JURNAL RESISTOR
Vol. 1 No 1 – April 2018
p-ISSN 2598-7542 (Print), e-ISSN 2598-9650 (Online)
Available Online at :
http://jurnal.stiki-indonesia.ac.id/index.php/jurnalresistor

ANALISIS *RENDERING* VIDEO ANIMASI 3D MENGGUNAKAN APLIKASI *BLENDER* BERBASIS *NETWORK RENDER*

I Nyoman Buda Hartawan[1], Ayu Manik Dirgayusari[2]

[1,2] Program Studi Sistem Komputer, STMIK STIKOM Indonesia
Denpasar, Indonesia

e-mail: buda.hartawan@gmail.com[1], manik.dirgayusari@gmail.com[2]

Received : February, 2018	Accepted : April, 2018	Published : April, 2018

Abstrak

Rendering merupakan sebuah proses untuk menghasilkan sebuah citra 2D atau 3D. Penggunaan spesifikasi komputer yang tinggi pada proses rendering mendorong perlunya penyediaan super computer. Pada penelitian ini dilakukan analisis rendering video animasi 3D menggunakan aplikasi blender berbasis network render. Pada penelitian ini digunakan metode penelitian eksperimen, yaitu dengan memberikan variasi nilai frame untuk mengetahui waktu eksekusi saat dilakukan proses rendering video animasi 3D. Pada setiap komputer dalam workstation dilakukan konfigurasi aplikasi Blender, sebagai Master, Slave, dan Client. Hasil penelitian menunjukkan proses rendering video animasi 3D mengalami peningkatan waktu eksekusi seiring dengan peningkatan jumlah frame, namun seiring dengan peningkatan jumlah komputer Slave waktu eksekusi proses rendering video animasi 3D mengalami penurunan.

Kata Kunci: *blender, animasi, super computer, network render, rendering*

Abstract

Rendering is a process for generating a 2D or 3D image. The use of high computer specifications in the rendering process encourages the need for super computer provision. In this research is done rendering video 3D animation analysis using blender application based on network render. In this research used experimental research method, that is by giving variation of frame value to know execution time when done rendering process of 3D animation video. On each computer in the workstation is done Blender application configuration, as Master, Slave, and Client. The results show that 3D animation video rendering process has increased execution time along with increasing number of frames, but along with increasing number of Slave computer execution time of video rendering process of 3D animation has decreased.

Keywords: *blender, animation, super computer, network render, rendering*

1. PENDAHULUAN

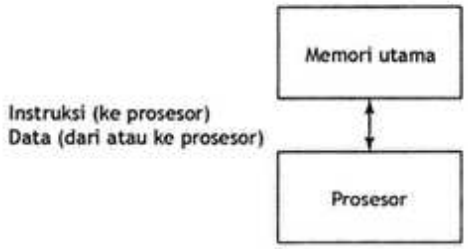
Dewasa ini perkembangan teknologi informasi semakin cepat, hal ini sejalan dengan peningkatan pemanfaatan teknologi informasi diberbagai bidang. Salah satu pemanfaatan teknologi informasi adalah dalam mendukung pengolahan citra dalam bidang desain dan multimedia. Salah satu proses yang umumnya menggunakan spesifikasi komputer tinggi adalah *rendering*.

Rendering merupakan sebuah proses untuk menghasilkan sebuah citra 2D dari data 3D. Proses ini bertujuan untuk untuk memberikan

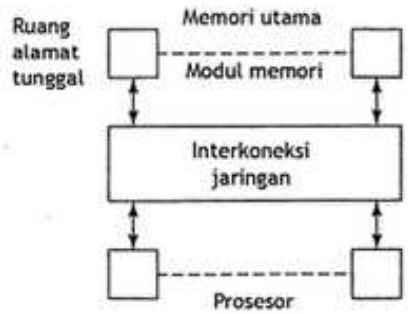
visualisasi pada *user* mengenai data 3D tersebut melalui monitor atau pencetak yang hanya dapat menampilkan data 2D. Untuk melakukan proses *rendering* dibutuhkan komputer dengan spesifikasi yang tinggi. Semakin tinggi spesifikasi komputer yang digunakan, maka semakin cepat proses *rendering* yang dilakukan.

Penggunaan spesifikasi komputer yang tinggi pada proses rendering mendorong perlunya penyediaan super computer. Menurut LIPI (Lembaga Ilmu Pengetahuan Indonesia), kendala utama *super computer* adalah pada biaya pengadaan dan operasionalnya. Untuk komputer super Hitachi yang dipakai untuk komputasi fisika energi tinggi di KEK Tsukuba, yang menjadi komputer super tercanggih tahun 1998, diperlukan biaya milyaran yen untuk pembuatan dan instalasinya. Alternatif populer saat ini adalah *cluster computer* (kelompok komputer) atau *parallel computer* (komputer paralel). *Rendering* dengan PC *cluster* lebih cepat dibandingkan dengan *single computer* [1].

