

Implementasi Sistem Partikel Menggunakan Metoda *Smoothed Particle Hydrodynamics (SPH)* untuk Simulasi Aliran Lava

Khairul Hamdi, Emil Mauludi Husni dan Tunggal Mardiono
Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung
Jl. Ganesha No.10, Bandung 40132
E-mail : khairul.hamdi@gmail.com, ehusni@lisk.ee.itb.ac.id

Abstrak

Simulasi aliran lava merupakan fenomena yang rumit dan kompleks karena banyaknya parameter yang terlibat. Metoda yang digunakan untuk mensimulasikan aliran lava adalah pendekatan Lagrangian dengan menganggap lava sebagai fluida yang disusun oleh sistem partikel. Metoda Smoothed Particle Hydrodynamics (SPH) diimplementasikan untuk interaksi antar partikel yang menyusun komponen fluida. Parameter fisis yang dimiliki oleh fluida lava berbasis partikel adalah kerapatan, viskositas, tekanan, tegangan permukaan, suhu dan gaya eksternal. Penggunaan metoda SPH memungkinkan sistem partikel dapat berinteraksi dengan obyek lain seperti terrain. Penentuan partikel tetangga terdekat menggunakan metoda staggered grid, fungsi smoothing kernel yang digunakan adalah kernel Spiky dan Poly6 yang tingkat kestabilan dan akurasi yang tinggi. Deteksi tumbukan antara partikel dan terrain menggunakan metoda Sphere Plane Sweep Test dan rendering karpet digunakan metoda quadtree. Aliran lava yang dihasilkan cukup realistis dan dapat mensimulasikan beberapa parameter fisis.

Kata kunci: SPH, smoothing kernel, staggered grid, terrain

1 Pendahuluan

Salah satu obyek kajian yang cukup menarik adalah pemodelan fenomena gunung api. Indonesia yang terletak dikawasan cincin api (*fred ring*) yang ditandai dengan tingginya aktifitas vulkanis dan terdapat banyak gunung berapi, merupakan tantangan tersendiri bagi peneliti untuk mempelajari berbagai seluk beluk tentang gunung berapi dan membuat berbagai pemodelan gunung berapi sebagai sarana belajar bagi masyarakat dan pihak-pihak yang berkepentingan lainnya. Salah satu pemodelan yang penting adalah pemodelan aliran lava.

Aliran lava dianggap sebagai fluida yang dibentuk oleh sistem partikel. Berbagai kelakuan yang terjadi pada fluida adalah hasil dari interaksi antara partikel yang ada dalam sistem. Interaksi antar partikel dimodelkan menggunakan metoda *Smoothed Particle Hydrodynamics (SPH)*. Metoda SPH merupakan metoda yang lazim digunakan dalam *Computational Fluid Dynamics (CFD)* berbasis *gridless* menggunakan pendekatan Lagrangian. Metoda SPH mempunyai kelebihan sebagai berikut :

1. cocok digunakan untuk memodelkan fluida dengan deformasi yang tinggi dan perpindahan antar muka;
2. mempunyai tingkat akurasi, adaptibilitas dan stabilitas yang tinggi;
3. aplikasi yang sangat luas dari pemodelan mikroskopis (komponen atom) sampai skala makroskopis (pemodelan gugus galaksi);
4. aplikasi SPH telah dapat mencakup berbagai aspek fisis fluida seperti viskositas, gaya eksternal, gaya internal, kerapatan, transfer panas dan lain sebagainya.

Aliran lava merupakan peristiwa yang kompleks yang dengan berbagai peristiwa fisika dan kimia yang berlangsung di dalamnya.

Tujuan dari penelitian ini adalah sebagai berikut :

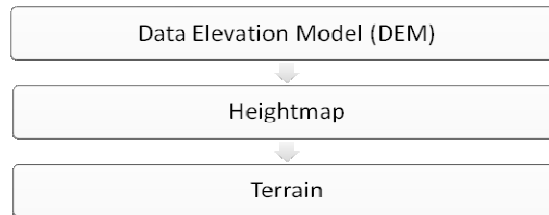
1. simulasi aliran lava yang mempunyai parameter fisis kerapatan, tekanan, viskositas, suhu dan gaya gravitasi;
2. simulasi aliran lava yang dapat berinteraksi dengan obyek lain;
3. visualisasi obyek dan aliran lava yang realistis dan mendekati fenomena sebenarnya.

