ISSN : 2302-3805

MANYLIGTHS PROJECT UNTUK PENGUJIAN KUALITAS GRAFIK KOMPUTER PADA VGA CARD

Robby Candra
Sistem Komputer Universitas Gunadarma
Jl. Margonda Raya 100 Pondok Cina Depok – Jawa Barat
email : robby.c@staff.gunadarma.ac.id

Abstrak

Kebutuhan akan kualitas grafik komputer yang besar saat ini sangat diperlukan mengingat penggunaan aplikasi komputer yang semakin berkembang untuk berbagaimacam aplikasi komputer, terutama untuk aplikasi desain grafis maupun game. Penggunaan dari grafik komputer tersebut diaplikasikan pada perangkat Video Graphics Adapter (VGA). Untuk mengetahui kualitas dari video grafik tersebut digunakan aplikasi Manylights yang dibuat menggunakan OpenGL. Metode yang digunakan yaitu dengan perhitungan dan pengaturan pencahayaan yang dapat diprogram secara fleksibel dengan menggunakan vertex program. Vertex program ini yang banyak digunakan dan banyak dipakai dalam pembuatan program grafik komputer. Dengan menggunakan Manylights Project kualitas tampilan dapat diketahui dengan cara melihat tampilan pergerakan cahaya, semakin halus pergerakan tampilan cahaya maka semakin bagus pula kualitas grafik komputer tersebut.

Kata kunci :
Grafik Komputer, Manylights, OpenGL, VGA

1. Pendahuluan

Kebutuhan akan kualitas grafik komputer yang besar saat ini sangat diperlukan mengingat penggunaan aplikasi komputer yang semakin berkembang untuk berbagaimacam aplikasi komputer, terutama untuk aplikasi desain grafis maupun game. Saat ini komputer grafik sangat pesat perkembangannya di dalam teknologi informasi yang diaplikasikan pada perangkat Video Graphics Adapter (VGA). Apalagi dengan terus berkembangnya sejumlah hardware untuk video grafik yang dikeluarkan oleh berbagai macam vendor yang terus meningkatkan kualitasnya di dalam grafik.Sebelum vendor pembuat video grafik menjual produknya, terlebih dahulu produk yang sudah dihasilkan diujicoba terlebih dahulu untuk mengetahui kualitas dari video grafik tersebut.

Salah satu cara untuk menguji coba dari video grafik tersebut yaitu menggunakan aplikasi Many Lights Project. Proyek Manylights ini merupakan salah satu dari proyek yang dibuat dengan menggunakan openGL dan menggunakan beberapa metode yang diperlukan. Project Many Ligthts ini menampilkan sebuah bentuk tunggal segi empat yang tidak ada jarak dan tidak tumpang tindih dengan 60 titik cahaya yang bergerak diatasnya [4].

Pada aplikasi Many Lights ini pencahayaan segi empat tersebut menggunakan tiga metode. Metode pertama adalah dengan menggunakan fixed function pipeline dengan melewati delapan pipeline yang diperlukan dan setiap pipeline digunakan delapan lampu standar OpenGL. Metode kedua adalah dengan menggunakan vertex program. vertex program dalah program kecil yang mengeksploitasi kemungkinan pada chip baru untuk menggunakan sebagian dari set instruksi chip secara langsung. Pada tahap ini akan dilakukan pengaturan pencahayaan. Dan metode ketiga adalah dengan menggunakan vertex program2. Program ini akan memanggil fungsi dalam vertex program sebelumnya dan melakukan perhitungan pada pencahayaan lampu.

Untuk mendapatkan kualitas kartu grafik komputer yang sesuai dengan kebutuhan maka diperlukan pengetahuan untuk mengetahui kualitas kartu grafik komputer tersebut. Metode yang dapat digunakan untuk menguji kualitas dari kartu grafik komputer yaitu Manylights Project. Dengan metode ini para pengguna kartu grafik komputer dapat memilih dengan tepat kartu grafik komputer yang akan digunakan sesuai dengan aplikasi yang dipakai.