Cara paling sederhana untuk mengembangkan prosesor tunggal adalah dengan menambahkan jumlah prosesor yang terhubung ke beberapa modul memory. Keadaan dimana setiap prosesor dapat mengakses modul memory manapun disebut konfigurasi *shared-memory*.



Gambar 1 Komputer konvensional yang memiliki prosesor tunggal dan memory
[sumber: Wilkinson, dkk. 2010]



Gambar 2 Model prosesor shared memory tradisional
[sumber: Wilkinson, dkk. 2010]

Koneksi antara prosesor dan memory dibangun melalui beberapa bentuk jaringan interkoneksi. Sistem multiprosesor *shared memory* menggunakan ruang alamat tunggal, yang berarti setiap lokasi pada keseluruhan memory memiliki alamat yang unik untuk digunakan oleh semua prosesor untuk mengakses lokasi. Meskipun telah nyata ditunjukkan pada model ini, sistem yang nyata memiliki *cache memory* berkecepatan tinggi [2].

Menurut Wilkinson dan Allen [2] selalu saja ada kebutuhan akan kecepatan komputasi yang lebih besar dari sistem komputer saat ini. Beberapa bidang yang membutuhkan komputasi tingkat tinggi adalah simulasi *numeric*, *problem-problem* ilmiah, dan teknik. *Problem-problem* tersebut seringkali membutuhkan kalkulasi *repetitive* pada sejumlah data yang besar guna memperoleh hasil yang valid. Sistem komputasi harus dapat diselesaikan dalam periode waktu yang masuk akal. Pada kenyataannya perusahaan sering menuntut agar waktu perhitungan dapat diselesaikan dalam hitungan menit bahkan detik, Ada beberapa problem yang memiliki tenggat waktu dalam proses komputasinya. Contohnya adalah ramalan cuaca. Jika suatu ramalan cuaca membutuhkan waktu 2 hari, maka informasi tersebut menjadi tidak berguna dihari berikutnya. *Problem* tersebut adalah problem yang tidak dapat diselesaikan dalam waktu yang selayaknya menggunakan komputasi komputer masa kini.

Komputer tradisional memiliki komputer tunggal untuk melaksanakan tugas-tugas dari suatu program. Salah satu cara meningkatkan kecepatan komputer adalah menggunakan beberapa prosesor dalam satu komputer (multiprosesor), maupun beberapa komponen yang mengerjakan satu tugas. Yang menjadi focus utama dalam merencanakan solusi mengunakan multiprosesor adalah perkiraan akan seberapa besar peningkatan kecepatan multiprosesor dalam menyelesaikan suatu problem. Perbandingan tersebut dapat dilakukan menggunakan solusi terbaik yang dapat dilakukan oleh prosesor tunggal, yakni algoritma sekuensial terbaik pada sistem prosesor tunggal dibandingkan dengan algoritma parallel pada multiprosesor [2].

Faktor pemercepat, S(p) adalah ukuran relatif performa yang didefinisikan sebagai berikut [2].:

$$S(p) = t_s/t_p \quad (1)$$

Dimana:

- *S(p)* adalah peningkatan kecepatan jika mengunakan multiprosesor
- t_s adalah waktu proses menggunakan *n* sistem prosesor tunggal (dengan algoritma sekuensial terbaik)
- t_p adalah waktu proses menggunakan multiprosesor dengan *p* prosesor

Algoritma yang digunakan pada komputasi parallel bisa jadi tidak sama dengan algoritma yang digunakan pada prosesor tunggal. Dalam analisis secara teoritis, faktor pemercepat dapat diturunkan berdasarkan banyaknya langkah komputasi [2]..

$$S(p) = \frac{\text{Banyaknya langkah komputasi menggunakan prosesor tunggal}}{\text{Banyaknya langkah komputasi menggunakan } p \text{ prosesor}} \quad (2)$$

$$S(p) \le \frac{t_s}{t_s/p} = p \quad (3)$$

Untuk sekuensial, sudah biasa bila menggunakan kompleksitas waktu untuk membandingkan algoritma yang berbeda. Kompleksitas waktu dapat diperluas kedalam algoritma parallel dan diaplikasikan untuk menghitung factor pemercepat. Namun demikian, tidak ada gunanya jika hanya mempertimbangkan langkah-langkah komputasi, karena pada implementasinya komputasi parallel memerlukan komunikasi mendalam antar bagian. Biasanya, komunikasi tersebut lebih banyak memakan waktu daripada langkah-langkah komputasinya [2].

Percepatan maksimal yang paling dimungkinkan adalah sebesar *p* jika menggunakan prosesor sebanyak *p* (percepatan linier). Percepatan *p* dapat dicapai jika komputasi dapat dibagi kedalam beberapa proses dengan durasi yang sama. Satu proses dikerjakan oleh satu prosesor dan tidak ada overhead lain pada solusi parallel [2].

Percepatan superlinier, dimana *S(p)>p*, mungkin terjadi beberapa kali. Hal tersebut terjadi karena penggunaan algoritma sekuensial yang kurang optimal, *feature* unik dari arsitektur yang mendukung formasi parallel, ataupun sifat dasar algoritma yang kurang menentukan. Umumnya, jika algoritma parallel semata-mata ingin mencapai percepatan lebih dari *p* kali dari algoritma sekuensial, algoritma parallel bias diemulasikan pada prosesor tunggal, masing-masing dikerjakan secara bergantian. Hal tersebut memberi kesan bahwa algoritma sekuensial yang asli tidak optimal.