Penelitian sebelumnya dilakukan oleh [Cani,1999] menggunakan metoda SPH untuk memodelkan aliran lava dengan menghitung densitas, viskositas dan transfer panas. Pengembangan selanjutnya menambahkan parameter lain seperti ukuran partikel, *smoothing length* dan *rendering karpet (carpet rendering)* yang melingkupi bagian atas fluida untuk visualisasi yang lebih realistis.

2 Terrain dan Sistem Partikel

Terrain umumnya diimplementasikan untuk obyek-obyek yang statis. Salah satu implementasi *terrain* yang banyak digunakan dalam pemodelan lanskap. Penggunaan *terrain* sebagai model lansekap ini banyak

didapati dalam game *motor rally* dan *real-time strategy*. Pembuatan model *terrain* dapat dilakukan menggunakan data *dummies* maupun data *heightmap* yang diolah dari peta digital.



Gambar 1 Pemrosesan data *Heightmap*.

Data *heightmap* yang mengandung informasi ketinggian dijadikan sebagai data masukan untuk men-*generate* *terrain*. *Terrain* digunakan sebagai model gunung api, model gunung api yang digunakan merupakan model yang sebenarnya karena data masukan *heightmap* diperoleh dari peta digital hasil pemotretan satelit. Tahapan dalam pemrosesan data *heightmap* adalah sebagai berikut :

1. data DEM diperoleh dari citra satelit, setiap titik dari peta DEM mengandung informasi dan letak dan ketinggian;
2. data DEM diubah menjadi *heightmap* menggunakan *software* 3DEM, keluaran yang diperoleh adalah gambar *heightmap* dengan format *grayscale*, warna putih menunjukkan ketinggian maksimum, warna hitam untuk area yang paling rendah;
3. mengatur kedalaman warna *heightmap* menggunakan *image editor*, dengan aplikasi ini kedalaman warna adalah 8 bit, sehingga mempunyai jangkauan 0 – 255, artinya *terrain* mempunyai 256 skala ketinggian.

Data *heightmap* dapat berupa file RAW atau BMP, *image grayscale* ini selanjutnya dijadikan data masukan untuk men-*generate* *terrain*. Untuk men-*generate* *terrain* digunakan *terrain engine*.

Partikel adalah entitas yang tidak mempunyai parameter ruang, merupakan obyek yang mempunyai parameter posisi, kecepatan dan gaya interaksi antar sesamanya serta dapat dipengaruhi oleh gaya luar. Sistem partikel adalah kumpulan massa titik pada ruang tiga dimensi yang dihubungkan dengan gaya tertentu dan dipengaruhi oleh gaya eksternal. Gaya eksternal yang mempengaruhi sistem partikel diantaranya adalah gaya gravitasi dan adanya gesekan yang terjadi antar partikel.

3 Aspek Fisis Aliran Lava

Aliran lava sebagai fenomena alam merupakan sistem yang rumit, kompleks dan melibatkan banyak parameter fisika dan kimia. Untuk mensimulasikan aliran lava, berbagai parameter yang ada disederhanakan dan parameter yang kontribusinya sedikit dapat diabaikan. Parameter fisis yang dimiliki oleh lava adalah sebagai berikut.