2. Tinjauan Pustaka

2.1. Video Graphics Adapter (VGA)

VGA singkatan dari Video Graphics Adapter adalah standar tampilan komputer analogyang dipasarkan pertama kali oleh IBM pada tahun 1987. VGA merupakan standar grafis terakhir yang diikuti oleh mayoritas pabrik pembuat kartu grafis komputer. Tampilan Windows sampai sekarang masih menggunakan modus VGA karena didukung oleh banyak produsen monitor dan kartu grafis. Kartu VGA (Video Graphic Adapter) berguna untuk menerjemahkan output(keluaran) komputer ke monitor [7]. Untuk menggambar / design graphic ataupun untuk bermaingame.VGA Card sering juga disebut Card display, kartu VGA atau kartu grafis.

Gambar yang anda lihat pada monitor tercipta dari titik-titik kecil yang disebut pixel. Biasa pada setting resolusi, layar display terdiri atas ber milyard-milyard pixel, dan komputer bertugas menyusunnya untuk menciptakan sebuah gambar atau citra. Komponen dasar dari sebuah kartu grafis yang terkoneksi pada motherboard dan monitor, processor dan memory. Sebuah kartu grafis bekerja pada prinsip dasar yang sama. CPU, bekerja dengan rangkaian software aplikasi,

ISSN : 2302-3805

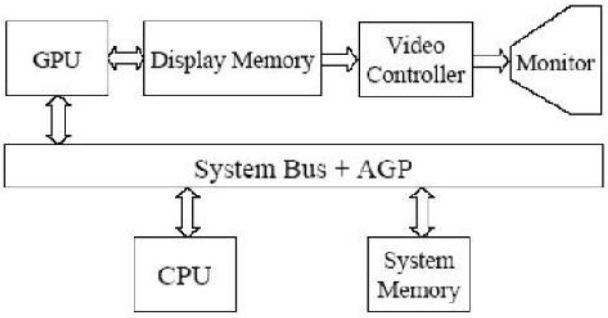
mengirimkan informasi gambar ke kartu grafis. Kemudian kartu grafis menentukan bagaimana cara mengolah pixel-pixel tersebut ke layar untuk menciptakan sebuah gambar. Lalu mengirimkan informasi itu ke layar monitor melalui kabel.

Sebuah Kartu Grafis dapat menyelesaikan tugas ini untuk empat komponen utama, yaitu :

1. Koneksi Motherboard untuk data dan power.
2. Processor untuk memutuskan apa yang harus dikerjakan untuk tiap pixel yang ada dilayar
3. Memori untuk menyimpan informasi tentang masing-masing pixel dan sebagai tempat menyimpan sementara gambar-gambar yg telah selesai diproses.
4. Koneksi ke monitor untuk melihat hasil akhir.

Komponen-Komponen VGA Card :

1. GPU (Graphic Processing Unit) GPU adalah prosesor dari sebuah video card, dan berfungsi untuk pengolahan data gambar yang akan ditampilkan di layer monitor.
2. Video Memory Berfungsi sebagai tempat penyimpanan sementara sebelum dan sesudah pemrosesan data pada GPU.
3. RAMDAC (Random Access Memory Digital – Analog Converter) Berfungsi mengubah gambar digital menjadi sinyal analog agar bisa digunakan oleh monitor.
4. Bus Interface Berfungsi menghubungkan motherboard dengan kartu grafis. Pada umumnya, bus interface ini tipe AGP dan PCI-Express.
5. Display Interface Berfungsi menghubungkan kartu grafis dengan monitor. Umumnya terdapat 3 port display, antara lain DVI, VGA, TV-Out
6. Heatsink dan Fan Berfungsi sebagai pendingin kartu grafis



Gambar 1. Arsitektur Sistem Grafik Komputer [7]

2.2. OpenGL

OpenGL adalah program aplikasi interface yang digunakan untuk mende_nisikan komputer grafis 2D dan 3D. Program platform API ini umum nya untuk menetapkan standar dalam industry komputer pada jenis interaksi komputer grafis 2D dan juga menjadi alat yang biasa digunakan dengan grafis 3D juga. Fungsi dasar dari OpenGL adalah untuk mengeluarkan koleksi khusus dari executable atau perintah ke sistem operasi. Dengan demikian, program ini bekerja dengan perangkat keras grafis yang sudah ada yang berada pada hard drive atau sumber tertentu lainnya. Setiap perintah di set rancang untuk melibatkan tindakan gambar tertentu, atau meluncurkan efek khusus tertentu yang terkait dengan grafis.[5]

