APLIKASI TEORI GRAF DAN POHON DALAM PERINGKASAN MODEL KOTA 3 DIMENSI

David Steven Wijaya

NIM 13505044

Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika,
Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : david_s@students.itb.ac.id

Abstrak

Dalam berbagai bidang, model 3 dimensi seringkali sangat membantu dalam memberikan citra yang komprehensif dan transparan, sehingga dapat memudahkan manusia untuk menginterpretasikannya dibandingkan dengan menggunakan model 2 dimensi. Dengan model 3 dimensi, dapat terlihat dengan jelas dan intuitif seberapa tinggi dan besar sebuah obyek. Hal ini sangat berguna dalam bidang perencanaan kota, mengingat banyaknya kalangan masyarakat yang tidak memiliki pengetahuan yang cukup tentang bagaimana cara menginterpretasikan peta datar (2 dimensi).

Akan tetapi, aplikasi pengembangan permodelan kota 3 dimensi ini mengalami hambatan dalam penggunaannya di internet, terutama karena ukuran data untuk informasi yang direpresentasikan dengan model 3 dimensi sangat besar sehingga mengakibatkan pemborosan *bandwidth* untuk jaringan komunikasi dan memperlambat proses transfer data, sehingga penggunaannya seringkali dibatasi. Oleh karena itu, akhir-akhir ini banyak dikembangkan metode-metode untuk mengkompresi ukuran model kota 3 dimensi. Metode kompresi yang termutakhir dapat meringkaskan hingga 95% dari ukuran model sebenarnya. Hal ini merupakan sebuah terobosan baru yang lebih efektif dibandingkan dengan metoda kompresi standar berbasis tekstual seperti kode *Huffman*.

Makalah ini akan membahas tentang algoritma-algoritma kompresi yang memanfaatkan teori graf dan pohon yang dapat digunakan untuk mengkompresi model 3 dimensi, dan mengulas aplikasinya dalam meringkaskan ukuran data permodelan kota 3 dimensi.

Kata kunci : 3 dimensi, peringkasan, model, kota

1. Pendahuluan

Pengembangan teknologi telah menghasilkan banyaknya penggunaan representasi digital 3 dimensi (3D) di banyak bidang, seperti industri manufaktur, arsitektur, pendidikan, navigasi, pengobatan, simulasi, dan perencanaan kota. Hal ini disebabkan oleh karena penggunaan representasi 3D dapat lebih memberikan gambaran dan bentuk observasi dan interaksi yang lebih intuitif dan komprehensif dibandingkan dengan representasi 2 dimensi, sehingga dapat lebih mudah dimengerti dan dipahami, khususnya bagi kalangan awam.

Dengan teknologi yang tersedia saat ini, membuat model 3D bukanlah hal yang sulit. Model 3D dapat dibuat dengan tingkat akurasi yang sangat tinggi. Beberapa skema telah dihasilkan dalam pembuatan representasi model

3D, dan seringkali dalam skema-skema tersebut, kompatibilitas lebih diutamakan daripada kemangkusan, sehingga ukuran data yang diperlukan untuk representasi 3D sangat besar. Akibatnya, penggunaan model 3D di internet seringkali dibatasi.

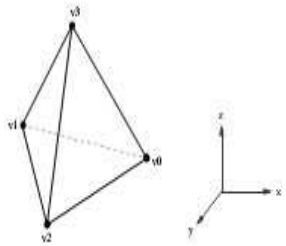
Banyak variasi dari metode peringkas telah dikembangkan untuk meringkaskan model 3D dari waktu ke waktu. Makalah ini akan membahas beberapa metode yang paling berhasil dan menganalisis kontribusi dan perkembangannya sampai saat ini sehingga kita dapat meringkaskan model 3D dengan persentase yang sangat besar. Selain itu akan dibahas bagaimana metode-metode tersebut diaplikasikan dalam meringkaskan model kota 3 dimensi.

2. Algoritma Peringkasan Model 3D

Pengaplikasian dari objek 3 dimensi dalam berbagai bidang dan semakin meluasnya kebutuhan akan permodelan 3 dimensi telah memancing pengembangan metode peringkasan model 3 dimensi yang mangus dan sangkil. Saat ini, peringkasan model 3 dimensi dengan menggunakan kakas-kakas peringkas umum barbasis tekstual seperti WinZip atau Gzip belum dapat menghasilkan hasil yang mencukupi. Di lain pihak, algoritma-algoritma khusus yang tidak berbasis representasi tekstual melainkan isi data 3D yang sebenarnya ternyata mampu meringkaskan hingga 95% (Ewald & Coors, 2005).