1. Nilai viskositas tergantung komposisi kimia material lava yang dikeluarkan dari dapur magma. Untuk kandungan kimia yang sama, nilai viskositas akan berkurang secara eksponensial saat temperatur menurun.
2. Kerapatan yang dimiliki oleh lava dipengaruhi oleh material yang ada di dalamnya. Aliran lava mengandung lelehan silikat dan aluminium yang mencair akibat suhu yang tinggi. Material ini membuat kerapatan lava menjadi tinggi. Nilai kerapatan massa lava basaltik adalah 2500 kg.m⁻³.
3. Suhu lava yang paling tinggi saat lava baru dikeluarkan dari kawah. Suhu lava dalam keadaan ini berkisar antara 1200°-1400°C [ESDM,--]. Suhu yang tinggi ini berkurang secara eksponensial yang dipengaruhi oleh lingkungan sekitar. Energi termal lava diserap oleh lingkungan sekitar seperti tanah pegunungan dan udara. Laju pendinginan lava sebanding gradien suhu antara lava tersebut dengan lingkungan sekitar.
4. Warna lava tergantung suhu yang dimilikinya. Menurut hukum Pergeseran Wien (Wien's displacement law) bila suhu berkurang maka kurva radiasi benda hitam akan berpindah ke arah intensitas yang lebih rendah dan panjang gelombang yang lebih panjang [Wikipedia,2008] sesuai dengan persamaan (1) :

$$T\lambda_{\max} = 2.898 * 10^6 \text{ nmK} \quad (1)$$

Suhu 3000 K, benda hitam memancarkan radiasi yang mempunyai intensitas yang tinggi dengan panjang gelombang 700 nm yang menghasilkan warna merah. Untuk lava, suhu awal 1200°C akan membuat lava berpijar dan berwarna orange kemerahan, disuhu rendah warna lava semakin meredup menjadi merah kehitaman.

4 Desain dan Implementasi Aliran Lava

Desain dan implementasi aliran lava menggunakan metoda SPH dilakukan dengan membuat model gunung api menggunakan terrain, partikel bergerak sepanjang terrain. Partikel mengalami *detection collision* dengan terrain dan dipantulkan dengan faktor pemantulan 0,25. Interaksi antar partikel menggunakan metoda SPH.

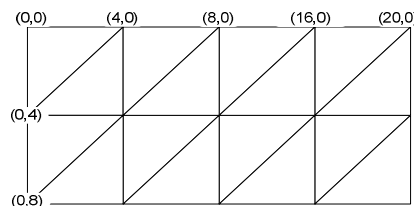
4.1 Generating Terrain

Metoda yang digunakan untuk *generating terrain* adalah memetakan ketinggian vertek *terrain* sesuai dengan skala yang terdapat dalam titik piksel dalam *heightmap*. Pengolahan bitmap *grayscale* menjadi *terrain*, data bitmap *grayscale* terlebih dahulu dikonversi menjadi *mesh* poligonal. Metoda ini, citra *heightmap* dianggap sebagai kisi yang mempunyai nilai-nilai ketinggian. Kisi-kisi ini akan diubah secara lansung menjadi rangkaian vertek-vertek yang tersusun. Nilai koordinat x dan y setiap piksel akan diubah menjadi posisi vertek. Nilai warna dari setiap lokasi piksel akan diubah menjadi posisi z dalam setiap vertek. Ketika dilakukan pembacaan nilai *grayscale* dalam jangkauan 0 sampai 255, kemudian dilakukan proyeksi ketinggian, dengan titik tertinggi adalah 255 dan titik terendah adalah 0, sehingga *heightmap* yang terbentuk mempunyai 256 skala.

Empat buah piksel persegi yang mempunyai dimensi panjang 2 piksel dan lebar 2 piksel, akan membentuk 2 buah segitiga yang terdiri dari empat vertek. Penyimpanan vertek sebagai sebuah daftar yang memuat informasi posisi dalam koordinat x, y dan z. Data segitiga yang disimpan mempunyai 3 indeks dalam daftar vertek, yang menggambarkan vertek-vertek yang digunakan untuk setiap segitiga. Sehingga terdapat dua jenis data yaitu *vertek* dan indeks *vertek*. Untuk penentuan posisi vertek, juga dihitung permukaan normal untuk setiap segitiga yang merupakan hasil perkalian silang antara 2 vektor yang telah ternormalisasi.

