Implementasi Aljabar Geometri dalam Tiling 2D Terrain dengan Cellular Automata dan Vektor

Davin Prasetya, 13514003[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13514003@itb.ac.id

***Abstraksi*—Medan / Terrain merupakan salah satu penentu vital bagi game, hampir semua game atau aplikasi menggunakan medan tertentu dalam fungsi nya, meski beberapa tidak memfokuskan fungsi game/aplikasi ke medan tersebut sehingga dapat dibuat seadanya saja seperti tembok yang statik. Tetapi untuk pembuatan game atau aplikasi yang berfokus pada medan yang dinamis, pembuatan struktur medan, variasi, dan hukum hukum fisika di dalam medan tersebut merupakan kunci vital bagi kesuksesan game atau aplikasi tersebut. Dalam makalah ini akan dibahas tentang pembuatan medan bersistemkan vektor dengan menggunakan cellular automata. Tujuan penggunaan sistema ini adalah pembuatan medan yang responsif terhadap perubahan medannya, terstruktur rapi tanpa ada petak yang terbang dan cepat untuk diload. Tujuan ini dibuat dengan pertimbangan suatu medan yang mendekati medan yang natural.**

***Keywords*—Struktur Pemetaan Terrain, 2D Tiling, Cellular Automata, Aplikasi Vektor**

I. PENDAHULUAN

Dalam pembuatan game bermodel pixelated 2D game, sering dibutuhkan suatu terrain/medan yaitu suatu tembok, pijakan atau benda lain yang menyatakan sesuatu berupa alam seperti air,daun dan lain-lain. Dalam pembuatan hal ini, seringkali beberapa masalah timbul karena medan yang dibuat kurang baik, kurang variatif, kurang natural, kurang fleksibel atau tidak peka terhadap perubahan medan. Di makalah ini, akan dibahas teknik pembuatan medan 2 dimensi yang cukup fleksibel untuk diubah ubah sesuai keinginan, ditambahkan jenis petaknya, tersusun rapi ,variatif, artinya berubah berubah setiap pembuatan medannya dan peka terhadap perubahan.

Teknik yang dipakai disini adalah gabungan dari teknik matriks geometri bernama celular automata dan aplikasi vektor. Teknik celular automata digunakan untuk merangkai medan dari kosong dengan memasukkan aturan aturan yang dapat kita masukan dengan cukup mudah dan hasilnya dapat rapi tersusun sesuai dengan yang kita inginkan. Vektor disini dipakai sebagai pembuat medan lebih dinamis karena apabila yang digunakan berupa matriks, sistem kurang fleksibel untuk menyesuaikan dengan perubahan nilai sebagai contoh penghilangan petak yang menyebabkan area dari medan terselebung tersebut berubah. Selain itu, aplikasi vektor juga dapat mempermudah objek objek petak yang dinamis seperti petak berjenis air.

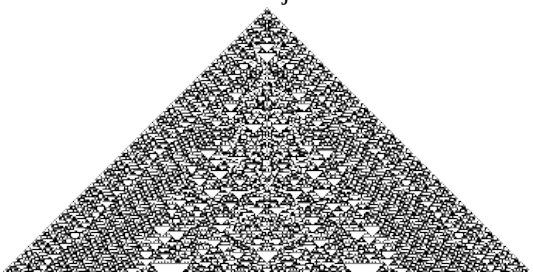
II. CELLULAR AUTOMATA

A. Penjelasan Singkat Cellular Automata

Cellular Automata adalah kumpulan dari suatu petak di dalam grid / kisi-kisi dengan bentuk tertentu, yang umumnya merupakan suatu persegi. Bentuk ini didapat melalui suatu proses secara diskrit dengan membandingkan satu petak dengan petak sekelilingnya. Cellular Automata ditemukan pertama oleh Stanislaw Ulam dan John von Neumann pada tahun 1940. Teknik ini kemudian terus dikembangkan hingga pada tahun 2002 teknik ini sudah berhasil untuk diaplikasikan ke dalam bidang komputer grafik dan juga kriptografi.