Cara membuat perintah dalam OpenGL dapat dilakukan dalam dua cara berbeda. Pertama, adalah programmer membuat dan menyimpan daftar perintah yang digunakan secara berulang. Ini adalah salah satu cara yang lebih rutin digunakan program antarmuka. Seiring dengan perkembangan maka dibuat kelompok perintah yang lebih permanen, juga memungkinkan untuk membuat dan menjalankan salah satu perintah dalam perimeter waktu dari grafis komputer juga.

Seiring dengan kemampuan melakukan antarmuka dengan sistem operasi, memberi manfaat perangkat keras grafis, OpenGL juga menyediakan beberapa protokol built-in yang mungkin berguna bagi pengguna akhir. Di antara fitur ini alat seperti alpha blending, pemetaan tekstur, efek atmosfer, dan surface removal. Alat ini menyesuaikan diri dengan sistem operasi yang sedang digunakan.[5]

OpenGL (Open Graphics Library) adalah spesifikasi standar yang mende_nisikan sebuah ross-bahasa, cross-platform API untuk menulis aplikasi yang menghasilkan komputer 2D dan 3D grafis. Terdiri dari lebih dari 250 panggilan fungsi yang berbeda yang dapat digunakan untuk menggambar tiga dimensi yang kompleks adegan-adegan dari primitif sederhana. OpenGL dikembangkan oleh Silicon Graphics Inc (SGI) pada tahun 1992 dan secara luas digunakan dalam CAD, Virtual Reality, Visualisasi Ilmiah, Visualisasi Informasi, dan Simulasi Penerbangan. OpenGL diterima menjadi salah satu bakuan (standard) dalam grafik akomputer dan
saat ini telah diimplementasikan dalam berbagai sistem komputer.

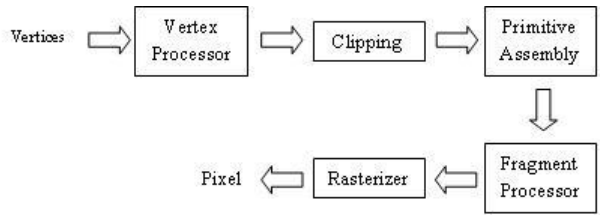
OpenGL merupakan pustaka program (program library) yang menyediakan sejumlah perintah yang berhubungan dengan grafik. OpenGL adalah API software untuk hardware grafis, dirancang sebagai antarmuka, efisien hardware independen untuk diterapkan pada banyak platform hardware yang berbeda Intuitif antarmuka, prosedural dengan C mengikat Tidak ada perintah windowing dan tidakada perintah tingkat tinggi untuk menggambarkan model objek tiga dimensi. OpenGL Utility Library (GLU) menyediakan banyak fitur pemodelan, seperti permukaan quadric dan NURBS Curves dan permukaan.[6]

3. Metode Penelitian

MANYLIGHTS merupakan sebuah program yang dibuat dengan menggunakan OpenGL. Didalam program ini akan ditampilkan sebuah bentuk tunggal segi empat, dengan 60 titik cahaya yang bergerak di atasnya. Pencahayaan segi empat tersebut menggunakan tiga metode. Metode pertama adalah menggunakan fixed function pipeline dengan melewati delapan pipeline yang diperlukan dan di setiap pipeline digunakan delapan lampu standar OpenGL. Metode kedua dengan menggunakan vertex program (NV vertex program), metode ketiga adalah menggunakan NV vertex

ISSN : 2302-3805

program2. Adapun alur yang digunakan seperti terlihat pada gambar 2 berikut ini.