4.2 Penentuan Posisi Partikel dalam Terrain

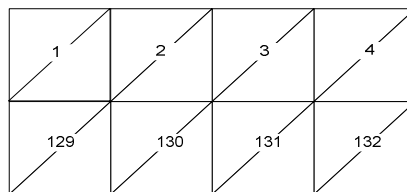
Ukuran *heightmap* yang digunakan adalah panjang 128 piksel dan lebar 128 piksel, ukuran ini digunakan agar program dapat dijalankan lebih cepat karena *heightmap* yang luas membutuhkan komputasi yang lebih besar. Setiap titik piksel membentuk satu buah vertek pada *terrain*. Tiga buah vertek yang berdekatan akan membentuk sebuah segitiga, sehingga empat vertek yang berdekatan membentuk dua segitiga seperti dalam Gambar 2.



Gambar 2 Penentuan *distance* perpiksel pada setiap vertek.

Triangle[0] mempunyai titik sudut yang bersesuaian dengan vertek[0], vertek[1] dan vertek[128]. Triangle[1] bersesuaian dengan vertek[1], vertek[128] dan vertek[129]. Jumlah segitiga yang terbentuk adalah 32258 yang mempunyai indeks Triangle[0] sampai Triangle[32257].

Dua buah segitita yang sisi miringnya berhimpit membentuk sebuah kotak. Jumlah kotak dalam satu baris atau satu kolom adalah 127 buah. Masing-masing kotak mempunyai indeks dari *box*[1] sampai *box*[16129].



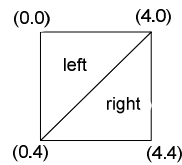
Gambar 3 *Indexing box* untuk penentuan area tumbukan.

Sebuah partikel yang mempunyai kedudukan (x, y, z) akan mengalami tumbukan dengan *terrain* bila posisi z sama dengan tinggi sumbu z dalam *terrain* sesuai dengan proyeksi ketinggian *heightmap* terhadap *terrain*. Untuk penentuan posisi tumbukan dalam koordinat (x,z), ditentukan dahulu nilai x_{local} dan z_{local} dengan persamaan (2):

$$\begin{aligned}
 x_{local} &= \text{ceil}\left(\frac{x}{4}\right) \\
 z_{local} &= \text{ceil}\left(\frac{z}{4}\right)
 \end{aligned}
 \tag{2}$$

Faktor pembagian 4 digunakan karena nilai *distance perpiksel* mempunyai nilai 4. Nilai *distance perpiksel* menyatakan banyaknya grid yang terdapat disatu piksel. Setelah nilai x_{local} dan z_{local} didapatkan, ditentukan dalam indeks *box* berapa partikel tersebut berada menggunakan persamaan (3) :

$$indexbox = (x_{local} - 1) + ((z_{local} - 1) * (width - 1)) + (z_{local} - 1) \tag{3}$$



Gambar 4 Segitiga atas dan segitiga bawah pada sebuah *box* lokal.

Setelah posisi partikel ditentukan, selanjutnya penentuan posisi partikel apakah berada di segitiga kiri atau segitiga kanan, terlebih dahulu ditentukan nilai x_{temp} dan z_{temp} menggunakan persamaan (4) :

$$\begin{aligned} x_{temp} &= x - (jarakperpiksel * (x_{local} - 1)) \\ z_{temp} &= z - (jarakperpiksel * (z_{local} - 1)) \end{aligned} \tag{4}$$

Segitiga atas dan segitiga bawah dibatasi oleh garis diagonal yang didefinisikan dengan persamaan (5) :

$$z = mx + c \tag{5}$$

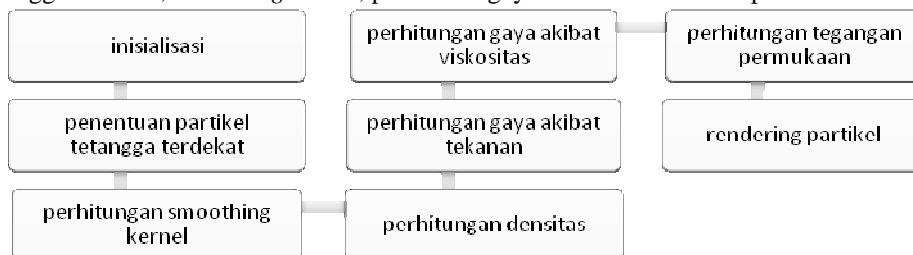
Jika

$x_{temp} > z_{temp}$ maka posisi partikel berada di segitiga kiri, atau

$x_{temp} < z_{temp}$ maka posisi partikel berada di segitiga kanan

4.3 Simulasi Aliran Lava Menggunakan SPH

Metoda *Smoothed Particle Hydrodynamics* digunakan untuk pemodelan interaksi antar partikel. Model SPH yang digunakan adalah model yang diajukan oleh [Muller,2005]. Pemodelan SPH yang diajukan meliputi penentuan tetangga terdekat, *smoothing kernel*, penentuan gaya akibat tekanan dan penentuan viskositas.



