Penggunaan Kurva *Lagrange* untuk Memperhalus Permukaan *Mesh* pada Metode *Ray Tracing*

Ferdi Atmaja Wong Susilo[1], Liliana[2], Gregorius Satia Budhi[3]
Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) – 8417658
E-mail: ferdiatmaja@gmail.com[1], lilian@petra.ac.id[2], greg@petra.ac.id[3]

ABSTRAK: Seiring berkembangnya teknologi, hampir semua game dan animasi berpindah dari berbasis 2D menjadi 3D. Ray tracing merupakan metode rendering untuk menghasilkan simulasi tampilan 2D dari objek 3D. Ray tracing memperhitungkan efek pencahayaan dan pencerminan pada setiap face objek mesh. Hal ini menyebabkan pewarnaan face berlangsung lama dan mesh yang digunakan terkadang menghasilkan gradasi warna permukaan yang kasar.
Untuk menyelesaikan permasalahan tersebut, digunakan *lagrange curve* yang memanfaatkan vektor normal dari interpolasi ketiga vektor pembentuk segitiga. Untuk mendapatkan vektor normal yang bervariasi di dalam *face*, digunakan *lagrange curve* dari *face* tetangga di ketiga sisi segitiga. Setelah itu dilakukan proses *rendering*, dimana vektor normal yang dihasilkan *lagrange curve* akan menghasilkan gradasi warna permukaan yang lebih halus.
Pengujian dilakukan dengan membandingkan hasil metode *lagrange curve* dengan metode *subdivision* dan *ray tracing* biasa. Waktu yang dibutuhkan untuk proses *ray tracing* dengan *lagrange curve* meningkat sebesar 23,96% dibandingkan *ray tracing* biasa, namun lebih cepat dari metode *subdivision* (383,71%). Dalam pemakaian *memory*, *lagrange curve* mengalami peningkatan paling besar, yaitu 51,23% dibandingkan *ray tracing* biasa, sedangkan *subdivision* (6.34%). *Lagrange curve* menghasilkan gradasi warna permukaan yang lebih halus dibandingkan dengan *ray tracing* biasa dan metode *subdivision*.

Kata Kunci: *Lagrange Curve, Ray Tracing, Rendering*

***ABSTRACT*:** *As technology develops, most of the games and animations are moving from 2D-based to 3D-based animation. Ray tracing is a rendering method in creating 2D-display simulation extracted from 3D mesh object. When coloring the faces of the mesh object, ray tracing calculates every face's lightings and reflections. This causes the calculation to be slow and the surface color gradation to be rough.*
Lagrange curve utilizing normal vector from interpolation of triangle forming vectors is used to solve this problem. To obtain varied normal vector in a face, lagrange curves from all neighbouring faces are used. The resulting normal vector is used in rendering to produce a smoother color gradation.
Testing is done by comparing the result of lagrange curve method with subdivision and simple ray tracing method. The time needed to process ray tracing using lagrange curve increased by 23.96% compared to simple ray tracing method, but it's faster than subdivision (383.71%). Lagrange curve have the biggest memory usage, 51.23% compared to simple ray tracing, while subdivision scores 6.34% more. Lagrange curve produces a smoother surface color gradation compared to simple ray tracing and subdivison method.