Kadangkala sangatlah berguna untuk mengetahui jumlah waktu yang diperlukan oleh prosesor untuk melakukan komputasi. Hal itu bias diketahui melalui efisiensi (sistem). Efisiensi E didefinisikan dengan [2]:

$$E = \frac{\text{Waktu eksekusi menggunakan satu prosesor}}{\text{Waktu eksekusi menggunakan multiprosesor} \times \text{jumlah proses}}$$

$$= \frac{t_s}{t_p \times p} \quad (4)$$

$$E = \frac{S(p)}{p} \times 100\% \quad (5)$$

Dimana *E* menghasilkan presentase. Contohnya, jika *E*=50%, prosesor hanya digunakan setengah waktu dari proses rata-rata pada komputasi sebenarnya. Efisiensi 100% terjadi apabila semua prosesor digunakan untuk komputasi sepenuh waktu dengan factor percepatan, *S(p)* sebesar *p* [2].

Sebelumnya Prawira melakukan penelitian tentang analisis performa system cluster pada proses distributed rendering menggunakan open source software. Aplikasi open source software yang digunakan adalah DrQueue. Penelitian ini membahas mengenai strategi infrastruktur dengan memanfaatkan resource komputer dengan spesifikasi yang rendah untuk melakukan rendering yang kompleks, serta penggunaan sistem distributed rendering ini dalam mempercepat proses rendering yang terjadi [3].

Mulyono dan Fatta melakukan penelitian pembuatan game labirin dengan menggunakan Blender 3D. Pada penelitian yang dilakukan menunjukkan bahwa, software berbasis open source bisa menjadi solusi untuk mengurangi biaya produksi [4].

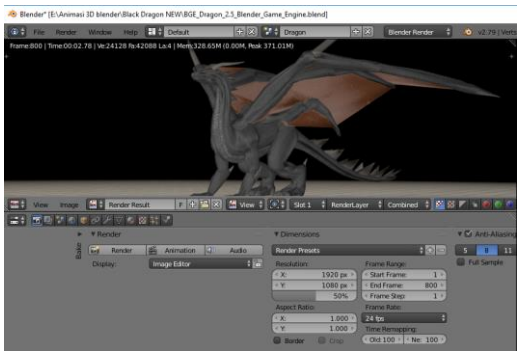
Pada penelitian ini dibuat rancang bangun super komputer untuk rendering video berbasis komputer paralel. Penelitian ini menggunakan aplikasi Blender 2.79. Aplikasi ini mendukung proses rendering dengan melibatkan beberapa komputer slave. Pada rancangan ini akan digunakan 3 buah komputer slave. Pengujian dilakukan terhadap tingkat perubahan waktu eksekusi pada setiap penambahan komputer slave.

2. METODE PENELITIAN

Pada penelitian ini digunakan metode penelitian eksperimen, yaitu dengan memberikan variasi nilai frame untuk mengetahui waktu eksekusi saat dilakukan proses rendering video animasi 3D. Variasi nilai frame tersebut diakukan pada setiap komputer slave, dan selanjutnya membandingkan hasil percepatan waktu eksekusi proses rendering video animasi 3D yang dilakukan. Pada ekperimen ini dilakukan observasi, untuk mengetahui pengaruh perubahan nilai frame terhadap waktu eksekusi proses rendering animasi 3D, pada jumlah komputer *slave* yang berbeda.

Eksperimen dilakukan pada laboratorium dengan menggunakan 5 buah komputer yang dikonfigurasi dalam jaringan *workstation*.

Pengujian komputer paralel yang dibangun dimulai dengan melakukan sinkronisasi waktu dari masing-masing komputer *Master, Slave*, dan *Client* dengan menggunakan *network time protocol*. Seluruh komputer memiliki penunjuk waktu yang sama, hal ini dilakukan agar waktu pengujian untuk seluruh komputer menjadi seragam. Keseragaman waktu diperlukan untuk mengetahui waktu eksekusi yang dibutuhkan saat melakukan proses *rendering*.



Gambar 3 Tampilan File Animasi

Pada proses pengujian digunakan file animasi "BGE Dragon 2.5 Blender Game Engine.blend" yang di-download pada link https://free3d.com/3d-model/black-dragon-rigged-and-game-ready-92023.html.

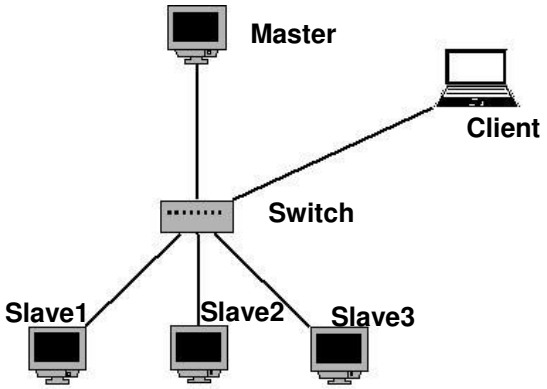
BGE_Dragon_2.5_Blender_Game_Engine.blend Animasi ini menggunakan 800 frame dengan resolusi 1920 x 1080 dan aspect ratio 1000 x 1000, serta frame rate-nya 24 fps.