B. Algoritma Cellular Automata

Algoritma utama dari cellular automata adalah membandingkan state petak yang satu dengan petak yang lain/neighbor. Agar proses perbandingan itu menghasilkan hasil yang diinginkan, dibuatlah rule sebagai penentu terhadap hasil. Pemilihan tipe tetangga ada 2 jenis yang terkenal, Von Neumann neighborhood dan Moore neighborhood. Sebagai contoh yang paling mendasar adalah *elementary cellular automata,* yaitu cellular automata yang berupa piramid segitiga dan terdiri dari hanya 2 state, yaitu state 0 dan state 1. Setiap petak bergantung pada nilai state petak diatasnya, dan 2 petak di sebelahnya/neighbor. Sebagai contoh adalah rule 30 class 3 cellular automata yang terkenal sebagai penggambaran dari kekacauan dan keacakan sejarah.



source : http://mathworld.wolfram.com/CellularAutomaton.html

Fig 1.Rule 30 Cellular Automata

Pattern	111	110	101	100	011	010	001	000
State	0	0	0	1	1	1	1	0

Table I. Set Rule 30

C. Neighborhood

Seperti yang dijelaskan sebelumnya, ada 2 jenis pemilihan tetangga/neighbor yaitu Von Neumann neighborhood dan Moore neighborhood. Von Neumann neighborhood menggunakan algoritma penentu petak dengan tetangga 4 petak yang mengelilinginya. Sedangkan Moore neighborhood menggunakan algoritma penentu petak dengan tetangga 8 petak/semua petak yang mengelilinginya. Kedua cara pengambilan tetangga ini bergantung pada program yang kita inginkan, Moore neighborhood digunakan di kebanyakan program pengolah gambar / image editing seperti Adobe Photoshop.

III. VEKTOR

A. Penjelasan Singkat Vektor

Vektor adalah bentuk yang merepresentasikan sesuatu yang mempunyai besar dan arah. Sejarah penemuan vektor sampai sekarang masih abu abu. Tidak jelas penemu awalnya, sehingga vektor diduga sebagai hasil Aristoteles (384-322 B.C) yang hilang. Perkembangan vektor secara sistematik baru kemudian dikembangan di abad 19 dan 20. Beberapa pengembang awal yang diketahui adalah Caspar Wessel (1745-1818) dan Jean Robert Argand (1768-1855).

B. Representasi Vektor dengan Dua Titik

Vektor pada dasarnya merupakan sebuah garis dengan atribut tambahan yaitu arah. Syarat cukup bagi terbentuk nya sebuah garis adalah 2 titik, demikian juga dengan vektor.

Misalkan ada titik P dengan koordinat (Xp,Yp) dan Q dengan koordinat (Xq,Yq), maka dapat dibuat sebuah vektor a dengan nilai

$$a = \begin{bmatrix} Xq - Xp \\ Yq - Yq \end{bmatrix} = \begin{bmatrix} Xa \\ Ya \end{bmatrix}$$

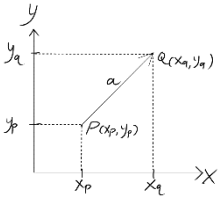


Fig 2.Vektor 2 Titik

Dengan besar vektor a adalah

$$\|a\| = \sqrt{{x_a}^2 + {y_a}^2}$$

Dan arah vektor a adalah

$$\cos \alpha = \frac{x_a}{\|a\|}$$

C. Hukum-hukum Dasar Vektor

Seperti aljabar biasa, vektor juga mempunyai hukum hukum dasar yang berlaku dalam vektor, seperti