Gambar 5 Desain sistem *Smoothed Particle Hydrodynamics*.

Dalam Gambar 5, untuk menentukan nilai *Smoothing kernel* terlebih dahulu ditentukan tetangga terdekat dan fungsi *smoothing kernel* berlaku untuk partikel tetangga terdekat tersebut. Setelah fungsi *Smoothing kernel* ditentukan, langkah selanjutnya adalah perhitungan nilai kerapatan, viskositas, tekanan dan tegangan permukaan.

4.4 Penentuan Tetangga Terdekat

Struktur data yang digunakan untuk pemodelan lava adalah metoda *staggered grids* yang dikembangkan oleh [Kipfer,2006]. Struktur data ini dikembangkan untuk sistem partikel dengan kerapatan partikel yang rendah. Struktur data *staggered grids* menggunakan memori yang lebih efisien dan komputasi yang tidak berubah saat aplikasi dijalankan.

Daftar linear digunakan untuk menyimpan partikel. *Grid* virtual digunakan untuk memetakan partikel. Indeks *bin* (penyimpanan) (i_x, i_y, i_z) dikombinasikan untuk membuat *identifier* 64 bit untuk setiap *bin id* = $|0| i_z | i_y | i_x |$, sorting partikel didasarkan daftar *identifier* ini. Untuk menentukan partikel tetangga terdekat, terlebih dahulu dilakukan pengecekan terhadap yang terdapat di sumbu y, misalkan radius *kernel* yang digunakan adalah r dan titik koordinat partikel referensi yang akan ditentukan tetangganya adalah $i(x,y,z)$ kemudian dilakukan *sorting* partikel yang mempunyai kedudukan $(x,y,z-r)$ sampai $(x,y,z+r)$. Setelah ditentukan penyortiran partikel dalam area tersebut, selanjutnya dilakukan penyortiran partikel dalam area $(x,y-r,z)$ sampai $(x,y+r,z)$. Langkah terakhir adalah menyortiran partikel dalam area $(x-r,y,z)$ sampai $(x+r,y,z)$. Hasil akhir yang didapatkan adalah daftar tetangga partikel $i(x,y,z)$ dengan radius sisi r. Akhirnya diketahui posisi partikel mana saja yang termasuk tetangga terdekat dari partikel referensi $i(x,y,z)$. Ketiga langkah ini dilakukan untuk setiap

partikel setiap kali step simulasi. Penggunaan pendekatan staggered grid ini, terdapat tiga daftar line yang sederhana. *Pseudocode* untuk pencarian tetangga terdekat adalah sebagai berikut.

```

untuk setiap partikel {
    tentukan partikel referensi
    untuk setiap partikel referensi {
        tentukan partikel pada arah z
        simpan koordinat posisi z
        tentukan partikel pada arah y
        simpan koordinat posisi y
        tentukan partikel pada arah x
        simpan koordinat posisi x
    }
}

```

Dalam penentuan tetangga terdekat ini setiap partikel dicek partikel tetangganya sehingga membutuhkan komputasi yang besar.

4.5 Smoothing Kernel

Kestabilan, akurasi dan kecepatan SPH yang digunakan sangat tergantung *smoothing kernel* yang digunakan. *Kernel* yang digunakan dalam pemodelan lava ini adalah kernel yang didesain oleh [Muller,2005]. Kelebihan yang dimiliki oleh kernel ini adalah r yang mempunyai nilai kuadrat yang dapat dievaluasi tanpa menghitung akar saat menghitung jarak. Jika kernel ini digunakan untuk menghitung gaya tekanan, partikel akan menggumpal dengan tekanan yang tinggi. Jarak yang sangat dekat antar partikel membuat gaya tolak-menolak menjadi hilang karena gradien kernel menuju nol di titik pusat. Untuk mengatasi masalah ini digunakan *spiky kernel* yang diperkenalkan oleh Desrun [Muller,2005]. Kernel Spiky akan meng-*generate* gaya tolak antar partikel, sehingga partikel tidak menggumpal di sebuah *cluster* dengan tekanan yang tinggi.

