Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 3, No. 8, Agustus 2019, hlm. 7990-7995 http://j-ptiik.ub.ac.id

Implementasi *2D Skeletal Based Animation Engine* Menggunakan *OpenGL*

Mokhamad Zukhruf Mifta Al Firdaus[1], Eriq Muhammad Adams Jonemaro[2], Tri Afirianto[3]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]zukhruf.mifta@student.ub.ac.id, [2]eriq.adams@ub.ac.id, [3] tri.afirianto@ub.ac.id

Abstrak

Dalam pembuatan animasi 2D, terdapat beberapa teknik yang cukup populer salah satunya adalah teknik *skeletal-based animation*. Teknik *skeletal-based animation* merupakan teknik animasi, dimana animator hanya membutuhkan potongan bagian dari karakter, kemudian disesuaikan letaknya sesuai dengan kerangka maya, teknik ini sama dengan teknik animasi pada 3D. Namun dalam perkembangannya, teknologi seperti *motion capture* maupun 3D *laser scanner* dalam pembuatan animasi 3D belum digunakan dalam pembuatan animasi 2D, terutama dengan teknik *skeletal-based animation*. Dan untuk menghasilkan animasi 2D dibutuhkan 50 hingga 300 animator untuk menggambar ribuan karakter dalam pose yang berbeda dengan menggunakan tangan, hal tersebut menyebabkan proses pembuatan animasi 2D menjadi melelahkan karena sebagian prosesnya dilakukan secara manual serta memakan waktu. Penilitian ini berfokus kepada implementasi *2D skeletal-based animation engine* menggunakan *OpenGL*. *Animation engine* yang diimplementasikan menggunakan API *OpenGL* karena *OpenGL* merupakan API yang khusus digunakan untuk pengolahan grafis, baik 2D maupun 3D. Serta menggunakan sensor *Kinect* yang merupakan salah satu teknologi *motion capture* yang tidak memerlukan marker. Hasil dari penelitian menunjukkan hasil valid pada pengujian fungsionalitas untuk komponen *AnimateFrame* serta *playImportKeyframe*, hal ini menunjukkan bahwa *animation engine* berhasil diimplementasikan dengan menggunakan API OpenGL serta sensor *Kinect* dan dapat menghasilkan animasi dengan teknik *skeletal-based animation*.

Kata kunci: *Skeletal-based Animation, Animation Engine, OpenGL, Kinect*

Abstract

In making 2D animation there are several techniques which are quite popular, one of them is skeletal-based animation technique. Skeletal-based animation is a technique where animator only needs parts form character, then adjust the location according to a virtual skeleton, this technique is the same as the animation technique in 3D. But in its development, motion capture or 3D laser scanner technology in making 3D animation have not been used for making 2D animation, especially with skeletal-based animation. And to produce 2D animation needs 50 to 300 animator to draw thousands of characters in different poses with human hands. This causes the animator mostly laborious and time-consuming in making animation. This research focuses on the implementation of 2D skeletal-based animation engine using OpenGL. Animation engine that is implemented using the OpenGL API because OpenGL is a specific API for graphics processing, both 2D and 3D. Also using a Kinect sensor which is one of motion capture technology that no need marker. The result of this research shows the valid result on functionality testing for AnimateFrame and playImportKeyframe component, it shows the animation engine successfully implemented by using OpenGL API and Kinect sensor and can produce animation with skeletal-based animation technique.

Keywords: *Skeletal-based Animation, Animation Engine, OpenGL, Kinect*

1. PENDAHULUAN

Dalam pembuatan animasi 2D, terdapat beberapa teknik yang cukup populer digunakan oleh animator, salah satunya adalah teknik *skeletal-based animation*. Pada teknik *skeletal-based animation* animator hanya membutuhkan satu gambar karakter yang telah di potong-potong sesuai dengan bagian tubuhnya dan menambahkan *skeleton* yang sesuai dengan karakter untuk membuat satu animasi (Lehtonen,

2016). Teknik ini memiliki kemiripan dengan teknik animasi pada animasi 3 dimensi (3D), yaitu menggunakan *skeleton* untuk menggerakan karakter (Dai et al., 2010).

Dalam pembuatan animasi 3D, animator menggunakan teknologi seperti *motion capture* dan *3D laser scanner*. Namun dalam pembuatan animasi 2D, animator belum menggunakan teknologi-teknologi tersebut dan untuk menghasilkan animasi 2D, dibutuhkan 50 hingga 300 animator untuk menggambar ribuan karakter dalam poses yang berbeda dengan menggunakan tangan. Hal tersebut menyebabkan animator kelelahan serta memakan waktu dalam pembuatan animasi (Pan dan Zhang, 2011).