1. Hukum identitas,
$$v + 0 = v + 0 = v$$
$$1 * v = v$$
2. Hukum inverse,
$$v + (-v) = (-v) + v = 0$$
3. Hukum asosiasi,
$$(v_1 + v_2) + v_3 = v_1 + (v_2 + v_3)$$
$$\lambda(\varepsilon v) = (\lambda\varepsilon)v$$
4. Hukum komutasi, dan
$$v_1 + v_2 = v_2 + v_1$$
5. Hukum distribusi.
$$\lambda(v_1 + v_2) = \lambda v_1 + \lambda v_2$$
$$(\lambda + \varepsilon)v = \lambda v + \varepsilon v$$

IV. TILING 2D TERRAIN

A. Penjelasan Singkat Tentang Tiling 2D Terrain

Tiling 2D Terrain adalah suatu proses untuk menghasilkan medan/terrain bermodelkan 2 dimensi secara rapi. Tujuan yang diinginkan adalah suatu medan dengan wilayah terbuka diatas permukaan tanah dan suatu area dibawah tanah yang mempunyai jalan, bisa diakses ,tidak mempunyai 1 blok yang berdiri sendiri (tidak punya neighbor), fleksibel menerima perubahan, dan mampu menerima konsep fisika dari fluida yakni mengalir dari tempat tinggi ke rendah tanpa merubah volume yang ia miliki.

Konsep yang digunakan disini adalah metode cellular automata dan konsep vektor. Ada beberapa proses yang harus dilalui program untuk dapat mencapai hasil yang diinginkan, yaitu

1. Tahap pengacakan dan pembuatan petak
2. Tahap penyusunan petak
3. Tahap implementasi jenis petak
4. Tahap perubahan petak menjadi vektor
5. Tahap visualisasi

B. Tahap Pengacakan dan Pembuatan Petak

Tahap pengacakan dan pembuatan petak mempunyai tujuan akhir yaitu terbentuknya sebuah susunan nilai petak sesuai atribut diinginkan yang secara acak terbentuk. Tahap pembuatan ini dilakukan secara iterasi setiap petak dan memasukkan atribut petak kedalamnya.

Dalam pembuatan petak awal ini, tetap perlu dimasukkan beberapa aturan pengacakan. Seperti jumlah persentasi petak yang diinginkan, dengan menggunakan batas maksimal dan minimum petak, karena apabila tidak dibatasi, bisa saja terjadi sebuah kasus ekstrim / edge case dimana petak yang dihasilkan hanya beberapa petak sehingga penyusunan petak pun tidak akan menghasilkan hasil yang memuaskan.

Selain dari aturan jumlah petak, perlu juga dimasukkan batas wilayah yang diinginkan. Tujuan yang diinginkan adalah wilayah natural dengan ruang kosong diatas nya, sehingga perlu disetel pengacakan dibawah ketinggian yang diinginkan sebagai acuan permukaan.

C. Tahap Penyusunan Petak

Pada tahap ini, akan diimplementasikan metode cellular automata. Dengan mengaplikasikan metode cellular automata, setiap petak yang sudah dibuat secara acak di tahap sebelumnya dapat dikelompokan menjadi beberapa kesatuan petak. Tetapi, ini tidak menjamin petak yang dibuat sudah rapi, pada medan yang sangat besar, petak

yang dirapikan dengan metode ini sering kali memiliki beberapa masalah kecil, seperti adanya beberapa petak yang berdiri sendiri atau suatu medak yang memiliki luas area dalamnya mencapai persentasi sangat besar.

Solusi dari permasalahan tersebut adalah dengan cara memodifikasi cellular automata dengan merubah beberapa aturan / *Rule Tweaking* . Sebagai contoh dengan membuat aturan bahwa satu petak harus dikelilingi minimal 5 petak sehingga petak tersebut baru dapat dinyatakan sebagai petak yang final. Tetapi apabila aturan ini diimplementasikan begitu saja, kemungkinan besar akan terjadi kekosongan ruang yang besar di tengah medan. Maka itu, dibuat aturan baru yaitu minimal petak final yang ada pada n langkah dari suatu petak. Sehingga tidak terjadi suatu wilayah yang luas. Demikian juga dengan masalah petak yang tidak berhubungan dengan petak lain, hanya perlu dilakukan perubahan sederhana pada aturan cellular automata. Nilai aturan ini dapat secara fleksibel diatur sedemikian rupa sehingga hasil mencapai sesuai yang diinginkan pembuat.