***Keywords*:** *Lagrange Curve, Ray Tracing, Rendering.*

1. LATAR BELAKANG

Seiring dengan berkembangnya kemajuan teknologi, hampir semua *game*, animasi, dan iklan yang dahulunya berbasis 2D, sekarang menggunakan teknologi berbasis 3D. *Rendering* merupakan suatu metode yang dilakukan untuk mensimulasikan tampilan 2D dari suatu objek 3D. Dalam proses *rendering* tersebut, disimulasikan efek pencahayaan, serta efek optik, seperti pencerminan pada permukaan objek 3D berdasarkan jenis dan sifat dari material pembentuk objek 3D. Banyak *software* komersial yang memiliki fitur untuk melakukan proses *rendering* terhadap hasil modelingnya. *Rendering* sendiri telah banyak digunakan untuk proses-proses pembuatan iklan dan video berbasis animasi 3D.
Ray tracing merupakan salah satu metode *rendering* yang banyak digunakan dalam aplikasi komersial untuk menghasilkan simulasi tampilan objek 3D yang menyerupai tampilan aslinya (foto realistik). Contoh aplikasi komersial yang menggunakan metode *ray tracing* dalam proses *rendering* adalah Blender dan 3ds Max. *Ray tracing* memperhitungkan berbagai macam efek pencahayaan dalam objek, serta dapat menjelaskan karakteristik iluminasi global pencahayaan langsung, bayangan, dan refleksi *specular*. Berbagai proses perhitungan detail pada setiap objek dibutuhkan untuk menghasilkan simulasi permukaan objek yang demikian, sehingga menyebabkan proses yang membutuhkan waktu yang relatif lama.
Proses *ray tracing* memperhitungkan seluruh objek, semakin banyak objek, proses menjadi lebih lama. *Mesh* merupakan bentuk objek kompleks yang sering digunakan dalam proses *ray tracing*. *Mesh* tersusun dari banyak segitiga yang menutupi seluruh permukaan. Untuk menghasilkan permukaan yang halus dan tidak terkesan kaku, dibutuhkan ukuran segitiga yang kecil dalam jumlah yang banyak pada satu *mesh*. Teknik *subdivision* digunakan untuk membagi segitiga menjadi beberapa bagian yang lebih kecil, sehingga permukaan menjadi lebih halus dan tidak terkesan kaku [2]. Namun dengan jumlah objek yang meningkat, waktu untuk melakukan proses *rendering* pada objek *mesh* meningkat juga.

Selain dalam jumlah objek, gradasi warna juga mempengaruhi halus atau tidaknya tampilan pemukaan objek. Sebuah objek segitiga diwarnai dengan sebuah warna, berdasarkan normal bidangnya sebagai penentu utama, yang dihasilkan dari perhitungan ketiga titik pembentuk segitiga. Pada objek primitif bola, setiap titik pada permukaan bola mempunyai normal tersendiri, yang menyebabkan adanya gradasi warna yang halus pada permukaan bola. Dari hal tersebut, dapat diambil hipotesa sebagai berikut, sebuah bidang yang memiliki normal di setiap titik pembentuk bidang, muncul banyak warna pada bidang tersebut, yang diharapkan dapat menghasilkan gradasi warna yang halus.

Dari permukaan bola yang merupakan lengkungan, jika dilakukan pembentukan permukaan dengan menggunakan kurva *lagrange*, akan mendapatkan normal bidang yang berbeda pada setiap titik pembentuk permukaan segitiga. Dengan adanya *lagrange surface* yang terbentuk dari kurva-kurva, maka permukaan yang datar akan tampil seolah-olah melengkung. Kurva *lagrange* dipilih karena metode ini dapat menghasilkan permukaan yang lebih halus [1].

2. *RAY TRACING*

Ray tracing adalah sebuah metode untuk menggambar objek 3D yang hasilnya realistik, sama seperti foto. *Ray tracing* akan menentukan permukaan yang tampak dan akan memberikan warna pada setiap *pixel*. Proses yang dilakukan dalam metode ini adalah dengan menelusuri sinar dari arah mata atau sumber cahaya. Oleh karena itu, *ray tracing* dikenal dengan metode *rendering* yang membutuhkan waktu lama dengan proses perhitungan yang banyak [6]. Jika sinar tersebut mengenai objek, maka akan dilakukan perhitungan intensitas dari objek tersebut. Hasil dari perhitungan intensitas dari objek inilah yang terlihat oleh mata.

Metode *backward ray tracing* adalah metode yang melakukan penelusuran sinar dari mata [5]. Sinar dipancarkan dari mata ke arah setiap *pixel* pada layar gambar, kemudian sinar tersebut diteruskan ke objek-objek yang akan digambar.

Ketika sinar yang melalui suatu *pixel* menabrak suatu objek, akan dilakukan perhitungan intensitas pada titik tabrak dari objek tersebut. Hasil perhitungan dari intensitas tersebut akan digunakan untuk memberi warna pada *pixel* tersebut.