3. HASIL DAN PEMBAHASAN

3.1 Deskripsi Data

Komputer parallel adalah suatu komputer dengan beberapa prosesor internal maupun beberapa komputer yang saling terhubung membentuk platform komputer tingkat tinggi yang koheren [2].

Implementasi komputer paralel pada penelitian ini dilakukan di Lab Jaringan komputer STMIK STIKOM Indonesia.



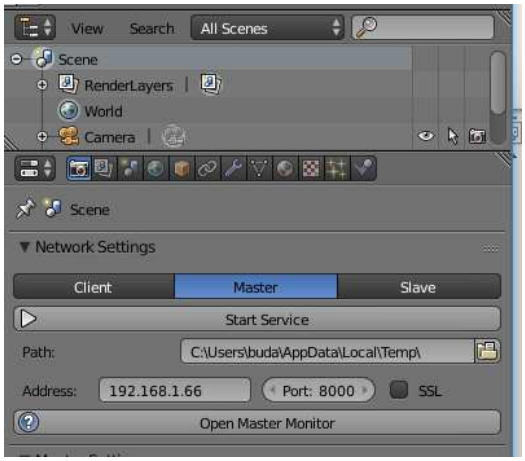
Gambar 4 Topologi Komputer Paralel

Gambar 4 menunjukan topologi komputer paralel yang digunakan dalam melakukan *rendering* video animasi 3D. Maisng-masing komputer pada topologi tersebut dilakukan konfigurasi IP *Address* dalam satu jaringan sehingga dapat saling berkomunikasi. IP Address yang digunakan adalah 192.168.1.0/24

Pada setiap komputer dalam workstation dilakukan konfigurasi aplikasi Blender, sebagai Master, Slave, dan *Client*.

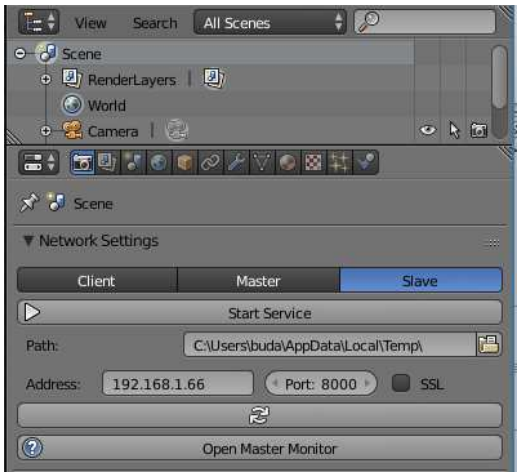
Komputer yang berfungsi sebagai Master dilakukan konfigurasi IP *Address* yang sesuai dengan IP *Address* komputer. Pada penelitian in komputer *Master* menggunakan IP *Address* 192.168.1.66/24. Komputer *Master* berfungsi dalam pendistribusian job ke komputer-komputer *Slave*.

Pada penelitian ini digunakan spesifikasi komputer yang sama antara *Master, Slave*, dan *Client*. Agar komputer *Master* dapat dikenali oleh komputer yang lainnya, maka setelah dilakuka pengaturan IP Address maka pilih *Start Service.*



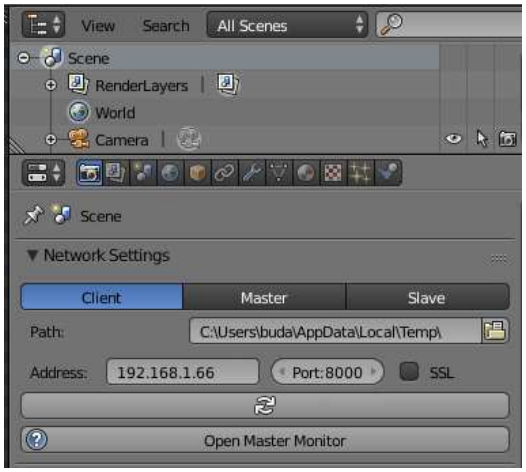
Gambar 5 Konfigurasi Komputer *Master*

Pada penelitian ini digunakan tiga unit komputer yang berfungsi sebagai Slave. Komputer *Slave* merupakan komputer yang akan melakukan proses *rendering*. Semakin banyak komputer *Slave* yang digunakan, maka semakin singkat waktu yang dibutuhkan untuk melakukan proses *rendering*. Gambar 6 menunjukkan konfigurasi dari komputer *Slave* yang digunakan.