Salah satu teknologi yang mampu menerima masukan alami dari manusia adalah teknologi Kinect. Teknologi ini banyak digunakan untuk berbagai penelitian, seperti penelitian dengan judul "*Human Motion Tracking & Evaluation using Kinect V2 Sensor*" yang menggunakan teknologi Kinect sebagai masukan untuk aplikasi rehabilitasi medis dan latihan olahraga untuk para trainee ketika pelatih tidak ada di tempat untuk secara langsung melatih para trainee (Alabbasi et al., 2015).

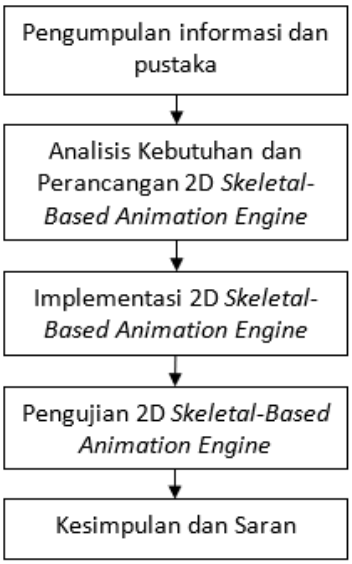
Penelitian ini akan membahas tentang mengimplementasikan 2D *skeletal-based animation engine* menggunakan *OpenGL*. Hasil yang diharapkan dari penelitian ini yaitu menghasilkan 2D *skeletal-based animation engine* yang diimplementasikan menggunakan API *OpenGL* dan menggunakan teknologi *Kinect* untuk menerima masukan dari pengguna sehingga dapat mempercepat proses pembuatan animasi 2D dengan konsep *skeletal*.

2. METODOLOGI PENELITIAN

Pada bagian ini, akan dijelaskan tahapan yang akan dilakukan dalam penelitian. Tipe penelitian dari penelitian ini adalah implementatif dengan fokus pengembangan 2D *skeletal-based animation engine*. Tahapan yang akan dilakukan dalam penelitian ini dapat dilihat pada Gambar 1 yang merupakan diagram alir dari metode penelitian pada penelitian ini.

Pengumpulan informasi dan pustaka dilakukan untuk mempelajari teori-teori dasar yang digunakan sebagai penunjang dalam penelitian. Informasi dan pustaka yang berkaitan dapat diperoleh melalui jurnal penelitian, buku, serta internet.

Pada tahap perancangan, dilakukan perancangan komponen-komponen dari *2D Skeletal-based animation engine*. Perancangan yang dilakukan adalah perancangan arsitektur dari *animation engine*, perancangan integrasi sensor *kinect* dengan *animation engine*, perancangan direktori sendi yang digunakan, dan perancangan *window workspace*.



Gambar 1. Diagram Alir Metodologi Penelitian

Pada tahap implementasi, akan dijelaskan komponen apa saja yang diimplementasikan. Untuk dapat mengimplementasikan *2D Skeletal animation engine*, diawali dengan mengimplementasikan arsitektur *animation engine*, kemudian mengintegrasikan sensor *Kinect* dengan *animation engine* serta yang terakhir adalah mengimplementasikan *window workspace*. Implementasi dari penelitian ini hanya sampai pada tahap *global pose generation* yang terdapat pada lapisan *animation pipeline*.

Pada tahap pengujian, terdapat satu skenario pengujian yaitu pengujian fungsionalitas yang menguji dua fungsi utama dari *animation engine* dengan tujuan untuk mengetahui fungsionalitas dari *animation engine*. Kemudian penarikan kesimpulan serta menambahkan saran pada tahap kesimpulan dan saran, penarikan kesimpulan dilakukan untuk dapat mengetahui inti dari penelitian, serta saran yang diharapkan dapat berguna pada penelitian serupa yang akan dilakukan di kemudian hari.

3. ANALISIS KEBUTUHAN DAN PERANCANGAN

3.1. Analisis Kebutuhan

Analisis kebutuhan dibutuhkan untuk dapat menentukan kebutuhan-kebutuhan dari *animation engine*. Kebutuhan-kebutuhan yang

dibutuhkan dari *animation engine* terdiri dari kebutuhan fungsional serta kebutuhan non-fungsional.