Perhitungan intensitas dilakukan dengan memperhitungkan efek pencahayaan dan efek optik. Ketika sinar yang dipancarkan tersebut tidak mengenai objek apapun, *pixel* tersebut akan diberi warna sama dengan warna latar belakangnya (*background*). Efek pencahayaan yang dihitung ,yaitu *ambient, diffuse,* dan *specular*.

Ambient adalah estimasi pencahayaan untuk penerangan global dalam suatu lingkungan sehingga arah cahaya tidak dapat diketahui, seakan-akan cahaya datang dari segala arah, seperti sinar matahari saat langit tampak berawan. Efek dari *ambient* akan mempengaruhi terang atau tidaknya suatu lingkungan yang terlihat oleh mata. Intensitas *ambient* pada suatu objek dapat dicari dengan persamaan 1 [4]:

$$I = I_a * K_a \quad (1)$$

dimana I_a merupakan intensitas *ambient* dan K_a merupakan koefisien *ambient*.

Diffuse adalah pencahayaan yang tergantung dari besarnya sudut yang dibentuk antara sinar dari lampu ke titik tabrak pada objek dengan normal objek. Posisi lampu sangat mempengaruhi efek *diffuse* ini. Intensitas *diffuse* dapat dicari dengan persamaan 2 menggunakan hukum Lambertian sebagai berikut:

$$I = I_d * K_d (L \bullet N) / d \quad (2)$$

dimana I_d merupakan intensitas *diffuse*, K_d merupakan koefisien *diffuse*, *L* merupakan vektor dari titik tabrak ke sumber cahaya, *N* merupakan vektor normal objek, dan d merupakan jarak antara sumber cahaya dengan titik tabrak.

Specular merupakan efek pencahayaan dimana bayangan sumber cahaya terlihat pada permukaan objek yang mengkilap. Semakin mengkilap permukaan suatu objek maka makin jelas bayangan sumber cahaya yang terlihat. Untuk mencari intensitas *specular* maka dapat digunakan persamaan 3 sebagai berikut:

$$I = I_p * K_s (R \bullet V)^n / d \quad (3)$$

dimana

$$R = -S + 2* (S \bullet N) * N \quad (4)$$

I_p merupakan intensitas *specular*, K_s merupakan koefisien *specular*, *R* merupakan vektor arah pantulan, *V* merupakan vektor negasi arah sinar, d merupakan jarak antara sumber cahaya dengan titik tabrak, *S* merupakan vektor dari titik tabrak ke sumber cahaya, dan *N* merupakan vektor normal dari objek.

3. INTERPOLASI *LAGRANGE*

3.1. Kurva *Lagrange*

Kurva *lagrange* adalah bentuk interpolasi *polynomial* dengan minimal titik berjumlah 3. Bentuk umum interpolasi *polynomial lagrange* berderajat *n* adalah:

$$f_n(x) = \sum_{i=0}^{n} L_i(x) f(x_i) \quad (5)$$

dimana

$$L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j} \quad (6)$$

Dari persamaan 5 dan persamaan 6, dapat dihitung interpolasi lagrange berderajat 2, yaitu dengan persamaan 7 berikut ini:

$$f_2(x) = L_0(x)\, f(x_0) + L_1(x)\, f(x_1) + L_2(x)\, f(x_2) \quad (7)$$

3.2. Permukaan *Lagrange*

Permukaan *lagrange* adalah bentuk interpolasi *polynomial F(x, y)* dengan sejumlah titik data, minimal ber-ordo tiga. Permukaan *lagrange* dihitung dengan persamaan 8 [3]:

$$f(x,y) = \sum_{i=1}^{n} L_i(x) * \sum_{k=1}^{m} L_k(y) * f(x_i, y_k) \quad (8)$$

dimana nilai $L_i(x)$ dan $L_k(y)$ dapat dicari dengan persamaan 9 dan 10 berikut:

$$L_i(x) = \sum_{\substack{i=1 \\ i \neq j}}^{n} \frac{x - x_j}{x_i - x_j} \quad (9)$$

