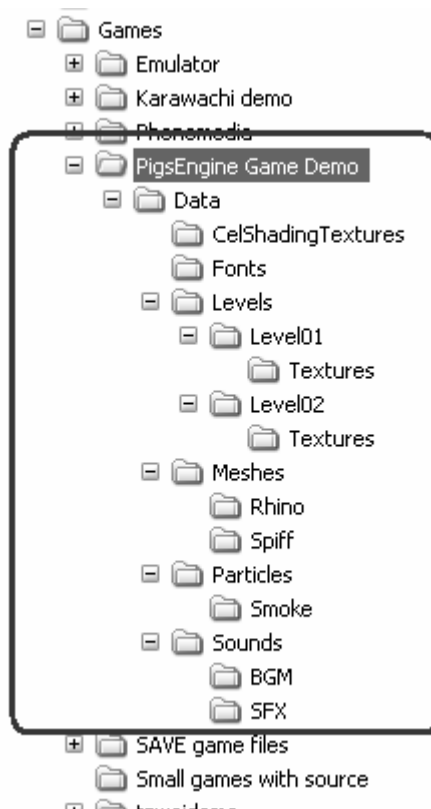


# MANUAL

## 1. Penggunaan *Engine*

Sebelum bisa menjalankan aplikasi yang dibuat dengan bantuan *engine* ini, ada beberapa hal yang perlu diperhatikan. Yang pertama adalah keberadaan *resources* yang diperlukan dalam menjalankan *game*. Pastikan *files* yang diperlukan telah tersedia dan berada dalam *path* yang benar. Contoh penyediaan *resources* ini bisa dilihat pada gambar 1.



Gambar 1. Penempatan *resources* harus pada *path* yang tepat.

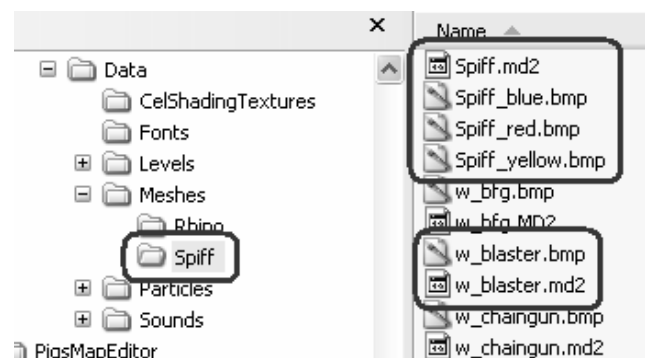
Berikutnya, pastikan terdapat *gamepad* untuk digunakan sebagai alat *input*. Pengujian dilakukan dengan menggunakan NTPAD XP con ForceFeedback 2.01. *Gamepad* harus terinstall dengan baik. Fungsi analog harus sudah

terkalibrasi. Untuk *setting* tombol, biarkan seperti semula saja. Ini bisa dilihat pada gambar 2.



Gambar 2. *Buttons mapping* tidak perlu diubah dari *default setting*-nya.

Nama *mesh files* untuk model tubuh harus sama dengan *folder* yang menampungnya, tetapi tekstur dari model tubuh bisa diberi nama apa pun. Setiap model tubuh bisa memiliki lebih dari satu tekstur dan lebih dari satu model senjata. Nama yang diberikan untuk tekstur senjata harus sama dengan dengan nama model senjata yang bersangkutan. Dengan demikian konsekuensinya adalah setiap model senjata hanya bisa memiliki satu tekstur. Lebih jelasnya, lihat gambar 3.



Gambar 3. Cara penamaan *mesh files*.

**Skripsi**

**STUDI PEMBUATAN PROTOTIPE *3D GAME ENGINE***

NRP: 26400100

Sony Tanoyo

Pembimbing I:

Ir. Djoni H Setiabudi, M.Eng.  
NIP: 85009

Pembimbing II:

Ibnu Gunawan, S.T.  
NIP: 02055

**2005**

## **2. Penggunaan Demo Aplikasi**

Dalam demo silakan mencoba kegunaan setiap tombol *joystick*. Untuk keluar tekan tombol SELECT. Bila program *crash* saat awal menjalankan karena ada *dll file* yang tidak tersedia, cobalah meng-*install* program MSVC++ .NET atau mencari *file dll* yang diminta.