Gambar 6 Konfigurasi Komputer *Slave*

Konfigurasi pada komputer *Slave* dilakukan dengan mengatur IP *Address* agar mengarah ke komputer *Master*. Pengaturan dilakukan dengan mengklik tombol *refresh* yang ada dibawah pengaturan IP *Address*. Karena sebelumnya sudah dilakukan konfigurasi pada IP *Address* komputer *Master*, maka IP *Address* akan muncul secara otomatis pada komputer Slave. IP *Address* yang muncul adalah IP Address dari komputer Master.

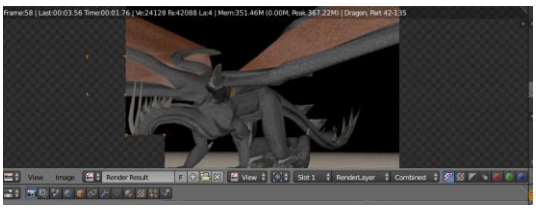


Gambar 7 Konfigurasi Komputer *Client*

3.2 Pembahasan

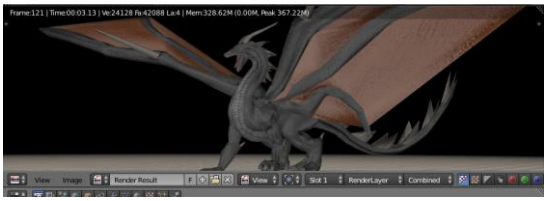
Proses rendering video animasi 3D menunjukkan tahapan proses rendering yang terjadi pada objek file animasi *"BGE Dragon 2.5 Blender Game Engine.blend"*.

Selama proses *rendering* video animasi 3D maka pada aplikasi dapat dilihat pergerakan animasi yang terjadi. Lamanya waktu proses *rendering* video animasi 3D tergantung dari nilai *frame* yang diatur pada setiap animasi.

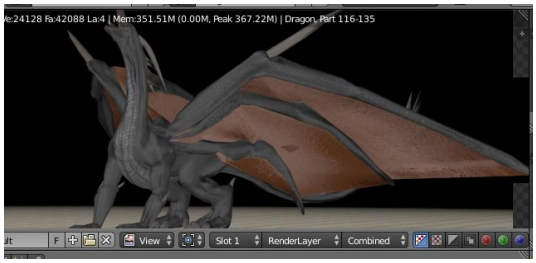


Gambar 8 Proses *Rendering*

Gambar 9 Proses *Rendering* Animasi Kepala Naga Dibawah



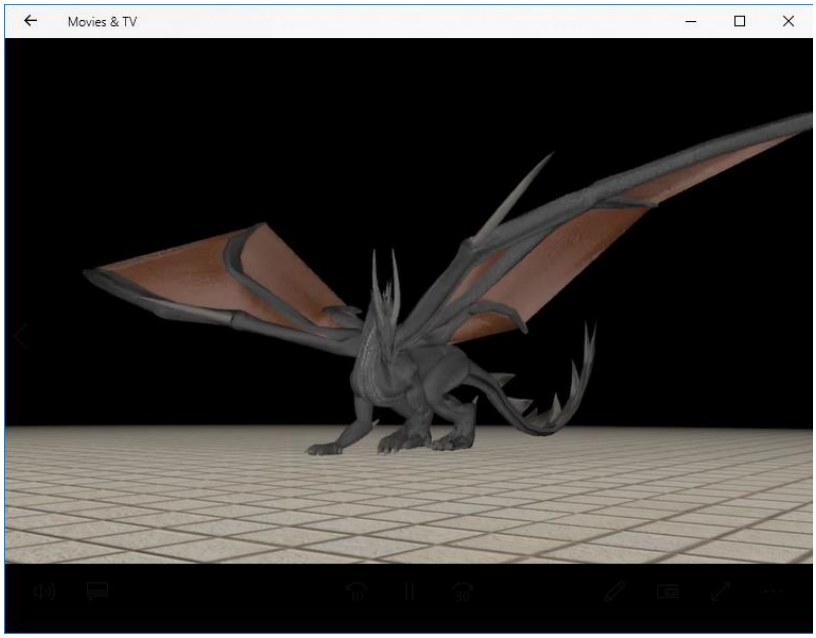
Gambar 10 Proses *Rendering* Animasi Kepala Naga Terangkat



Gambar 11 Proses Rendering Animasi Kepala Naga Mendongak Keatas

Gambar 8, Gambar 9, Gambar 10, dan Gambar 11 menunjukkan cuplikan proses rendering video animasi 3D dengan file animasi *"BGE Dragon 2.5 Blender Game Engine.blend"* sedang berlangsung. Pada gambar dapat dilihat pergerakan kepala dan sayap naga seiring dengan peningkatan waktu proses *rendering*. Ini menunjukkan bahwa rendering animasi 3D yang dilakukan berhasil berjalan sesuai dengan pergerakan animasi yang direncanakan.

Pada Gambar 12 ditunjukkan hasil rendering animasi 3D dengan file animasi *"BGE Dragon 2.5 Blender Game Engine.blend"* yang diputar dengan aplikasi *movie player* bawaan dari Microsoft Windows.



Gambar 12 Hasil Rendering Video Animasi 3D

a. *Rendering* video animasi dengan 1 komputer *slave*.