Viskositas merupakan fenomena yang disebabkan oleh gesekan yang mengurangi energi kinetik fluida dan mengubahnya menjadi panas. Untuk menghitung gaya viskositas, digunakan kernel jenis ketiga yang diperkenalkan oleh [Muller,2005].

4.6 Gaya Akibat Tekanan dan Viskositas

Fluida mempunyai parameter tekanan dan viskositas. Tekanan yang dialami oleh sebuah partikel terjadi akibat akumulasi tekanan partikel tetangga. Besarnya pengaruh tekanan partikel tetangga tergantung kepada jarak partikel tersebut terhadap partikel referensi. Partikel dengan jarak terdekat mempunyai pengaruh paling signifikan sedangkan partikel yang tidak masuk dalam daftar partikel tetangga terdekat tidak mempunyai pengaruh sama sekali [Sariel,2008]. Gaya yang dialami sebuah partikel akibat tekanan dinyatakan dalam persamaan (6).

$$f_i^{\text{tekanan}} = -\sum_j m_j \frac{p_i + p_j}{2\rho_j} \nabla W(r_i - r_j, h) \quad (6)$$

Gaya viskositas tergantung perbedaan kecepatan dan tidak tergantung kecepatan absolut. Gaya akibat viskositas dirumuskan oleh [Muller,2005] sebagai berikut.

$$f_i^{\text{viskositas}} = \mu \sum_j m_j \frac{v_i + v_j}{\rho_j} \nabla^2 W(r_i - r_j, h) \quad (7)$$

4.7 Rendering Karpet

Konstruksi karpet dapat digunakan sebagai *mesh* teratur yang melingkupi *terrain*. Pendekatan yang diimplementasikan dalam konstruksi karpet ini adalah metoda karpet virtual. Karpet dikonstruksi dan dirender hanya saat partikel ada di suatu *terrain*. Struktur data *quadtree* digunakan untuk mengidentifikasi posisi partikel secara efisien. Metoda yang digunakan untuk konstruksi karpet ini adalah sebagai berikut [Kipfer,2006]:

1. partikel disimpan di daun / anak *quadtree*;
2. untuk setiap simpul, nilai ketinggian maksimum ditarik *tree*, nilai ini mendefinisikan permukaan karpet virtual;
3. bagian yang terlihat dalam karpet mengalami *rendering* melewati *tree* tergantung ketinggian fluida yang berada di atas *terrain*.

Karpet terdiri dari $n \times n$ persegi virtual, yang merupakan resolusi karpet selama proses *rendering*. Simpul *tree* menyimpan tinggi *terrain* minimum absolut daerah ini, yaitu tinggi karpet dan nilai kecepatan. Kemudian dilakukan penyimpanan anak *tree* dengan menyimpan tinggi absolutnya yang berkorespondensi ruang menggunakan operasi max. Kemudian karpet didorong ke arah ketinggian tepat diatas partikel. Kecepatan karpet ini diset sama dengan nol. Selanjutnya simpul *inner quadtree* di *update* dengan menjalarkan nilai ketinggian dari atas kebawah untuk setiap langkah.

Rendering dilakukan secara efisien menggunakan metoda *quadtree*, jika nilai ketinggian partikel ditemukan berada dibawah nilai ketinggian *terrain*, maka rekursi dibatalkan dan tidak terjadi *rendering* karpet. Bila sampai simpul anak, dapat dipastikan bagian tersebut bagian tampak yang akan mengalami *rendering*. Karpet diakselerasi oleh percepatan gravitasi dan posisi karpet dilengkungkan ke bawah. Jika partikel berpindah posisi, maka karpet akan turun sampai menuju permukaan *terrain*. Tahap ini tidak mengalami *rendering* karena posisi partikel sudah tidak ada lagi disana. Jika partikel masih ada disana, maka algoritma *quadtree* tahap kedua akan memulihkan kembali nilai ketinggian yang betul. Karpet dapat *diupdate* secara *increment*, struktur data spasial yang digunakan dalam model SPH mempunyai informasi yang dapat digunakan untuk *update* karpet ini.