## USULAN TUGAS AKHIR

### 1. Judul Tugas Akhir

Penerapan Grafik *Cel-Shading* Pada Pembuatan Prototipe *Engine Game Action-Espionage*

### 2. Latar Belakang Masalah

Sejak dulu pembajakan di Indonesia berjalan dengan sangat bebas, khususnya pembajakan *software game*. Namun belakangan ini pemerintah memperketat UU tentang hak cipta. Dalam waktu yang relatif singkat pembajakan di Indonesia mulai tidak bisa bergerak bebas. Penulis mempunyai pemikiran bahwa *developer* dan *publisher game* yang dulunya mengharamkan Indonesia pelan-pelan akan mulai melirik pasar potensial di sini. Indonesia adalah lokasi yang sangat strategis, baik sebagai *portal* distribusi, maupun sebagai pusat pengembangan *game*. Karena selain potensi pasar yang besar, tenaga kerja Indonesia tergolong murah dan berkualitas.

*Genre game* yang banyak disukai antara lain Role-Playing Game (RPG) dan *Action*. Kedua *genre* ini selain memperhatikan *storyline* dan *gameplay*, juga mengandalkan sisi grafis. Tak bisa dipungkiri bahwa bagus-tidaknya grafik dalam suatu *game* berpengaruh besar pada tingkat penjualan. Karena yang memegang kunci dalam keberhasilan marketing adalah promosi. *Storyline* dan *gameplay* jelas tidak mungkin dibocorkan, setidaknya sebagian besar harus dirahasiakan. Jadi satu-satunya yang bisa dengan bebas disebarluaskan dalam masa promosi adalah grafiknya. Dengan demikian grafik, dan

mungkin juga kualitas suara, diakui atau tidak, telah menjadi senjata vital dalam promosi.

Penggunaan gambar 2D dalam RPG dan *Action Game* di masa lalu memang lazim dan dipandang cukup. Tetapi sekarang, untuk menghasilkan kualitas yang prima, sebuah RPG atau *Action Game* dituntut untuk terus-menerus memperbaiki bidang sinematografi dalam penyuguhan cerita. Termasuk di dalamnya tata letak kamera dan sebagainya. Artinya grafis 3D adalah jalan keluar yang tepat. Tetapi bagaimana untuk *game-game* yang lebih cocok digambarkan dalam suasana 2D? Jelas bisa membatasi kreatifitas sutradara (*director*) dalam mengembangkan *game* tersebut. Karena dia tidak bisa seenaknya mengganti sudut pandang kamera. Sebab sampai saat ini nyaris tidak mungkin bagi *artist* untuk bisa menggambarkan semua objek dalam 2D dari segala sudut pandang dan ukuran. Kalau pun bisa, hal itu membutuhkan kerja keras dalam waktu yang sangat lama dan media penyimpanan yang luar biasa besar.

### 3. Perumusan Masalah dan Ruang Lingkup

Masalah yang muncul adalah:

- a. Bagaimana cara pembuatan *engine game action-espionage* pada *platform Windows* yang bisa dikembangkan lebih lanjut di kemudian hari?
- b. Bagaimana penerapan teknik *cel-shading* yang merupakan suatu cara untuk membuat gambar 3D tampil seperti 2D?
- c. Bagaimana mengimplementasikan teknik *cel-shading* ke dalam *engine game action-espionage*?

Ruang Lingkup terdiri dari:

- a. Menggunakan *platform* Windows XP yang mempunyai DirectX 9.0.
- b. Pembuatan program menggunakan Microsoft Visual C++ .NET.
- c. *Engine* yang dibuat hanya akan dites pada sistem PC AMD Athlon XP 2000+ dengan kapasitas *memory* 256 MB dan *video card* MSI MS-StarForce GeForce4 Ti 4200 saja.
- d. *Engine* yang dibuat tidak akan 100% *bug-free*, terutama (dan tidak terbatas) pada bagian *collision detection*.
- e. *Engine* yang dibuat hanya akan terbatas pada *graphic engine* dan sedikit fungsi-fungsi umum lainnya khusus untuk *game* ber-genre *action-espionage* yang menggunakan grafik *cel-shading*.
- f. *Engine* akan disertai dengan contoh *game* yang menggunakan *engine* tersebut.
- g. Contoh *game* yang disertakan pada *engine* ini hanya akan berupa demo dari seluruh kemampuan yang dimiliki oleh *engine*.
- h. Kemampuan *engine* yang direncanakan antara lain:
  - Inisialisasi DirectX  
 Inisialisasi ini meliputi DirectX Graphics (Direct3D), DirectInput dan DirectSound. Khusus untuk DirectSound, akan diinisialisasi secara transparan dengan menggunakan tool FMod.
  - Fungsi Kamera  
 Kamera adalah ‘mata’ pada lingkungan aplikasi 3D. Kameralah yang menentukan sudut pandang yang akan ditampilkan pada monitor. Fungsi kamera yang akan dibuat adalah pandangan orang

ke-3 dengan posisi di atas karakter utama, namun pada keadaan tertentu bisa berpindah untuk memudahkan pemain melihat keadaan sekitarnya.