Ada bermacam-macam metode untuk merepresentasikan model 3 dimensi, seperti konstruksi geometri padat, representasi segibanyak, dan representasi bidang segitiga (segitiga bertautan / *triangular mesh*). Dalam makalah ini, metode representasi 3D dibatasi pada representasi segitiga bertautan saja mengingat kemangkusannya dalam penggunaan memori dan waktu proses.

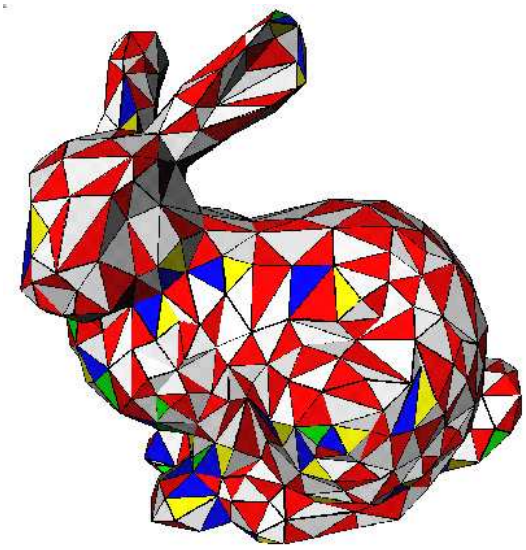
Sebuah segitiga bertautan dapat direpresentasikan dengan *informasi geometrinya*, yaitu lokasi simpul dari tautan, dan *informasi penghubungannya*, yang memberikan informasi tentang hubungan antara tiap segitiga dalam permukaan tautan dengan simpul-simpul yang membatasinya. Atribut-atribut lain seperti informasi warna dan tekstur dapat dimasukkan dalam informasi penghubungan. Segitiga bertaut yang paling sederhana adalah tetrahedron, yang terdiri dari 4 simpul dan 4 sisi segitiga.



simpul vi	kordinat simpul	segitiga ti	simpul-simpul ti
v0	(1, -1, 0)	t0	v0, v2 v1
v1	(-1, -1, 0)	t1	v0, v1, v3
v2	(0, 1, 0)	t2	v1, v2, v3
v3	(0, 0, 1)	t3	v2, v0, v3

Gambar 1 : representasi tetrahedron

segitiga bertaut yang lebih besar biasanya mengandung segitiga sebanyak sekitar dua kali dari jumlah simpulnya, sehingga informasi penghubungan memakan memori dua kali lipat dari informasi geometri. Gambar 2 menunjukkan penggunaan segitiga bertaut untuk membangun model 3 dimensi dari seekor kelinci.



Gambar 2 : model kelinci dengan menggunakan segitiga bertaut

Secara umum, informasi geometri dari segitiga bertaut dapat diringkaskan dengan cara mengurangi ketelitian dari kordinat simpul-

simpulnya, dengan menggunakan kombinasi dari:

- *Pembulatan*, dengan memetakan kordinat simpul dengan nilai yang kurang akurat, menghilangkan angka-angka yang kurang penting
- *Prediksi*, untuk menyandikan vektor-vektor koreksi, menggantikan kordinat absolut simpul
- *Kode entropi*, seperti kode Huffman untuk meminimalisasi ukuran vektor koreksi dengan variasi panjang bit dalam kode

Di lain pihak, informasi penghubungan tidak dapat diringkas dengan metode di atas. Beberapa metode telah dikembangkan untuk meringkaskan informasi penghubungan. Metode-metode yang telah berhasil dan pengembangannya antara lain *Carikan Segitiga (Triangle Strips), Topological Surgery, Edgebreaker, Delphi Compression, Cut Border Machine, Valence Coding, dan Angle Analyzer*

2.1 Carikan Segitiga (Triangle Strips)

Prinsip dari metoda ini adalah menghindari pemrosesan dari simpul yang sama berkali-kali. Aturan dimulai dari segitiga asal (segitiga utama), segitiga baru akan dihasilkan dengan menggabungkan sebuah simpul baru dengan dua simpul lain yang sudah ada, sehingga tiap segitiga akan bersisian dengan segitiga sebelumnya dalam carikan. Dengan menggunakan aturan ini pada semua segitiga dalam segitiga bertaut, maka 1 bit akan cukup untuk memberikan informasi apakah segitiga baru dihasilkan di sebelah kiri atau kanan dari segitiga sebelumnya.