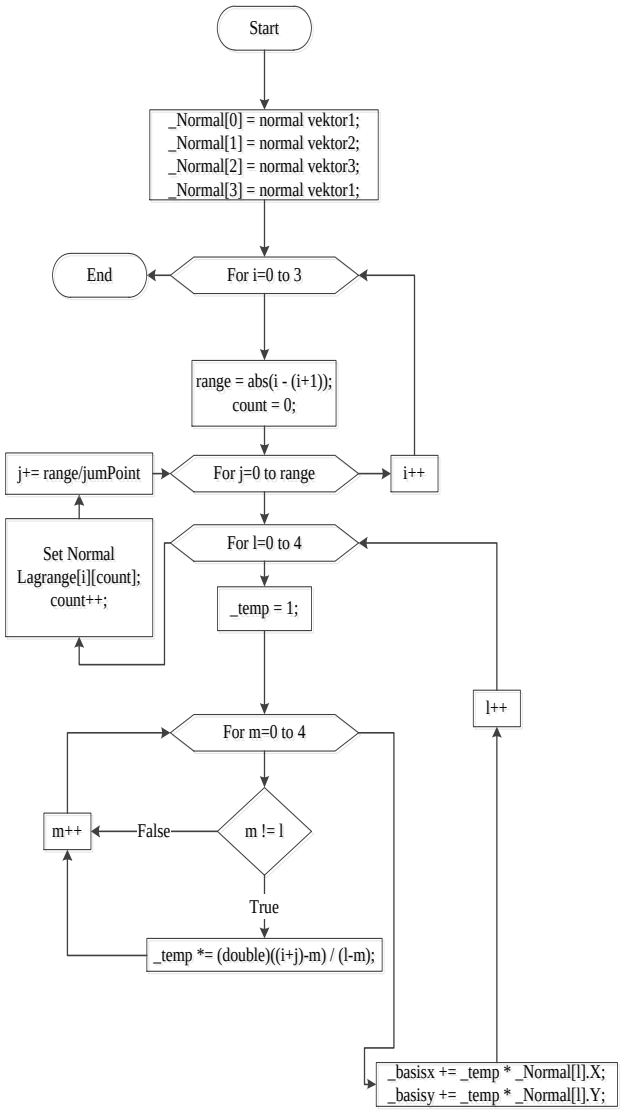
$$L_k(x) = \sum_{\substack{k=1 \\ k \neq l}}^{m} \frac{y - y_l}{y_k - y_l} \quad (10)$$

4. DESAIN SISTEM

Pada pembuatan aplikasi ini terdapat beberapa tahapan proses yang akan dilakukan. Proses pertama adalah proses insialisasi *ray tracing*, dilakukan *input* objek mesh beserta konfigurasi pada kamera, sumber cahaya, dan perhitungan awal vektor ke *pixel* pertama di ujung kiri atas layar.

Proses kedua merupakan proses perhitungan nilai setiap *pixel* dengan memperhitungkan setiap efek pencahayaan, yaitu *ambient, diffuse,* dan *specular*, beserta efek pemantulan dan bayangan.