- Fungsi *cel-shading* yang meliputi *sharp-lighting* dan *outline*  
*User* dari *engine* ini (dalam hal ini *programmer*) bisa memilih untuk mengaktifkan *feature sharp-lighting* dan *outline*.
- *Collision detection*  
*Collision detection* adalah pengecekan tumbukan yang terjadi pada *game*. Hal ini dilakukan untuk mencegah hal-hal yang tidak diinginkan, misalnya karakter utama berjalan menembus tembok.

#### 4. Tujuan Tugas Akhir

Tujuan akhir dari tugas akhir ini adalah untuk membuat sebuah *engine* sederhana dari *game* ber-genre *action-espionage* yang menerapkan grafik *cel-shading* dengan menggunakan Visual C++ dan Microsoft DirectX 9.0 SDK.

#### 5. Tinjauan Pustaka

*Engine* ini merupakan *engine* untuk *game action-espionage* yang menampilkan grafik 3D dengan menggunakan teknik *cel-shading*. Maksudnya adalah untuk memberikan nuansa 2D atau kartun pada *game* tersebut. Salah satu contoh *game* yang menggunakan teknik ini adalah Dragon Quest VIII™ seperti yang terlihat pada gambar 1 di bawah ini.



Gambar 1. Dragon Quest VIII™ oleh SquareEnix™

Secara singkat teknik *cel-shading* dibagi menjadi dua, yaitu pembuatan *outline*, yakni garis tepi hitam yang biasa ada pada gambar kartun, dan pengelompokan beberapa warna ke dalam blok-blok warna tertentu, sehingga jenis warna yang ada menjadi lebih terbatas tetapi dalam distribusi tertentu. Beberapa orang menyebutnya *sharp-lighting*.



Gambar 2. Contoh blok warna dalam *greyscale*



Gambar 3. Breath of Fire V<sup>TM</sup> oleh Capcom<sup>TM</sup> menggunakan *outline* saja



Gambar 4. Legend of Zelda<sup>TM</sup> oleh Nintendo<sup>TM</sup> menggunakan *sharp-lighting* saja

Gambar 3 di atas menunjukkan teknik *cel-shading* yang terbatas pada penggunaan *outline* saja. Sedangkan pada gambar 4, penggambaran karakter tidak menggunakan *outline* sama sekali, tetapi warnanya terpisah dengan kentara, tidak berupa gradasi sempurna seperti layaknya gambar 3 dimensi biasa. Inilah yang disebut *sharp-lighting*.



Gambar 5. Jet Set Radio Future™ oleh Sega™ menggabungkan kedua teknik

Gambar 5 menampilkan sebuah *scene* atau *screen-shot* dari *game* yang menggunakan penggabungan *sharp-lighting* dan *outline*.

Software yang akan digunakan dalam pembuatan *engine* ini ialah Microsoft Visual C++ .NET. Alasannya adalah karena C++ merupakan bahasa pemrograman *low-level* yang memberikan banyak kebebasan bagi *programmer* dalam membuat program. Selain itu *platform* yang akan digunakan adalah Windows XP dengan DirectX 9.0, dan compiler C++ yang paling kompatibel dengan *platform* tersebut adalah Microsoft Visual C++. Pemilihan jatuh pada Visual Studio .NET dan bukan Visual Studio 6 karena ada struktur DirectX 9.0 yang tidak didukung oleh Visual Studio 6, misalnya D3DXMATRIXA16.

## 6. Metodologi Penelitian

Rencana penelitian dilakukan sesuai dengan langkah-langkah berikut:

- a. Studi Literatur

Manfaat dari *engine* ini adalah untuk memudahkan proses pembuatan *game action-espionage* khususnya yang menggunakan grafis *cel-shading*.

[illegible]

## 9. Daftar Pustaka

LaMothe, Andre. (2002). Trick of the Windows *Game* Programming Gurus

2<sup>nd</sup> edition. Sams Publishing.

Luna, Frank D. (2003) Introduction to 3D Game Programming with DirectX

9.0. Wordware Publishing.

Walsh, Peter. & Perez, Adrlan. (2003). Advanced 3D Game Programming

with DirectX 9.0. Wordware Publishing.

Dalmau, Daniel Sanchez-Crespo. (2003). Core Techniques and Algorithms

in Game Programming. New Riders.

Game Development, <http://gamedev.net>

Game Tutorials, <http://gametutorials.com>

Andypike, <http://andypike.com>

PhenomenA Games, <http://phenomenagames.no-ip.org>

IGN.com, <http://www.ign.com>