Pada kebutuhan fungsional, dijelaskan mengenai apa saja yang harus dapat dilakukan oleh *animation engine*. Hasil dari analisis kebutuhan fungsional terdapat pada Tabel 1.

Tabel 1. Kebutuhan Fungsional *Animation Engine*

No	Spesifikasi Kebutuhan
1	Sistem harus dapat menganimasikan *frame* n hingga *frame*
2	Sistem harus dapat menganimasikan *frame* n hingga *frame* n+1 yang datanya berasal dari luar sistem

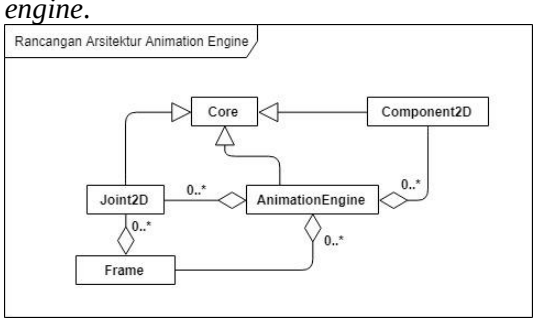
Untuk kebutuhan non-fungsional, akan dijelaskan kebutuhan-kebutuhan pendukung agar membantu kebutuhan fungsional dari *animation engine*. Hasil dari analisis kebutuhan non-fungsional terdapat pada Tabel 2.

Tabel 2. Kebutuhan Non-Fungsional *Animation Engine*

No	Definisi Kebutuhan
1	Sistem dapat mengaktifkan sensor *Kinect*
2	Sistem dapat mendapatkan informasi sendi yang ditangkap oleh sensor *Kinect*

3.2. Perancangan Arsitektur *Animation Engine*

Gambar 2 merupakan rancangan diagram kelas arsitektur dari 2D *skeletal-based animation engine*.



Gambar 2. Rancangan Kelas Diagram Arsitektur 2D *Skeletal Based Animation Engine*

Menurut Gregory (2018), arsitektur dalam *animation engine* terdiri dari 3 lapisan yaitu, *animation pipeline*, *action state machine*, dan *animation controller*. Tiap lapisan mempunyai fungsi, *animation pipeline* berfungsi untuk menganimasikan setiap karakter dan objek yang ada hingga menjadi sebuah animasi. *Action state machine* berfungsi untuk memodelkan animasi yang telah dibuat menjadi sebuah *state* dengan bantuan *finite state machine* (FSM), FSM dalam animasi disebut dengan *action state machine* (ASM). *Animation controller* berfungsi sebagai *high-level system* yang mengatur animasi. Namun dalam penelitian ini, implementasi hanya sebatas pada lapisan *animation pipeline*.

Arsitektur dari *animation engine* yang dikembangkan terdiri dari 4 kelas yaitu, kelas *Core* yang merupakan kelas dasar dari *animation engine*. Dalam kelas *Core*, dilakukan proses inisialisasi dari *SDL*, *OpenGL*, *Shader*, serta pendefinisian *engine loop*. Kemudian kelas *AnimationEngine*, yang dimana fungsi integrasi *Kinect* serta fungsi menganimasikan *keyframe* di definisikan. Kelas *Joint2D* merupakan kelas yang merepresentasikan sendi melalui titik titik yang ditampilkan pada *workspace*. Berawal dari inisialisasi titik, kemudian menampilkannya dalam *engine loop*. Dalam kelas ini juga dilakukan proses menyimpan nilai vector dari tiap sendi. Kelas *Component2D* merupakan kelas yang berfungsi untuk menampilkan komponen *sprite*. Berawal dari inisialisasi komponen, kemudian menampilkannya dalam *engine loop*.

3.3. Perancangan Integrasi Sensor *Kinect* dengan *Animation Engine*

Agar dapat menggunakan sensor *Kinect* dalam *animation engine*, diperlukan integrasi antara sensor dengan *animation engine*. Proses integrasi dilakukan dengan memanggil API dari Kinect di dalam kelas *Core*, dan mendefinisikan fungsi inisialisasi sensor *Kinect* serta fungsi deteksi sendi di dalam kelas *AnimationEngine*.

3.4. Perancangan Direktori Sendi

Sendi sendi yang digunakan, terhimpun dalam satu kelas Enumerasi dengan nama jointDirectory. Isi dari jointDirectory adalah sendi dari *Kinect* yang dipetakan sesuai dengan urutan yang ada pada Tabel 3.

