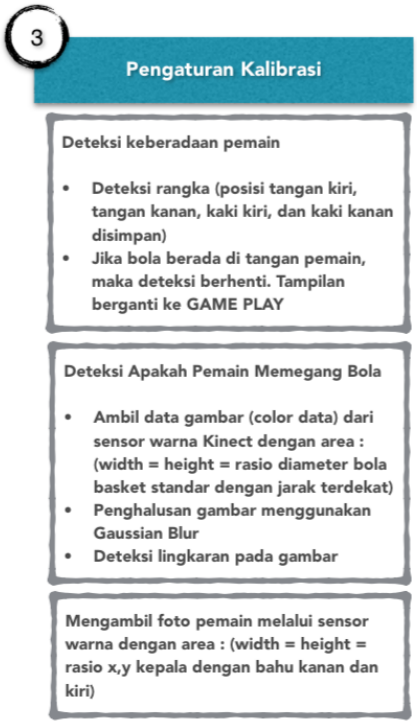
10. Diagram Alur Kerja *Basketball Jam*

Bagian ini menjelaskan hasil implementasi desain sistem *Basketball Jam* dalam bentuk alur kerja. Terdapat lima fase dalam implementasi sistem *Basketball Jam*. Gambar 10 menunjukkan fase 1 dan 2. Fase pertama adalah inisialisasi program untuk tampilan menu permainan dan *Kinect*. Pada fase kedua *SkeletonData* dari *Kinect* untuk mendeteksi keberadaan pemain dan menyimpan posisi sendi-sendi tubuh.

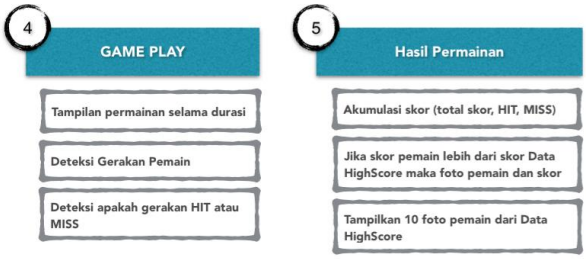


Gambar 10. Alur Kerja Program *Basketball Jam* fase 1 dan 2

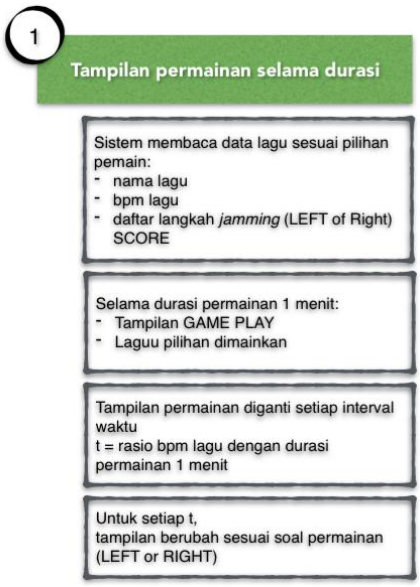
Gambar 11 menunjukkan alur kerja fase ketiga, yaitu pengaturan kalibrasi untuk mengetahui posisi pemain dan bola basket. Pengaturan ini dijadikan acuan apakah permainan dapat dimulai. Pada fase ini foto pemain juga diambil sebagai data apabila pemain berhasil mencapai skor tertinggi. Sementara gambar 12 menunjukkan alur kerja sistem fase keempat dan kelima. Fase keempat merupakan fase *Game Play*. Pada fase ini permainan dimulai dan deteksi dilakukan setiap t, t adalah rasio antara ketukan lagu dengan sekon (waktu). Gambar 13 menunjukkan fase awal saat *Game Play*, fase ini mengatur tampilan selama durasi waktu permainan. Sistem membaca data lagu yang akan digunakan sebagai acuan untuk tampilan kiri, kanan, dan tengah. Gambar 14 menunjukkan fase deteksi gerakan dribel yang dilakukan selama *Game Play*. Hasil dideteksi diidentifikasi dalam integer (0 = tengah, 1 = kiri, 2 = kanan). Deteksi dilakukan dengan menyimpan citra dengan area tinggi dari lutut hingga akhir bingkai *ColorData*. Posisi pusat tubuh, tangan kanan, tangan kiri diperoleh *SkeletonData*.