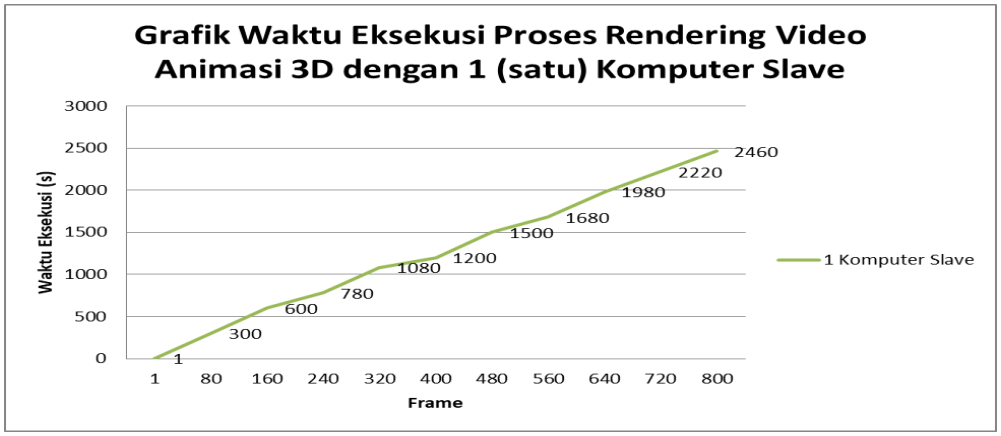
Pada skenario pengujian ini digunakan 3 (tiga) unit komputer yang berfungsi sebagai *Master, Slave*, dan *Client*. Pada proses *rendering* digunakan variasi frame yaitu 1, 80, 160, 240, 320, 400, 480, 560, 740, 800. Besarnya nilai *frame* disesuaikan dengan spesifikasi dari animasi yang digunakan yaitu 800 *frame*.

Implementasi komputer paralel pada penelitian ini dilakukan di Lab Jaringan komputer STMIK STIKOM Indonesia

Hasil dari waktu eksekusi proses *rendering* yang dilakukan disajikan pada tabel 1.

Tabel 1 Waktu Eksekusi dengan 1 komputer *Slave*

Frame	Waktu Eksekusi (s)
1	1
80	300
160	600
240	780
320	1080
400	1200
480	1500
560	1680
640	1980
720	2220
800	2460



Gambar 13 Grafik Waktu Eksekusi Proses *Rendering* Video Animasi 3D dengan 1 (satu) Komputer *Slave*

Pada gambar diatas ditunjukkan bahwa terjadi peningkatan waktu eksekusi seiring dengan peningkatan jumlah *frame*, namun selisih waktu eksekusi antar *frame* yang berbeda mengalami fluktuasi..

b. *Rendering* video animasi dengan 2 komputer slave.

Pada skenario pengujian ini digunakan 4 (empat) unit komputer yang berfungsi sebagai Master, 2 Slave, dan Client. Sama seperti scenario dengan menggunakan 1 unit computer slave, pada proses rendering digunakan variasi frame yaitu 1, 80, 160, 240, 320, 400, 480, 560, 740, 800.

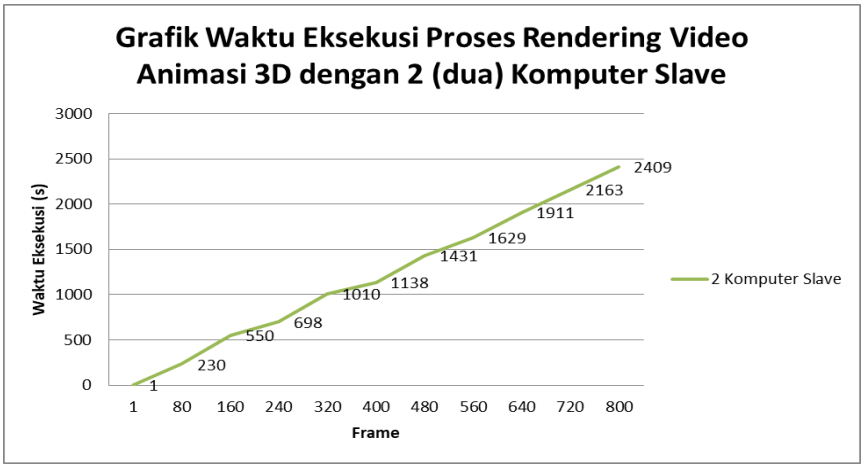
Hasil dari waktu eksekusi proses *rendering* yang dilakukan disajikan pada tabel 2.

Tabel 2 Waktu Eksekusi dengan 2 komputer Slave

Frame	Waktu Eksekusi (s)
1	1
80	230
160	550
240	698
320	1010
400	1138
480	1431
560	1629
640	1911
720	2163
800	2409

Rendering video animasi 3D dengan 2 (dua) komputer *Slave* artinya terdapat 2 (dua) komputer yang bertugas melakukan proses *rendering*. Walaupun terdapat 4 (empat) unit komputer yang digunakan, namun pada skenario pengujian ini hanya terdapat 2 (dua) komputer *Slave* yang digunakan untuk proses rendering video animasi 3D.