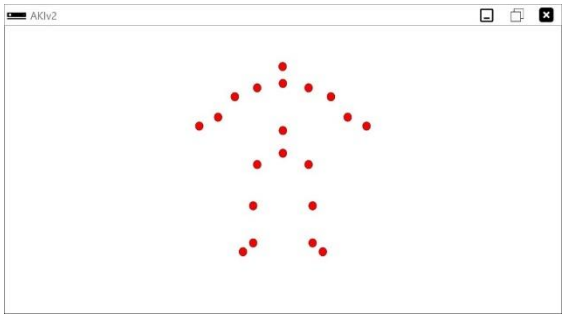
Tabel 3. Pemetaan Sendi pada *Animation Engine*

No	Sendi pada *Animation Engine*	Sendi pada *Kinect*
0	Head	JointType_Head
1	Neck	JointType_SpineShoulder
2	ShoulderLeft	JointType_ShoulderLeft
3	ElbowLeft	JointType_ElbowLeft
4	HandLeft	JointType_HandLeft
5	ShoulderRight	JointType_ShoulderRight
6	ElbowRight	JointType_ElbowRight
7	HandRight	JointType_HandRight
8	SpineBase	JointType_SpineBase
9	HipLeft	JointType_HipLeft
10	KneeLeft	JointType_KneeLeft
11	AnkleLeft	JointType_AnkleLeft
12	HipRight	JointType_HipRight
13	KneeRight	JointType_KneeRight
14	AnkleRight	JointType_AnkleRight

Tujuan dari pemetaan sendi adalah melakukan penyesuaian direktori sendi yang telah disediakan oleh *Kinect* karena tidak semua sendi digunakan pada *animation engine*.

3.5. Perancangan *Window Workspace*

Perancangan *window workspace* merupakan perancangan window utama dari *animation engine* seperti yang ada pada Gambar 3.



Gambar 3. *Mockup Animation Engine*

4. IMPLEMENTASI

4.1. Implementasi Arsitektur *Animation Engine*

Untuk dapat mengimplementasikan arsitektur dari *animation engine* dibuat kelas-kelas yang telah dirancang pada bab perancangan. Kelas-kelas tersebut adalah kelas *core*, kelas *animation engine*, kelas *Joint2D*, dan kelas *Component2D*, dimana fungsi dari masing masing kelas telah dijelaskan pada bagian perancangan.

4.2. Integrasi Sensor *Kinect* dengan *Animation Engine*

Untuk dapat mengintegrasikan sensor Kinect dengan *animation engine* agar dapat menggunakan seluruh fitur yang disediakan oleh sensor *Kinect*, diperlukan header dari API *Kinect*, *Kinect.h* pada file *Core.h*.

4.3. Implementasi Direktori Sendi

Direktori sendi yang telah dirancang pada bagian perancangan, diimplementasikan menjadi sebuah kelas enumerasi yang sesuai dengan urutan yang telah dirancang.

4.4. Implementasi *Window Workspace*

Window workspace merupakan jendela utama dari aplikasi *animation engine* ini dan untuk dapat mengimplementasikannya, dilakukan instansiasi objek window dari *library SDL*. Hasil dari implementasi *window workspace* terdapat pada Gambar 4.

Gambar 4. Hasil Implementasi *Window Workspace*

5. PENGUJIAN

5.1. Pengujian Unit Integrasi Sensor Kinect

Pengujian unit dilakukan pada *method initKinect* yang berfungsi untuk menginisialisasi sensor *Kinect*, serta *method getBodyData* yang berfungsi untuk mendeteksi sendi dari tubuh yang terdeteksi oleh sensor *Kinect*. Pengujian unit dilakukan dengan menggunakan metode *whitebox testing*. Hasil dari pengujian unit integrasi sensor *Kinect* terdapat pada Tabel 4.

Tabel 4. Hasil Pengujian Unit Integrasi Sensor *Kinect*

Nama Method	*Cyclomatic Complexity*	Jumlah Kasus Uji	Hasil
initKinect	2	2	*Pass*
getBodyData	4	4	*Pass*

5.2. Pengujian Fungsionalitas Pada *Animation Engine*

Fungsionalitas pada *animation engine* diuji dengan menggunakan metode *blackbox testing*. Pengujian ini berfungsi untuk menguji validasi dari fungsi *AnimateFrame*, dan fungsi *playImportKeyframe*. Hasil dari pengujian terdapat pada Tabel 5.