Gambar 2 Diagram Alur Proses

Gambar 3 Manylights

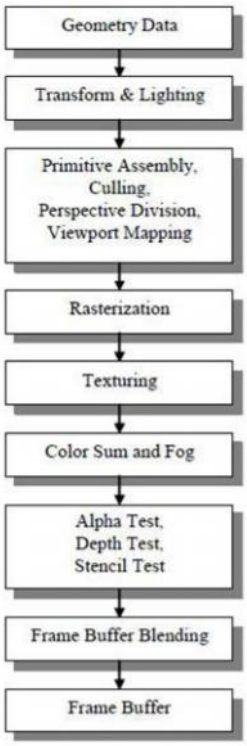
4. Hasil dan Pembahasan

4.1. Fixed Function Pipeline

Metode pertama yang dilakukan adalah dengan menggunakan fixed function pipeline. Fixed function pipeline(FFP) merupakan sebuahcpipa rendering yang menggunakan perangkat-algoritma yang disediakan untuk rendering, dengan aplikasi memodi_kasi sejumlah faktor. Hasil dari fungsi-tetap adalah Per-Vertex Lighting.Metode ini banyak digunakan pada 3D chip vendors,sebagaimana yang dijelaskan pada buku "Fixed function pipeline using vertex programs"karangan Lars Ivar Igesund and Mads Henrik Stavang.

Pada awal mula pemrograman Direct3D (sebelum DirectX 8), satu-satunya cara untuk menampilkan gra_k 3D pada layar komputer adalah menggunakan fixed function pipeline. Pada fixed function pipeline, pemrosesan vertex dan pixel menghasilkan sejumlah fungsi untuk memanipulasi operasi vertex dan pixel. Fungsi ini tidak dapat diubah, fungsi-fungsi tersebut telah didefinisikan sebelumnya atau tetap, oleh karena itu dinamakan 'fixed function pipeline'. 3D chip menggunakan pipeline ketika input ialah simpul-simpul dari objek yang akan ditampilkan dan outputnya adalah frame yang akan ditunjukkan di monitor. Pipeline tersebut dapat dibagi menjadi beberapa langkah. Jika vertex program digunakan, beberapa langkah pada pipeline dapat terlewati, sehingga memberikan aplikasi yang lebih fleksibel bagi programmer.

Pipeline di atas sering disebut fixed function pipeline (FFP), dan memiliki fungsi yang jelas yang didefinisikan oleh standar yang telah di tetapkan biasanya oleh DirectX atau OpenGL. 3D chip sendiri dapat dibuat/dikonstruksikan dengan berbagai cara, selama driver meyakinkan hasil akhirnya berkorespondensi dengan definisi standar yang digunakan. Setiap pembuat chip memiliki solusi yang berbeda-beda didalam chip dan drivernya, tetapi chip harus mendukung sketsa umum pipeline pada tabel di atas. Hanya fase transformasi dan pencahayaan yang dilewati ketika vertex program digunakan. Untuk membuat proyek many lights ini fase fixed function pipeline di atas harus dilewati sebanyak 8 kali dan di setiap fasenya menggunakan 8 pencahayaan standar OpenGL.



Gambar 4 Blok Diagram Fixed Function Pipeline

4.2. Vertex Program

Metode kedua yang dilakukan adalah menggunakan vertex program. Dalam buku karangan Martin Ecker "XEngine Programmable Graphics Pipeline Architectures" dijelaskan bahwa vertex program merupakan ekstensi OpenGL yang mendefinisikan sebuah lingkungan eksekusi shader vertex dengan sebuah bahasa naungan tingkat rendah yang menyertai. Dalam paper Chris J. Thompson "Using modern graphics Architectures for general-Purpose Computing : A framework and Analysis" dijelaskan bahwa vertex program menghitung fungsi.

Vertex program tidak dapat mengakses memori utama secara langsung, tetapi mereka dapat membaca nilai dari blok 96 konstan register yang dapat digunakan untuk melewatkan informasi ke vertex program. Vertex program akan mengambil fungsi dari fixed function pipeline. Vertex program adalah program kecil yang mengeksploitasi kemampuan dari chip untuk menggunakan bagian dari set instruksi chip secara

ISSN : 2302-3805

langsung. Kebanyakan chip baru yang muncul pada tahun 2002, sudah dapat mendukung untuk program semacam ini. Istilah shader digunakan untuk program ini, tetapi kebanyakan menyesatkan seperti instruksi yang tersedia yang digunakan untuk membuat program kecil yang belum tentu bayangan simpul (walaupun hal itu memungkinkan). 3D chip ini dapat diprogram, berbeda dengan fixed function pipeline.