Karena tiap segitiga utama memerlukan 3 simpul untuk membentuknya, maka diusahakan agar jumlah segitiga utama seminimal mungkin, karena sebuah carikan panjang akan lebih baik daripada beberapa carikan pendek. Persoalan untuk mencari sebuah carik yang mengandung semua segitiga adalah persoalan *sirkuit Hamilton*, yang merupakan masalah kelas NP-lengkap.

Penghubungan dari graf segitiga planar dapat dimodelkan dengan pohon merentang. Pohon merentang adalah pohon yang dihasilkan dari graf yang dipotong sisi-sisinya sehingga tidak mengandung sirkuit (Munir, 2006). Ada dua jenis pohon merentang yang digunakan, yaitu pohon merentang simpul (*Vertex Spanning Tree / VST*) dan pohon merentang segitiga (*Triangle Spanning Tree / TST*). VST adalah upaset dari sisi segitiga bertaut yang membentuk pohon dari seluruh simpul dari tautan. Semua sisi yang termasuk dalam VST disebut *cut-edges*, sedangkan sisi yang tidak termasuk dalam VST disebut *hinge-egdes* . TST adalah graf berarah yang dibentuk dari semua muka (segitiga) dan *hinge-ends* pada tautan.

Dengan metode ini, sebuah segitiga bertaut dapat disandikan dengan menyandikan TST dan VSTnya. TST disandikan dengan 2t bit (t = jumlah segitiga), dengan satu bit menyatakan keberadaan anak kiri dan bit yang lain menyatakan keberadaan anak kanan. VST disandikan dengan 2v bit (v = jumlah simpul), dengan satu bit untuk membedakan simpul dengan daun, dan bit yang lain untuk mengindikasikan apakah simpul tersebut adalah anak terakhir dari orang-tuanya. Dengan menganggap bahwa t = 2v, maka keseluruhan bit yang digunakan untuk menyimpan segitiga bertaut adalah

$$2v + 2(2v) = 6v \text{ bit}$$

2.2 Pembedahan Topologis (Topological Surgery)

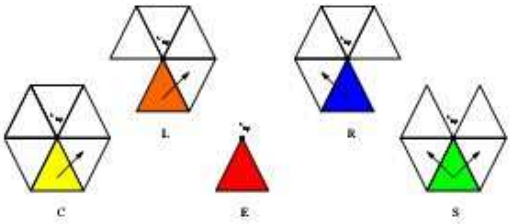
Metode ini dikembangkan oleh Taubin dan Rossignac pada tahun 1998 yang memanfaatkan keteraturan yang dihasilkan dengan menyambungkan kalang konsentris dari VST yang menghasilkan *spiralling VST* dengan jumlah cabang yang lebih sedikit. Metode ini telah digunakan sebagai standar dalam *Three Dimensional Mesh Coding (3DMC)* di MPEG-4

2.3 Edgebreaker

Edgebreaker (Rossignac 1999) dikenal sebagai salah satu dari algoritma peringkas sekali-jalan yang paling mangkus. Seperti dalam pembedahan topologis, *Edgebreaker* membuat *spiralling VST* dan membuat sebuah string sandi bernama *CLERS string* di mana tiap sandi bersesuaian dengan sebuah segitiga di dalam tautan. *CLERS string* mengandung semua informasi yang dibutuhkan untuk algoritma pemulih-ringkas untuk membangun informasi penghubungan dari tautan dengan memasang segitiga baru pada segitiga yang telah dibentuk sebelumnya.