Tabel 5. Hasil Pengujian Fungsionalitas pada *Animation Engine*

No	Komponen	Kasus Uji	Hasil	Valid
1	*Animate Frame*	Menganimasikan *Frame* 0 hingga *Frame* n	Menampilkan animasi dari *Frame* 0 hingga *Frame* n	Ya
2	*playImport Keyframe*	Menganimasikan *Frame* 0 hingga *Frame* n dari file .txt yang diimpor ke dalam aplikasi	Menampilkan animasi dari *Frame* 0 hingga *Frame* n hasil dari file .txt yang telah diimpor ke dalam aplikasi	Ya

Dari pengujian ini, didapatkan hasil valid untuk komponen *AnimateFrame* yang berfungsi menganimasikan *frame* n hingga *frame* n+1 dengan memanfaatkan fungsi LERP, serta *playImportKeyframe* yang berfungsi menganimasikan *frame* n hingga *frame* n+1 dari file .txt yang diimpor ke dalam aplikasi. Hasil tersebut menunjukkan bahwa fungsionalitas dari *animation engine* telah diimplementasi serta berfungsi dengan benar.

6. KESIMPULAN DAN SARAN

6.1. Kesimpulan

Dari penelitian ini dapat ditarik kesimpulan yaitu:

1. Arsitektur dari 2D *skeletal based animation engine* terdiri dari beberapa proses, yang berawal dari proses ektraksi koordinat dari tiap sendi yang di dapat melalui sensor Kinect, kemudian menginstansiasi objek untuk masing-masing sendi. Lalu masuk pada proses penghimpunan objek sendi yang telah diinstansiasi menjadi satu objek *frame*. Kemudian melakukan proses animasi *frame* yang selanjutnya ditampilkan pada layar. Untuk dapat merealisasikan proses-proses tersebut dibutuhkan kelas *Core*, kelas *AnimationEngine*, kelas *Frame*, kelas *Joint2D*, serta kelas *Component2D*.
2. Dari hasil pengujian yang telah dilakukan, didapatkan hasil pengujian dengan nilai valid. Nilai valid yang didapatkan menunjukkan bahwa *animation engine* telah berhasil diimplementasikan dan sesuai dengan kebutuhan.

6.2. Saran

Dari penelitian yang telah dilakukan, terdapat beberapa saran yang perlu dipertimbangkan agar *animation engine* ini dapat dikembangkan lebih lanjut yaitu:

1. Penambahan window pemilihan karakter untuk karakter bertipe *non-humanoid*.
2. Menambahkan fungsi kliping agar dapat mengkombinasikan dua animasi atau lebih.
3. Menambahkan editor untuk *sprite* agar dapat memodifikasi *sprite.*

7. DAFTAR PUSTAKA

Alabbasi, H., Gradinaru, A., Moldoveanu, F. dan Moldoveanu, A., 2015. Human motion tracking & evaluation using Kinect V2 sensor. In: *2015 E-Health and Bioengineering Conference (EHB)*. [online] 2015 E-Health and Bioengineering Conference (EHB). Iasi, Romania: IEEE.pp.1–4. Tersedia melalui: IEEExplore < https://ieeexplore.ieee.org/document/7391465> [Diakses 5 Oktober 2018].

Dai, H., Cai, B., Song, J. dan Zhang, D., 2010. Skeletal Animation Based on BVH Motion Data. In: *2010 2nd International Conference on Information Engineering and Computer Science*. [online] 2010 2nd International Conference on Information Engineering and Computer Science. Wuhan, China: IEEE.p.4. Tersedia melalui: IEEExplore < https://ieeexplore.ieee.org/document/56782

92> [Diakses 5 Oktober 2018].

Gregory, J., 2018. *Game Engine Architecture*. 3rd ed. Florida: A K Peters/CRC Press.

Lehtonen, J., 2016. FROM 2D-SPRITE TO SKELETAL ANIMATIONS. *FROM 2D-SPRITE TO SKELETAL ANIMATIONS – Boosting the performance of a mobile application*, [online] Tersedia di: < https://www.theseus.fi/bitstream/handle/10024/113773/Lehtonen_Jenni.pdf > [Diakses 17 November 2018].

Pan, J. dan Zhang, J.J., 2011. Sketch-Based Skeleton-Driven 2D Animation and Motion Capture. In: Z. Pan, A.D. Cheok and W. Müller, eds. *Transactions on Edutainment VI*. [online] Berlin, Heidelberg: Springer Berlin Heidelberg.pp.164–181. Tersedia melalui: Springer < https://link.springer.com/chapter/10.1007%2F978-3-642-22639-7_17> [Diakses 15 Oktober 2018].