Chip ini dapat diprogram secara manual untuk melakukan rutinitas khusus, termasuk pekerjaan-pekerjaan yang ada pada fixed function pipeline. Hal tersebut membuat chip lebih fleksibel. Pembuat program yang menginginkan membuat program untuk efek grafis spesial sekarang memiliki kemungkinan yang lebih luas dalam mengembangkannya melewati set instruksi yang tersedia. Jika chip yang digunakan merupakan chip yang tidak dapat diprogram, pembuat program akan memaksa untuk menggunakan fixed function pipeline yang ada atau menggunakan driver perangkat agar chip dapat meniru kemampuan memprogram menggunakan CPU.
Kemungkinan lain adalah bagi pembuat program aplikasi untuk memprogram fungsi yang dibutuhkan dengan menggunakan CPU. Sejak instruksi-instruksi tersebut diperkenalkan oleh chip-chip baru, chip-chip tersebut disesuaikan untuk penggunaan grafis dan menghasilkan keluaran tinggi dari data geometri, hal tersebut akan menjadi pilihan yang baik ketika bekerja dengan efek grafis. Contoh Vertex Program Manylights Bagian ini menyajikan contoh kode simpul program. Program vertex mengimplementasikan sebuah ambien sederhana, specular, dan menyebar tak terbatas perhitungan pencahayaan dengan cahaya tunggal dan mata-space normal.

```
# Vertex Program for point lights,
using vp1.
# Handles 20 point lights
# c[0]-c[3] modelviewProjection
# matrix
# c[4] ambient brightness
# c[5] light 0 position
# c[6] light 0 color
# c[7,8] similar for light 1
# ..
# c[43, 44] similar for light 19

# compute position
DP4 o[HPOS].x, c[0], v[OPOS];
DP4 o[HPOS].y, c[1], v[OPOS];
DP4 o[HPOS].z, c[2], v[OPOS];
DP4 o[HPOS].w, c[3], v[OPOS];

#Output decal texture coords
MOV o[TEX0].xy, v[TEX0];

#Begin color with ambient
#contribution
MOV R0, c[4];
#Compute light0 contribution

#Calculate point to light vector
ADD R1, c[5], -v[OPOS];

#R2 = Normalize R1
DP3 R2.w, R1, R1;
RSQ R2.w, R2.w;
MUL R2.xyz, R1, R2.w;

#Dot this with the normal
DP3 R1, R2, v[NRML];

#Modulate by color and add to R0
MAD R0, R1, c[6], R0;
```

4.3. NV Vertex Program

Metode ketiga yang dilakukan adalah vertex program2. Metode ini merupakan kelanjutan dari metode vertex program. Dalam buku karangan Martin Ecker "XEngine Programmable Graphics Pipeline Architectures" dijelaskan bahwa metode ini melakukan perpanjangan lingkungan eksekusi dari vertex program sebelumnya. NV vertex program adalah ekstensi OpenGL yang mende_nisikan vertex shader execution environment dengan bahasa shading tingkat rendah yang menyertainya. Awalan NV mengindikasikan bahwa ekstensi yang ada dikembangkan oleh vendor perangkat keras kartu grafis NVIDIA.

Pada saat publikasi ini dikeluarkan,ekstensi NV vertex program tersedia pada semua kartu grafis NVIDIA seri GeForce, kartu grafis Matrox Parhelia, kartu 3Dlabs terbaru dengan P10 GPU, dan Mesa, OpenGL terlihat seperti perangkat lunak untuk melakukan rendering, versi 4.1 ke atas. Eksekusi lingkungan dari NV vertex program pada dasarnya sama dengan lingkungan dari DirectX 8 vertex shaders dan tidak terkomputasi lebih baik. Hal tersebut dapat dilihat sebagai kesamaan DirectX 8 vertex shaders dengan OpenGL. Kecuali untuk pasangan instruksi baru yang dapat meniru dengan menggunakan instruksi lain pada Direct3D.