Permukaan yang *smooth* dapat dibuat dengan meminimalisasi lengkungan lokal sehingga karpet yang dirender akan kelihatan bagian yang terpisah atau terisolasi. Bagian permukaan spasial tidak harus terkoneksi karena ada partikel yang terpisah dari kumpulan partikel lain. *Pseudocode* untuk *rendering* karpet menggunakan metoda *quadtree* dengan *pseudocode* sebagai berikut.

```

class Quadtree {
    Rectangle bounds;
    int num;
    Point point;
    Quadtree sub[]; }
void query(vector set, Rectangle range) {
    if (pada perpotongan batas {
        if (num == 1) {
            if (pada jangkauan ditemukan partikel {
                tambahkan elemen
            }
        } else if (num > 1) {
            for (int i=0; i<4; i++) {
                generate struktur anak (daun)
            }
        }
    }
}
masukan titik
if (num == 0) {
    point = p;
}
else if (num == 1) {
    split ();
    masukan titik
    point = null;
    masukan titik pada subkuadran
}
else {
    masukan titik
}
num++;
}

```

Dalam pengembangan *terrain* menggunakan metode *quadtree*, setiap simpul mempunyai 4 anak, simpul yang tidak mempunyai anak dinamakan dengan daun (*leaf*). Bagian *terrain* yang datar, yang tidak mempunyai detail yang banyak, maka bagian tersebut simpul *quadtree* tidak mempunyai anak. Sedangkan bagian *terrain* yang mempunyai banyak detail, bagian tersebut mempunyai simpul dalam struktur data *quadtree* mempunyai anak yang kedalamannya sesuai dengan detail *terrain* dalam area tersebut.

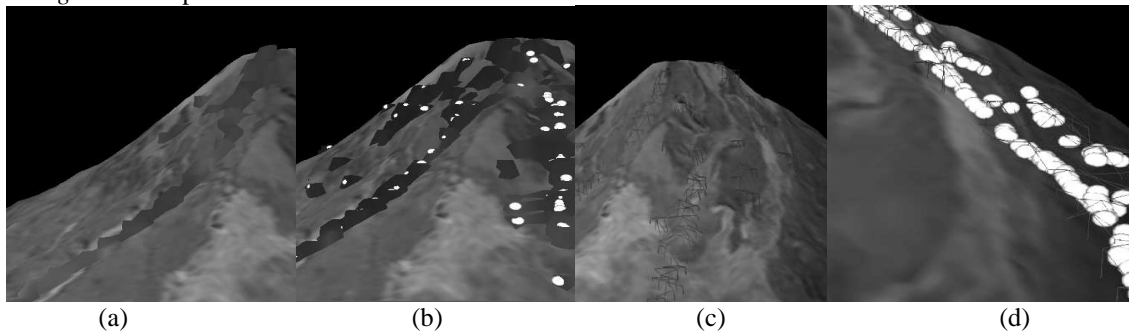
5 Hasil dan Pembahasan

Spesifikasi perangkat lunak adalah kompiler Microsoft .NET, *graphic utility* openGL sebagai *tool* aplikasi grafik. Inisialisasi komputasi SPH dengan menetapkan nilai-nilai variabel seperti yang tercantum dalam Tabel 1.

Tabel I Inisialisasi nilai-nilai variabel aplikasi SPH.