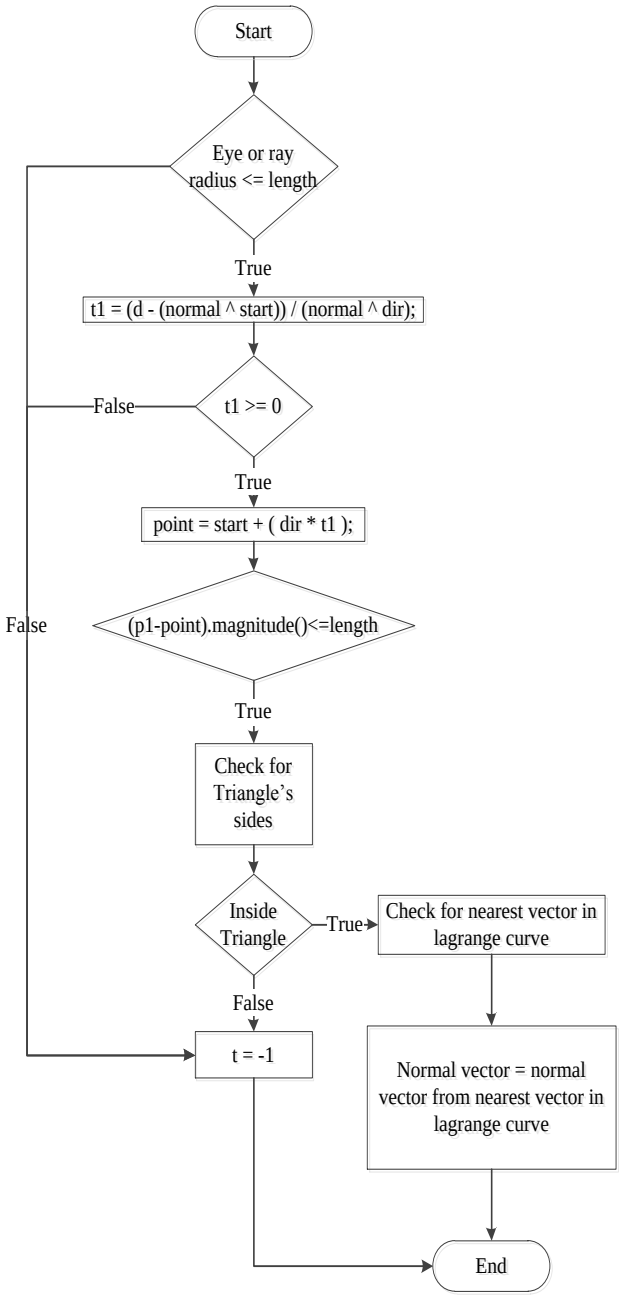
Proses dimulai dengan melakukan *rendering* pada *pixel* ujung kiri atas layar hingga *pixel* ujung kanan bawah layar.
Proses terakhir adalah menampilkan *output* program berupa tampilan 2D dari hasil *rendering*.
Dalam proses inisialisasi pada objek *mesh*, dilakukan perhitungan normal bidang dari tiga buah vektor pembentuk *face*, selain itu dilakukan juga pembentukan kurva *lagrange* beserta normal vektornya pada setiap titik pembentuk kurva.
Untuk mendapatkan interpolasi normal vektor menggunakan *lagrange*, digunakan proses seperti pada Gambar 1 berikut ini:



Gambar 1. Diagram alir pembentukan *lagrange normal*

Interpolasi vektor normal pada *lagrange curve* dibentuk dari 3 vektor pembentuk *face*, dimana nilai setiap vektornya didapatkan dari rata-rata normal *face* yang menggunakan vektor tersebut. Proses untuk mendapatkan vektor normal dari kurva *lagrange* dilakukan dengan cara mencari vektor terdekat dari setiap kurva lagrange yang dimiliki tetangga pada *face*, dengan titik tabrak sinar terhadap bidang *face*. Nilai dari interpolasi vektor normal inilah yang akan digunakan pada proses perhitungan warna.

Pada proses *rendering* setiap *pixel*, dilakukan proses *intersection* terhadap setiap objek, objek yang terkena cahaya akan diambil warnanya. Dalam proses *intersection* terdapat tahap proses pencarian vektor terdekat pada *lagrange curve* dari seluruh tetangga *face* tersebut dengan titik jatuh sinar dan mengambil vektor normalnya yang akan dilakukan seperti diagram alir pada Gambar 2.



Gambar 2. Diagram alir proses pencarian vektor terdekat pada *lagrange curve* dari seluruh *face* tetangga

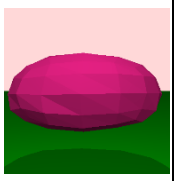
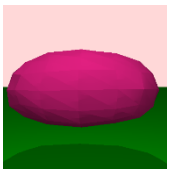
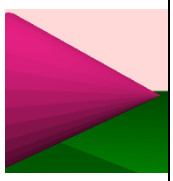
Setelah mendapatkan vektor normal dari *lagrange*, dilakukan perhitungan warna dengan efek *ambient*, *diffuse*, dan *specular*.