Pada saat peringkasan, tautan dapat terdiri dari beberapa daerah segitiga R*i* yang bersambungan. Tiap R*i* bersesuaian dengan sebuah tautan segitiga yang bersambungan dan bervariasi. Untuk tiap R*i*, bermula dari segitiga asal, *Edgebreaker* akan menghapus segitiga-segitiganya satu per satu dari segitiga sebelumnya ke segitiga berikutnya yang bertetanggaan melalui sisi yang bersesuaian (*gate*). Dengan melihat posisi dari simpul ujung V*tip* (simpul yang tidak berada dalam *gate*) dari segitiga yang baru, *Edgebreaker* akan memisahkan 5 kasus, yaitu C, L, E, R, dan S, yang akan disandikan ke dalam *CLERS string.*

- C, jika V*tip* belum dikunjungi sebelumnya dan bukan bagian dari batas Ri
- L, jika V*tip* berada di kiri *gate*
- E, jika V*tip* berada di kiri dan kanan *gate*
- R, jika V*tip* berada di kanan *gate*
- S, jika V*tip* tidak berada di kiri dan kanan *gate*

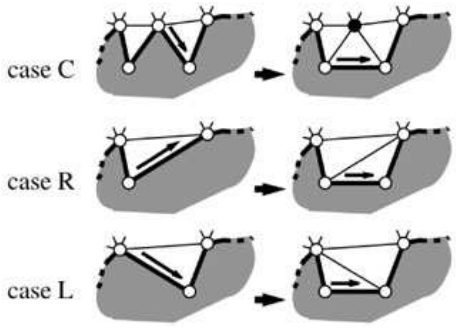


Gambar 3 : Kasus-kasus dalam *Edgebreaker*

Kasus S akan ditangani secara rekursif, dengan mengunjungi segitiga di sebelah kanan, lalu kiri. Selain *CLERS string*, *Edgebreaker* juga menghasilkan referensi simpul, untuk mengindentifikasi posisi simpul dengan kasus C (belum pernah dikunjungi sebelumnya). Gambar 4 menunjukkan urutan proses penelusuran algoritma *Edgebreaker* dalam membuat *CLERS string*.

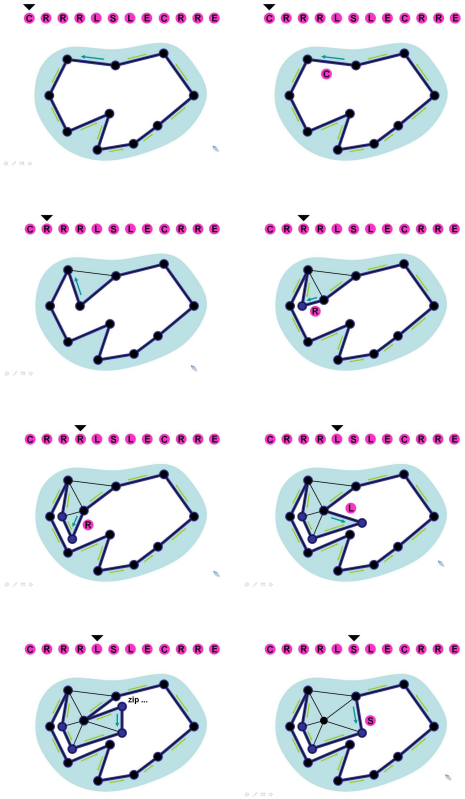
Gambar 4 : rute pembentukan *CLERS string.* Kasus S yang merupakan pencabangan ditangani secara rekursif, dengan mengunjungi segitiga di sebelah kanan terlebuh dahulu.