Pada Gambar 14 menunjukkan bahwa terdapat peningkatan waktu eksekusi seiring dengan peningkatan jumlah *frame*.



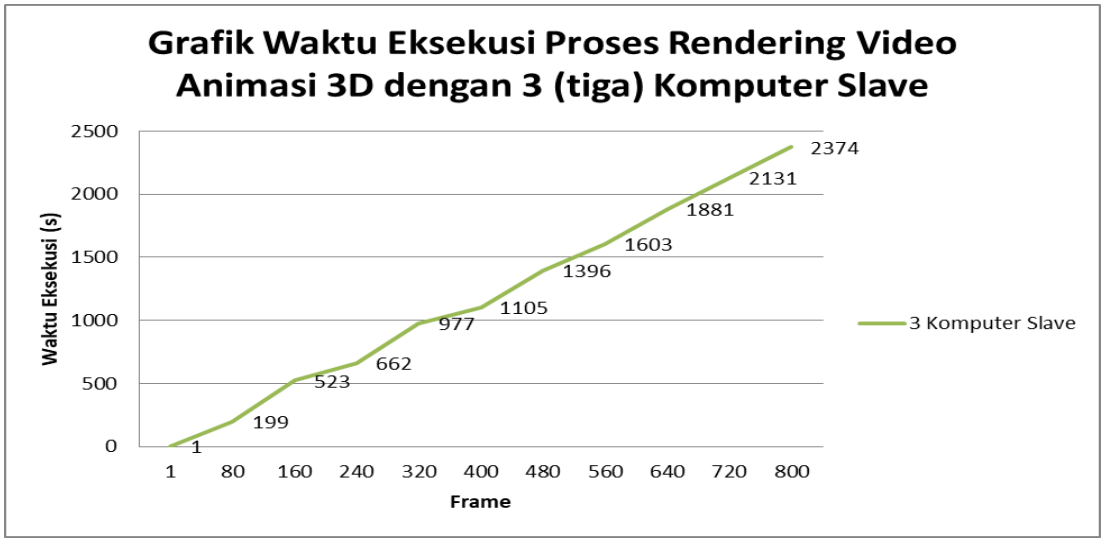
Gambar 14 Grafik Waktu Eksekusi Proses *Rendering* Video Animasi 3D dengan 2 (dua) Komputer *Slave*

c. *Rendering* video animasi dengan 3 komputer *slave*.

Pada skenario pengujian ini digunakan 5 (lima) unit komputer yang berfungsi sebagai *Master*, 3 *Slave*, dan *Client*. Sama seperti skenario sebelumnya, pada skenario ini digunakan 5 unit komputer yang sudah terkoneksi dalam jaringan lokal (*Local Area Network*). Pada proses rendering digunakan variasi frame yaitu 1, 80, 160, 240, 320, 400, 480, 560, 740, 800. Hasil dari waktu eksekusi proses rendering yang dilakukan disajikan pada tabel 3.

Tabel 3 Waktu Eksekusi dengan 3 komputer *Slave*

Frame	Waktu Eksekusi (s)
1	1
80	199
160	523
240	662
320	977
400	1105
480	1396
560	1603
640	1881
720	2131
800	2374



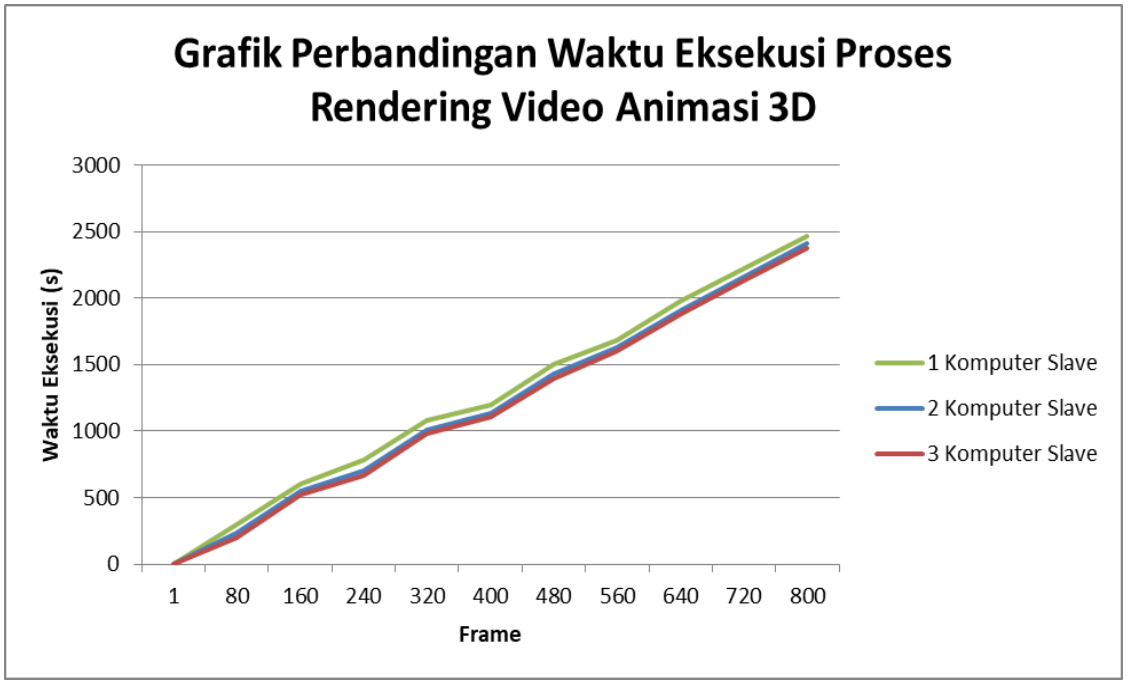
Gambar 15 Grafik Waktu Eksekusi Proses *Rendering* Video Animasi 3D dengan 3 (tiga) Komputer *Slave*