Gambar 11. Alur Kerja Program *Basketball Jam* fase 3

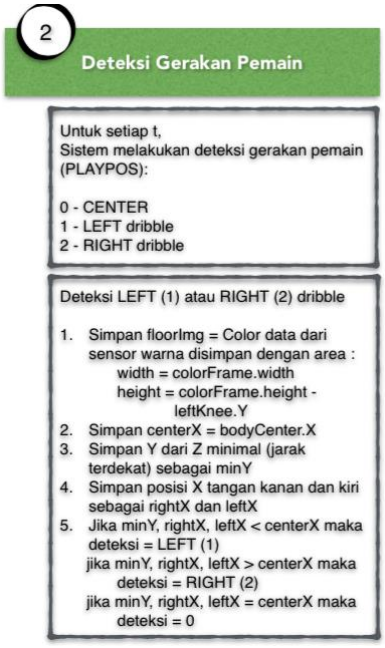


Gambar 12. Alur Kerja Program *Basketball Jam* fase 4 dan 5

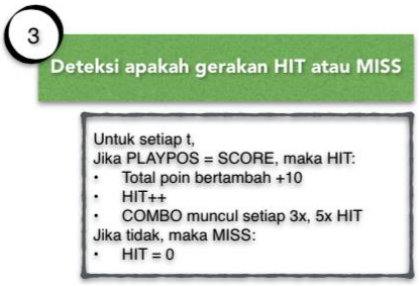


Gambar 13. Alur Kerja Program *Basketball Jam* Tahap *Game Play* tahap 4-1

Gambar 14 merupakan fase penentuan apakah gerakan dribel pemain sesuai dengan skor acuan. Pengecekkan dilakukan setiap *t* antara hasil gerakan dribel dengan skor dari data sistem. Apabila sesuai maka total poin pemain bertambah 10 poin. Selain itu terdapat fitur *Hit*, *Miss*, dan *Combo* yang akan muncul pada tampilan permainan sesuai perolehan skor.



Gambar 14. Alur Kerja Program *Basketball Jam* Tahap *Game Play* tahap 4-2



Gambar 15. Alur Kerja Program *Basketball Jam* Tahap *Game Play* tahap 4-3

11. Kesimpulan

Berdasarkan hasil implementasi program mampu memproses *ColorData*, *DepthData*, dan *SkeletonData* dari *Kinect*. Deteksi bola basket untuk tangan dan objek terdekat berhasil namun fitur lingkaran masih harus disempurnakan lagi karena mengurangi performa program. Deteksi gerakan dribel berhasil namun beberapa kasus saat pemain menggunakan tangan kiri dan gerakan dribel *cross-over* tidak dapat dideteksi.

DAFTAR PUSTAKA

Arymurthy, A., Setiawan, S. (1992). *Pengantar Pengolahan Citra.* Indonesia: PT. Elex Media Komputindo

Deligiannidis, L., Arabnia, H. (2015). *Emerging Trends in Image Processing, Computer Vision, and Pattern Recognition.* United States of America: Elsevier

Goshtasby, A. (2005). *2-D and 3-D Image Registration for Medical, Remote Sensing, and Industrial Applications.* New Jersey: John Wiley and Sons, Inc.

Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L. (2006). *Feature Extraction: Foundations and Applications*. United States of America: Springer.

Joshi, M. (2006). *Digital Image Processing: An Algorithm Approach.* India: Prentice-Hall of India.

Miles, R. (2012). *Start Here! Learn The Kinect API.* United States: Microsoft Press.

Mughal, K. (2015, March 7). Top 10 Most Popular Sports World. Retrieved April 10, 2015.

Oliver, J. (2004). *Basketball Fundamental: A Better Way to Learn The Basics*. United States of America: Human Kinetics Publishers, Inc.

Saphiro, L. (2000). *Computer Vision.*Washington: The University of Washington

Seul, M., O'Gorman, L., Sammon, M. (2000). *Practical Algorithms for Image Analysis: Description, Examples, and Code.* United Kingdom: Cambridge University Press.

Statista. (2015). Digital Games Industry Revenue in The United States in 2014, By Game Category (In Million U.S. Dollars). Retrieved April 10, 2015.

Triano, J. (2009). *Basketball Basics: How to Play Like The Pros.* Canada: Greystone Books.

Webb, J., Ashley, J. (2012). *Beginning Kinect Programming With The Microsoft Kinect SDK 1st edition*. U.S.A: Apress