VARIABEL	NILAI
Jumlah partikel	1.000
Percepatan gravitasi	-9.8
<i>Sampling distance</i>	0.5
<i>Smoothing length</i>	2.5
Kerapatan awal	2500
Radius partikel	0.5
Volume partikel	5

Simulasi dijalankan menggunakan prosesor Intel Core 2 Duo 2.0 Gh dan *graphic card* NVidia GeForce 8600. Nilai *frame persecond* (fps) untuk jumlah partikel 100 adalah 39 fps. Ketika jumlah partikel yang dirender 500, nilai fps yang dihasilkan 8 fps dan turun menjadi 3 fps saat partikel yang dirender 1000 partikel. Hasil *rendering* lava terdapat dalam Gambar 6.



Gambar 6 Simulasi aliran lava menggunakan SPH, (a) *Rendering* lava menggunakan karpet, (b) *Rendering* karpet dan partikel, (c) *Rendering* wireframe, (d) *Rendering* partikel.

Gambar 6 mevisualisasikan aliran lava dengan jumlah partikel 1000. Nilai fps menurun tajam saat jumlah partikel meningkat karena komputasi yang tinggi saat penentuan partikel tetangga. Simulasi aliran lava menghasilkan visualisasi yang cukup realistis untuk menggambarkan aliran lava pada lereng gunung.

Daftar Pustaka

- B. C. Vemuri, Y. Cao, Chen, L. (1998) *Fast collision detection algorithms with applications to particle flow*. *Computer Graphics Forum*, hal. 121–134.
- Brid,R., Muller-Fischer, M. (2007), *Fluid Simulation*, SIGGRAPH, Course Notes, Zürich.
- Crawford, Chris (2003), *Chris Crawford on Game Design*, New Riders, Indianapolis.
- Gomez , Miguel (1999), *Simple Intersection Test for Games*, Gamasutra, http://www.gamasutra.com/features/19991018/Gomez_1.htm, 26 Mei 2008, 9.35 WIB
- Har-Peled, Sarel (2008), *Quadrees – Hierarchical Grids*, Creative Commons, San Francisco, California.
- Kipfer, P., Westermann, R. (2006), *Realistic and Interactive Simulation of Rivers*, Graphics Interface, Quebec.
- Liu, G.R., Liu, M.B. (2003), *Smoothed Particle Hydrodynamics – a meshfree particle method*, World Scientific, Singapore.
- Muller, Matthias, et.al. (2003), *Particle-Based Fluid for Interactive Applications*. Dept. Comp. Science, Federal Institute of Tech. Zurich (ETHZ), Zürich.
- Muller , M., Solenthaler, B., Keiser, R., Gross, M. (2005), *Particle-Based Fluid-Fluid Interaction*, Eurographics/ACM SIGGRAPH Symposium on Computer Animation, Zürich.
- Reiterer, Florian (2006), *Particle Based Real Time Fluid*, Institute of Computer Graphic and Algorithms, TU Vienna, Vienna.
- Roy, Trina, M. (1995), *Physically Based Fluid Modelling using Smoothed Particle Hydrodynamics*. Master Thesis, University Illinois at Chicago, Chicago.
- Samet,H. (1989), *Spatial Data Structures: Quadrees, Octrees, and other Hierarchical Methods*, Addison-Wesley, Massachusetts.
- Sempe, L. (2002), *Terrain Rendering using Heightmaps*, <http://www.spheregames.com>, 20 Mei 2008, 09.30 WIB.
- Snook, Greg (1996), *Real-Time 3D Terrain Engines Using C++ and DirectX 9*, Charles River Media, Hingham. Massachusetts, 119-152.
- Stahler ,Wendy (2004), *Beginning Math and Physics for Game Programmers*. New Riders, Indianapolis.
- Stora, Dan et.al., *Animating lava flows*. In Graphics Interface, pages 203–210, 1999, Paris.
- _____, *Black Body Radiation*, WIKIPEDIA, http://en.wikipedia.org/wiki/Talk:Black_body, 12 juli 2008, 10.10 WIB.
- _____, *Differential Equations, Linear Equation Application*. Lecturer Notes. <http://www.math.stcc.edu/DiffEq>, 2 Juli 2008, 15.35 WIB
- _____, Pusat Vulkanologi dan Mitigasi Bencana Geologi, Pengenalan Gunung Api, Departemen ESDM RI, <http://merapi.vsi.esdm.go.id/vsi/>, 6 Januari 2008, 8.10 WIB.