Pengubah register yang sama, seperti tujuan register mask, sumber register negation, dan sumber register sqizzle mask telah mendukung. Begitu juga jumlah dari input, output, temporary, address, dan parameter registers yang tersedia sama untuk eksekusi lingkungan Direct3D 8 vertex shader. Secara sintaks, mnemonics yang digunakan dalam NV vertex program adalah Upper-Case sebagai lawan dari Lower-case, semua instruksi harus diakhiri oleh sebuah semicolon dan untuk nama input dan output register terspesifikasi menggunakan sintaks index array. Tiga instruksi tambahan yang diberikan NV vertex program melalui bahasa Direct3D 8 vertex shading terlihat pada tabel 1.

Tabel 1 Tiga instruksi tambahan NV vertex program

ISSN : 2302-3805

Opcode	Arity	Description	Example
ABS	Unary	Assigns the component-wise absolute value of the source vector to the destination register.	`ABS o[HPOS], c[1]`
DPH	Binary	Calculates the four-component dot product of the two source vectors assuming, however, that the fourth component of the first source vector is 1.0. The result is replicated to all four components of the destination register.	`DPH R1, R0, c[0]`
RCC	Unary	Calculates the reciprocal value of the source scalar and clamps the result to the range $[2^{-64},2^{64}]$, if the reciprocal value is positive, or $[-2^{64},-2^{-64}]$ otherwise. The reason for this clamping is to keep a certain amount of floating-point precision for subsequent calculations.	`RCC R0.x, R0.x`

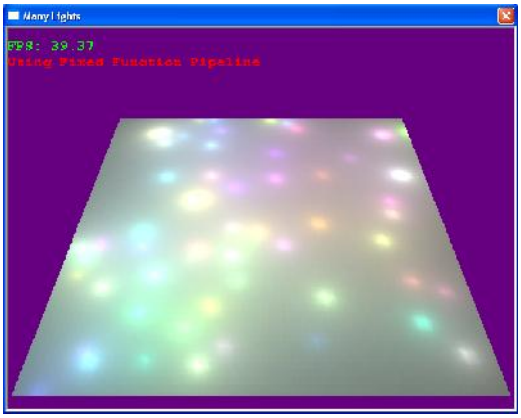
Ekstensi OpenGL NV vertex program2 [Kilg02b] [Kilg02c] NVIDIA memperkenalkan perluasan lingkungan eksekusi untuk ekstensi NV vertex program. Saat ini hanya tersedia pada generasi terbaru GPU NVIDIA GeForce FX. Ekstensi baru ini menawarkan sesuatu yang baru, instruksi yang sangat kuat, seperti Dynamic bercabang, perulangan dan subrutin panggilan. Juga sinus dan kosinus, presisi tinggi eksponensial dan logaritma, dan beberapa instruksi lainnya yang telah ditambahkan, yang bisa digunakan. Sebagian besar akan ditiru menggunakan beberapa instruksi dalam versi sebelumnya bahasa.