5. PENGUJIAN

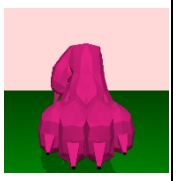
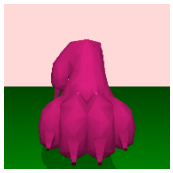
Dalam melakukan pengujian perangkat lunak akan dilakukan perbandingan hasil *ray tracing* menggunakan *lagrange curve* yang dibuat pada skripsi ini dengan *subdivision.*

Pengujian pertama dilakukan untuk membandingkan hasil *rendering* dari metode-metode tersebut dalam hal gradasi warna permukaan yang lebih halus, penggunaan waktu dalam satu kali buah proses *rendering*, serta penggunaan *memory* dalam satu kai proses *rendering*. Pengujian dilakukan dengan menggunakan 8 buah objek *mesh*, dan hasil pengujian dapat dilihat pada Tabel 1 berikut ini:

Tabel 1. Pengujian terhadap gradasi warna permukaan

No	Objek Asli	*Subdivision*	*Lagrange Curve*
1			
2			
3			
4			
5			

Tabel 2. Pengujian terhadap gradasi warna permukaan (lanjutan)

No	Objek Asli	*Subdivision*	*Lagrange Curve*
6			
7			
8			

Dari hasil pengujian 8 buah objek *mesh*, didapatkan bahwa objek *mesh* dengan menggunakan *lagrange curve* memiliki gradasi warna permukaan yang lebih halus dibandingkan dengan objek aslinya maupun objek yang telah di proses dengan metode *subdivision.*

Berdasarkan urutan objek yang digunakan pada Tabel 1, berikut adalah perbandingan jumlah *face* yang digunakan untuk pengujian:

Tabel 2. Jumlah *face* yang digunakan pada pengujian

No	Objek Asli	*Subdivision*	*Lagrange Curve*
1	192	768	192
2	62	372	62
3	1152	4608	1152
4	80	480	80
5	1692	10134	1692
6	968	3936	968
7	830	3079	830
8	1572	6244	1572

Dari Tabel 2, diketahui bahwa dengan menggunakan *lagrange curve*, tidak akan menambah jumlah *face* dan menghasilkan gradasi warna permukaan yang lebih halus dari objek aslinya. Dengan menggunakan metode *subdivision*, jumlah *face* bertambah, namun gradasi warna permukaan yang dihasilkan tidak nampak jelas.

Dari hasil pengujian-pengujian pada Tabel 1, akan dibandingkan peningkatan waktu untuk satu kali proses *ray tracing* antara *mesh* tanpa metode *smoothing*, dengan *mesh* yang telah menggunakan metode *lagrange curve* dan *subdivision*, baik *simple subdivision*, maupun *catmull-clark subdivision.*
Berikut adalah hasil peningkatan waktu untuk satu kali proses *ray tracing* dari *simple ray tracing* dengan metode pengujian:

Tabel 3. Peningkatan waktu pada proses *rendering*

No	*Simple Ray Tracing*	*Subdivision*	*Lagrange Curve*
1	0%	530%	105,65%
2	0%	500 %	25%
3	0%	283,62%	8,62%
4	0%	150,00%	36,36%
5	0%	501,74%	1,74%
6	0%	376,99%	9,73%
7	0%	420,25%	2,83%
8	0%	307,06%	1,76%
Rata-Rata	0%	383,71%	23,96%

Dari Tabel 3 di atas, diperoleh bahwa penggunaan metode *subdivision* dan *lagrange curve* akan meningkatkan pemakaian waktu untuk satu kali proses *ray tracing*. Peningkatan waktu tersebut terjadi karena adanya proses tambahan yang terjadi pada proses *intersection* yang berlangsung semakin lama seiring sebakin bertambahnya jumlah *face* pada suatu *mesh*.
Berdasarkan pengujian ketiga metode tersebut, metode *lagrange curve* memiliki peningkatan waktu yang terendah dan membuat metode *lagrange curve* unggul dalam hal waktu proses *rendering*.
Berikut adalah hasil peningkatan *memory* yang digunakan untuk satu kali proses *ray tracing* dari *simple ray tracing* dengan metode pengujian:

Tabel 4. Peningkatan penggunaan *memory* pada proses *rendering*

No	*Simple Ray Tracing*	*Subdivision*	*Lagrange Curve*
1	0%	2,57%	7,28%

Tabel 4. Peningkatan penggunaan *memory* pada proses *rendering* (lanjutan)

No	*Simple Ray Tracing*	*Subdivision*	*Lagrange Curve*
2	0%	2,30%	3,93%
3	0%	7,99%	72,84%
4	0%	0,97%	4,21%
5	0%	16,51%	106,03%
6	0%	6,39%	61,98%
7	0%	5,64%	58,83%
8	0%	8,36%	94,74%
Rata-Rata	0%	6,34%	51,23%

Dari Tabel 4 di atas, diperoleh bahwa penggunaan metode *subdivision* dan *lagrange curve* akan meningkatkan pemakaian *memory* untuk satu kali proses *ray tracing*.

Berdasarkan pengujian dengan metode tersebut, metode *lagrange curve* memiliki peningkatan waktu yang tertinggi dan membuat metode *lagrange curve* tidak unggul dalam hal pemakaian *memory*.

6. KESIMPULAN

Berdasarkan hasil pengujian yang dilakukan didapatkan beberapa kesimpulan, yaitu:

- Gradasi warna permukaan yang dihasilkan dengan menggunakan *lagrange curve* lebih *smooth* dan terlihat tidak terkotak-kotak dibandingkan dengan objek aslinya maupun dengan metode *subdivision*.
- Waktu yang dibutuhkan untuk sebuah proses *ray tracing* menggunakan *lagrange curve* mengalami peningkatan rata-rata sebesar 23,96%. Sedangkan proses dengan menggunakan metode *subdivision* mengalami peningkatan waktu rata-rata sebesar 383,71%. Hal ini membuktikan bahwa metode *lagrange curve* menghasilkan proses yang relatif cepat dibandingkan dengan proses *ray tracing* tanpa metode *smoothing* dan lebih unggul dalam penggunaan waktu proses dibandingkan metode *subdivision.*
- *Memory* yang dibutuhkan untuk sebuah proses *ray tracing* menggunakan *subdivision* mengalami peningkatan rata-rata sebesar 6,34%. Sedangkan proses dengan menggunakan metode *lagrange curve* mengalami peningkatan waktu rata-rata sebesar 51,23%. Peningkatan *memory* yang tinggi ini disebabkan oleh pembentukan *lagrange curve* pada setiap *face*, yang membuat pemakaian *memory* yang besar pada *pre-processing.* Hal ini membuktikan bahwa metode *lagrange curve* kurang unggul dalam penggunaan *memory* dibandingkan metode *subdivision.*

7. REFERENSI

[1] Alani, T.F., Bedan, A.S. 2008. *3D Surface Generation Algorithm Using Lagrange Basis Functions in CAD/CAM Application*.

[2] Benthin, C., Boulos, S., Lacewell, D., Wald, I. 2007. *Packet-based Ray Tracing of Catmull Clark Subdivision Surfaces*.

[3] Bozorgmanesh, A.R., Otadi, M., Kordi, A.A.S., Zabihi, F., Ahmadi, M.B. 2009. *Lagrange Two-Dimensional Interpolation Method for Modeling Nanoparticle Formation during RESS Process*. IJIM.

[4] Liliana. 2006. *Memperhalus Permukaan Obyek Mesh yang Direnderr Dengan Menggunakan Ray Tracing*. Prosiding Seminar Nasional Transformasi Teknologi Untuk Peningkatan Kualitas Hidup Manusia Universitas Teknologi Yogyakarta.

[5] Shirley, P., Marschner, S. 2010. *Fundamentals of Computer Graphics 3rd Edition*. Taylor and Francis Group, LLC.

[6] Wald, I., Slusallek, P., Benthin, C., Wagner, M. 2001. *Interactive Rendering with Coherent Ray T racing*. Computer Graphics Group: Saarland University.