Selain dari perubahan rule, dapat juga dilakukan perubahan beberapa kali iterasi cellular automata, tiap iterasi hasilnya kemungkinan besar akan berubah, meski perubahannya lama kelamaan akan menipis karena petak makin rapi setiap kali iterasi. Karena itu perlu dilakukan beberapa kali percobaan dalam menentukan jumlah iterasi karena setiap medan memerlukan perbedaan jumlah iterasi sesuai dengan kompleksitasnya.

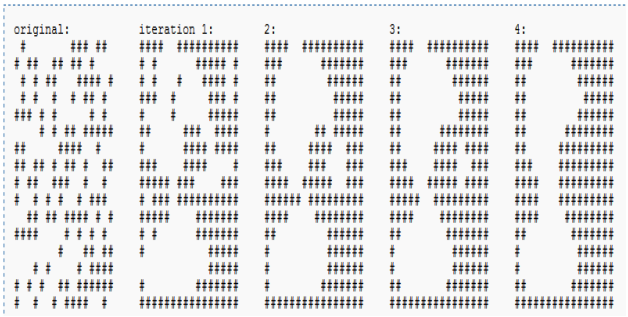


Fig 3. Perbedaan medan tiap iterasi

D. Tahap Implementasi Jenis Petak

Tahap implementasi jenis petak adalah tahapan yang perlu dilakukan karena apabila dalam satu medan hanya ada satu jenis petak, medan akan terasa kurang variatif. Karena itu, tahapan ini cukup penting untuk memvariatifkan medan. Serupa dengan langkah langkah sebelumnya, hal yang pertama dilakukan adalah membuat secara acak petak petak baru dengan atribut berbeda dari yang petak pertama. Perbedaan dengan langkah pertama adalah, pembuatan petak petak baru ini tidak dilakukan di petak kosong, melainkan merubah petak petak yang sudah dirapikan sebelumnya. Dengan cara ini, kontur medan yang sudah dibuat tidak akan berantakan karena bentuk medan tidak akan berubah dari medan sebelumnya.

Setelah pembuatan secara acak, dilakukan penyusunan kembali petak petak yang atribut nya sudah diubah tersebut sehingga dikelompokan menjadi beberapa bagian. Hal ini tidak berat untuk diimplementasikan karena hanya perlu merubah aturan/rule yang telah dibuat sebelumnya dengan sedikit. Ditahapan ini juga tidak perlu dilakukan suatu penyusunan rule secara teliti seperti yang dilakukan di tahap kedua, karena perubahan jenis petak ini tidak perlu sedetil tahapan sebelumnya. Jumlah dari rule bergantung pada banyak tile yang diinginkan, semakin banyak jenis petak, maka akan semakin banyak rule yang harus dikelola. Atau jika ingin merubah secara lebih mudah tetapi lebih menghabiskan waktu komputasi, dapat dilakukan iterasi secara masing masing jenis petak, sebagai contoh petak batu dibuat terlebih dahulu lalu dirapikan, dari petak tanah sisanya, dibuat lagi petak emas, kemudian dirapikan lagi dan seterusnya.

Tahapan ini sebaiknya dipisah dari tahap pertama dan kedua tidak tanpa alasan, jika program diserahkan pengurusan jenis petak sekaligus merapikan kontur medan, cukup banyak kombinasi aturan yang perlu dibuat sehingga program dapat menyusun semuanya dengan sekali iterasi. Kecepatan disini bukanlah sebuah hal yang vital sehingga kita harus memaksa pembuatan aturan aturan yang kompleks tersebut menjadi nyata, karena pembuatan medan hanya terjadi satu kali dan tanpa ada batas waktu yang singkat, tetapi lebih memfokuskan pada kerapian medan tersebut untuk digunakan dan semenarik mungkin.