Jumlah maksimum dari instruksi per shader telah dijadikan dua kali lipat ke 256. Jumlah register parameter telah ditingkatkan dari 96 menjadi 256. Feature-wise, ekstensi sesuai dengan vertex shading 9 Direct3D versi bahasa 2.x. Sintaxwise bahasa yang sedikit berbeda. Sebagai contoh, label di NV vertex program2 dinyatakan dengan menetapkan suatu pengenal diikuti oleh titik dua, sedangkan di titik 9 Direct3D label shading bahasa dideklarasikan menggunakan label pseudo-instruksi. Selanjutnya, dalam Direct3D 9 hanya forward calls diperbolehkan, yaitu, label harus dideklarasikan setelah semua instruksi cabang atau panggilan referensi bahwa label. NV vertex program2 tidak memiliki pembatasan seperti itu. Terlepas dari perbedaan-perbedaan sintaksis, kedua bahasa tersebut ter-komputasi dan sama kuat Berikut ini contoh dari NV vertex program untuk Manylights :

```
!!VP2.0
#Vertex Program for point lights,
# using vp2. Handles 60 point
#lights via a loop
# c[0]-c[3] modelviewProjection matrix
# c[4] ambient brightness
# c[5] light 0 position
# c[6] light 0 color
# c[7,8] similar for light 1
# ...
# c[123, 124] similar for light 59
# c[125] - 60.0, - Number of lights
# 5.0, - Base for light data
# -1.0, - loop counter
# 2.0 - stride for
#light data

# compute position
DP4 o[HPOS].x, c[0], v[OPOS];
DP4 o[HPOS].y, c[1], v[OPOS];
DP4 o[HPOS].z, c[2], v[OPOS];
DP4 o[HPOS].w, c[3], v[OPOS];

#Output decal texture coords
MOV o[TEX0].xy, v[TEX0];

#Begin color with ambient
# contribution
MOV R0, c[4];

#Load address register with c[125],
# updating the condition register
ARLC A1, c[125];

#Loop through lights and
#calculate their contribution
startloop: #start loop

CAL lightCalculation (GT.x);
#Call "lightCalculation" if condition
#register.x>0.0

ARAC A1.xy, A1;
#A1.x=A1.x-1, A1.y+=2.
#Condition register=A1

BRA startloop (GT.x);
#Return to startloop if condition
#register.x>0.0

endloop:

#Output Color
MOV o[COL0], R0;

#Subroutine "lightCalculation"
#Compute the
#contribution from 1 light
#A1.y points to the
# start of the
#data for the light
#(eg 5 for light0)

lightCalculation:

#Calculate point to light
#vector
ADD R1, c[A1.y], -v[OPOS];

#R2 = Normalize R1
DP3 R2.w, R1, R1;
RSQ R2.w, R2.w;
MUL R2.xyz, R1, R2.w;

#Dot this with the normal
DP3 R1, R2, v[NRML];

#Modulate by color and add to R0
MAD R0, R1, c[A1.y+1], R0;

#return
RET;
```

ISSN : 2302-3805

END



Gambar 5 Tampilan pengujian resolusi tampilan grafik layar komputer menggunakan manylights project

5. Kesimpulan dan Saran

Programabilitas GPU telah meningkat pesat dalam beberapa tahun terakhir dan akan mungkin terus melakukan peningkatan selama beberapa waktu ke depan.Dari ketiga metode yang digunakan dalam pembuatan proyek manylights, vertex program adalah metode yang banyak digunakan dan banyak dipakai dalam pembuatan sebuah program grafik komputer. Berdasarkan hasil pengujian menggunakan manylights project dapat dilihat kehalusan resolusi dari grafik komputer yang dihasilkan. Cara lain untuk menguji kualitas dari grafik komputer yaitu dengan melakukan benchmarking.

Daftar Pustaka

[1] Chris J Thompson, Sahngyun Hahn, M. O., *Using modern graphics architectures for general-purpose computing : A framework and analysis*, Department of Computer Science and Engineering University of Washington
[2] Ecker, M., 2002, *Programmable graphics pipeline architectures*. XEngine Corporation.
[3] Lars Ivar Igesund, M. H. S., 2002, *Fixed function pipeline using vertex program*, NTNU Norwegian University of Science and Technology.
[4] Many, *Opengl manylights.* http://www.paulsprojects.net/opengl/manylights/manylights.html, 9 Desember 2011
[5] Qbonk, *Apa itu opengl dan fungsinya*. http://www.qbonk.net/apa-itu-opengl-dan-fungsinya.html, 25 Juli 2012
[6] Shreiner, D, 2009, Opengl programming guide seventh edition : *The oficial guide to learning opengl, version 3.0 and 3.1 addison wesley.*
[7] VGA, *Prinsip kerja vga card*, http://blakbin.blogspot.com/2010/03/prinsip-kerja-vga-card.html, 25 Juli 2012

Biodata Penulis

Robby Candra, memperoleh gelar Sarjana Komputer (SKom), Jurusan Sistem Komputer Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Gunadarma, lulus tahun 1998. Tahun 2007 memperoleh gelar Magister Teknik (MT) dari Program Magister Teknik Elektro Jurusan Elektronika Telekomunikasi Universitas Gunadarma.