Pada Gambar 15 ditunjukkan grafik waktu eksekusi proses rendering video animasi 3D dengan 3 (tiga) komputer *Slave*. Hasil yang ditunjukkan tidak jauh berbeda dengan skenario pengujian yang sebelumnya, yaitu menggunakan 1 (satu) komputer *Slave* dan 2 (dua) komputer *Slave* terdapat peningkatan waktu eksekusi proses *rendering* video animasi 3D seiring dengan peningkatan jumlah *frame*. Selisih peningkatan waktu proses *rendering* pada setiap pengujian mengalami fluktuasi, hal ini dapat disebabkan oleh berbagai faktor.

Tabel 4 menunjukkan perbandingan waktu proses rendering dengan menggunakan 1 *Master*, 1 *Slave*, dan 1 *Client*; menggunakan 1 *Master*, 2 *Slave*, dan 1 *Client*; dan menggunakan 1 *Master*, 3 *Slave*, dan 1 *Client*.

Tabel 4 Perbandingan Waktu Proses Rendering

Frame	**Waktu Eksekusi (s)**		
	1 *Slave*	**2 *Slave***	**3 *Slave***
1	1	1	1
80	300	230	199
160	600	550	523
240	780	698	662
320	1080	1010	977
400	1200	1138	1105
480	1500	1431	1396
560	1680	1629	1603
640	1980	1911	1881
720	2220	2163	2131
800	2460	2409	2374



Gambar 16 Grafik Perbandingan Waktu Eksekusi Proses *Rendering* Video Animasi 3D

Pada Gambar 16 ditunjukkan grafik perbandingan waktu eksekusi proses *rendering* video animasi 3D. Hasil rendering dengan variasi jumlah komputer *Slave* menunjukkan peningkatan waktu eksekusi proses rendering seiring dengan peningkatan jumlah *frame*. Namun seiring dengan penambahan jumlah komputer *Slave* terdapat penuruan waktu eksekusi proses *rendering*. Hal ini menunjukkan bahwa dengan menggunakan lebih banyak komputer *Slave*, maka akan diperoleh penurunan waktu eksekusi proses *rendering* video animasi 3D.

4. KESIMPULAN

Perancangan dan pembangunan komputer paralel untuk melakukan *rendering* video animasi 3D, dapat dilakukan menggunakan aplikasi *Blender*, Aplikasi *Blender* mendukung dalam proses rendering video animasi 3D melalui jaringan dengan melibatkan beberapa komputer. Dalam menggunakan aplikasi *Blender* untuk melakukan *rendering* video animasi melalui jaringan dibutuhkan minimal 3 (tiga) unit komputer yang berfungsi sebagai *Master, Slave*, dan *Client*.

Proses *rendering* video animasi 3D mengalami peningkatan waktu eksekusi seiring dengan peningkatan jumlah *frame*, namun seiring dengan peningkatan jumlah komputer *Slave* waktu eksekusi proses *rendering* video animasi 3D mengalami penurunan. Semakin banyak jumlah komputer *Slave* yang digunakan maka akan diperoleh percepatan waktu eksekusi proses rendering video animasi 3D yang semakin meningkat.

5. DAFTAR PUSTAKA

[1] B.W.K. Malubaya, A.M. Rumagit, A.S.M. Lumenta, B.A. Sugiarso, "Perancangan PC Cluster Untuk Render Animasi 3D", Jurnal Teknik Elektro dan Komputer, Vol. 3 No 1, 2014.

[2] B. Wilkinson, M. Allen, Parallel Programming Edisi 2: Teknik dan aplikasi menggunakan jaringan workstation dan computer parallel., Yogyakarta: ANDI, 2010.

[3] D. Prawira, "Analisis Performa Sistem Cluster Pada Proses Distributed Rendering Menggunakan Open Source Software", Sekolah Tinggi Manajemen Informatika dan Komputer Pontianak. Program Studi Teknik Informatika, Vol. 2 No 1, 2012.

[4] K.M. Mulyono, H.A. Fatta, "Pembuatan Game Labirin Dengan Menggunakan Blender 3D", Jurnal Dasi, Vol. 13 No. 2, 2012.