E. Tahap Perubahan Petak menjadi Vektor

Petak yang sudah rapi dibuat sebelumnya di tahap ini akan diproses dan bagian terluar nya akan diubah menjadi vektor-vektor yang menandakan batas wilayah dari area area medan.

Petak diubah menjadi vektor dikarenakan vektor dapat dengan lebih efektif diproses daripada memproses tiap petak. Contoh nya saat dilakukan pengeload an data, dengan hanya menyimpan vektor yang telah dibuat, proses pengeload an dapat dilakukan dengan jauh lebih cepat dikarenakan vektor dapat dengan mudah mengisi areanya dengan sesuatu yang sejenis. Mengetahui hal ini, maka vektor dibuat semacam tingkatan pengisian, sehingga area jenis petak yang lain tidak ikut tertimpah oleh proses pengisian vektor lainnya yang lebih besar.

Selain untuk mempercepat proses load, hal ini dapat diimplementasikan ke kecepatan program dalam menerima perubahan. Ketika suatu petak mengalami penghapusan, vektor dapat langsung menyesuaikan sehingga set kebiasaan/behavior dari area maupun petak tersebut dapat dengan cepat di update tanpa harus melakukan iterasi.

Hal lain yang dapat didukung oleh sistem ini adalah petak jenis petak dinamis. Petak dinamis adalah petak yang nilainya dapat berubah sesuai dengan kondisi dari medan tersebut. Contoh dari petak dinamis ini adalah fluida, yang dalam kasus umumnya adalah air. Ketika suatu petak yang menopang petak air itu dihapus, maka secara idealnya air akan mengalir kebawah mengikuti alur yang petaknya dihapus. Jika sistem vektor tidak dibuat, maka setiap petak harus melakukan pengecekan terhadap tetangga bawahnya, apakah ia sebuah petak atau petak kosong. Jika ia maka ia harus mengecek tetangga kiri kanan nya lagi, agar petak dapat mengalir ke kiri atau kanan. Bayangkan jika ada

banyak petak air yang dapat berubah secara drastis karena satu perbuatan, dan setiap proses oleh petak menghabiskan 3 kali pengecekan, kompleksitas dari program akan secara drastis meningkat. Dengan menggunakan vektor, program hanya perlu melakukan pemindahan vektor sampai ia menemukan vektor lain, kemudian terbelah menjadi 2 yaitu vektor kiri dan vektor kanan. Petak air yang lainnya juga mudah mengikuti alur vektor yang telah dilalui vektor pendahulu. Akibatnya, kompleksitas program untuk mengurus jenis petak fluida dapat dikurangi dengan banyak.

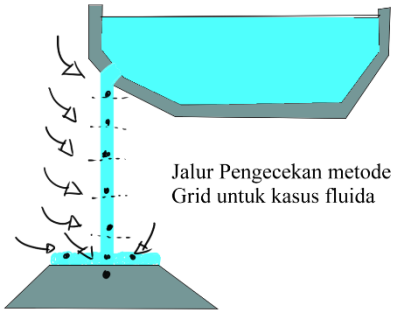


Fig 4. Jalur Pengecekan metode Grid untuk kasus fluida

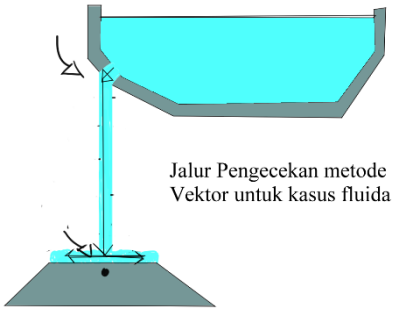


Fig 4. Jalur Pengecekan metode Grid untuk kasus fluida

F. Tahap Visualisasi

Tahap visualisasi adalah tahap finalisasi dari program medan / terrain yang telah kita buat. Tujuan dari tahap ini adalah membuat setiap area vektor dinyatakan sebagai petak petak serupa dengan jenisnya. Area vektor batu akan divisualisasikan dengan batu, air dengan air dan seterusnya. Sehingga setelah melewati proses ini, medan yang sudah dibuat akhirnya dapat direalisasikan sebagai salah satu bentuk pendukung ataupun vital dari suatu game maupun aplikasi lainnya.

G. Kelebihan dan Kekurangan

Setiap program mempunyai kelebihan dan kekurangannya masing masing, demikian pula dengan algoritma ini.

Kelebihan :

1. Fleksibel dengan perubahan
2. Mudah untuk disetel sesuai keinginan dengan merubah aturan/rule
3. Hasil rapi sesuai tingkat kesolid an aturan/rule
4. Dapat mengatasi proses load dengan cukup cepat

Kelemahan

1. Iterasi masih cukup banyak dilakukan sehingga menggunakan waktu yang cukup banyak untuk menyelesaikan program
2. Jika aturan yang dibuat kurang benar maka medan yang tercipta dapat berantakan
3. Aturan harus dianalisa sendiri, tergolong cukup susah untuk debugging.

V. KESIMPULAN

Program ini dapat cukup menyelesaikan pembuatan medan dengan cukup baik dan fleksibel. Tetapi sebagian besar dari algoritma perlu menggunakan rule / aturan yang harus dibuat dengan baik dan teliti karena bila tidak akan merusak bentuk medan. Penggunaan algoritma tidak terlalu berat karena yang perlu di modifikasi dari algoritma cellular algoritma hanya bagian aturannya saja, sedangkan yang lain dapat dipakai secara langsung tanpa banyak perubahan. Pengaplikasian vektor juga mudah untuk diimplementasikan ke dalam program. Waktu yang dibutuhkan dalam program untuk pembuatan medan masih tergolong cukup lama akibat banyak iterasi yang dilakukan program dalam cellular algoritma ditambah dengan pembuatan jenis medan.

VII. UCAPAN TERIMA KASIH

Terima kasih penulis sampaikan kepada Tuhan Yang Maha Esa karena tanpa Nya paper ini tidak akan jadi. Terima kasih juga atas orang tua yang telah membesarkan dan memberi dukungan kepada penulis karena tanpa mereka penulis tidak akan bisa sampai tahap sini. Terima kasih banyak kepada pihak dosen Dr.Ir. Rinaldi Munir dan Drs. Judhi Santoso, M.Sc yang telah mengajarkan materi aljabar geometri kepada penulis dan membimbing penulis hingga dapat menyelesaikan paper ini. Terima kasih kepada pihak Institut Teknologi Bandung yang telah menjadi medium penulis untuk belajar dan berkembang. Terima kasih juga kepada semua pihak lainnya yang telah mendukung pengerjaan paper ini.

REFERENCES

[1] http://mathworld.wolfram.com/CellularAutomaton.html diakses tanggal 15 Desember 2015
[2] http://mathworld.wolfram.com/vonNeumannNeighborhood.html diakses tanggal 15 Desember 2015
[3] http://mathworld.wolfram.com/MooreNeighborhood.html diakses tanggal 15 Desember 2015
[4] http://www.math.mcgill.ca/labute/courses/133f03/VectorHistory.html diakses tanggal 15 Desember 2015
[5] http://www.wesley-kerr.com/caves/ diakses tanggal 15 Desember 2015
[6] http://www.roguebasin.com/index.php?title=Cellular_Automata_Method_for_Generating_Random_Cave-Like_Levels diakses tanggal 15 Desember 2015
[7] John Vince, "Computer Algebra for Computer Graphics,"UK,Bournemouth University, 2008, published.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 16 Desember 2015

Davin Prasetya 13514003