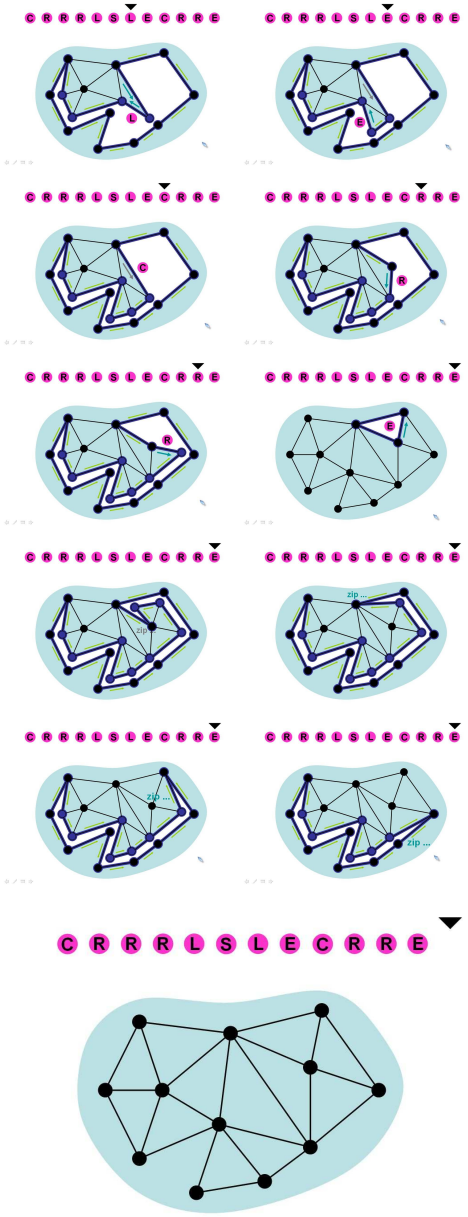
Karena semua simpul selain 3 simpul pembentuk segitiga asal akan disandikan dengan C, maka bit untuk merepresentasikan sandi C adalah bit terpendek (misal C = 0).



Gambar 5 : pembentukan segitiga dari *CLERS string* untuk kasus C, R, dan L

Untuk memulih-ringkas, ada beberapa metode yang dapat digunakan, salah satunya adalah metode *Wrap & Zip (1999)*, yaitu dengan membaca *CLERS string* secara langsung, dan membangun segitiga bertaut darinya dengan mencari informasi penghubungannya dari mambandingkan simpul-simpul yang identik.





Gambar 6 : urutan skema proses pulih ringkas dari *CLERS string* "CRLSLECRRE" yang dihasilkan oleh proses peringkasan *Edgebreaker* dengan menggunakan metode *Wrap & Zip*. Pada kasus 'S', pemasangan simpul dilakukan di sebelah kanan terlebih dahulu

2.4 Delphi Compression

Metode Delphi Compression dikembangkan oleh Rossignac (pembuat *Edgebreaker*) dan Coors pada tahun 2004. Pada dasarnya, metode Delphi Compression sama dengan metode *Edgebreaker*, hanya dalam Delphi Compression, penyandian sandi *CLERS string* tidak dilakukan secara langsung. Metode ini memprediksi tiap sandi *CLERS* berdasarkan bentuk geometri dan ketersambungan dari segitiga yang dikunjungi sebelumya diawali dari segitiga awal.

Dalam tiap tahapnya, algoritma peringkas menyandikan indentifikasi dari V*tip* segitiga yang baru. Hal ini dilakukan dengan melakukan perkiraan terhadap lokasi estimasi g(V*tip*) dengan menggunakan skema prediksi tertentu, misalnya *parralelogram*. Jika lokasi estimasi berada cukup dekat dengan simpul V*x* dari tautan, sehingga jarak antara g(V*tip*) dengan V*x* kurang dari *d* di mana *d* adalah nilai hampiran yang diterima, maka V*x* adalah V*tip*. Jika tidak , sebuah simpul baru akan diperkirakan (kasus C). Jika perkiraan benar, sebuah bit konfirmasi tunggal akan dituliskan sebagai *Appolo sequence* . Jika perkiraan gagal, maka bit koreksi tambahan akan ditambahkan untuk meralat terkaan.

2.5 Cut Border Machine

Metode *Cut Border Machine* dikembangkan oleh Gumhold dan Straber pada tahun 1998. Mirip dengan metode *Edgebreaker*, metode ini menginklusikan segitiga-segitiga ke dalam sebuah perbatasan ketika mentraversal *Spriralling TST*. Dalam tiap tahapnya, ia merekam hubungan kebersisian antara segitiga dengan perbatasan yang sedang dihampiri, yang hasilnya disandikan ke dalam V*ST*. Kode yang dihasilkan cukup untuk mengembalikan keterhubungan dari model yang diringkas.

2.6 Valence Coding

Metode *Valence Coding* dikemukakan oleh Touma dan Gotsman pada tahun 1998. Metode ini juga menyandikan simpul-simpul sepanjang VST. Akan tetapi, ia hanya membedakannya ke dalam dua kasus, bukan lima seperti pada metode *Edgebreaker*, yaitu *split* dan *add*, yang bersesuaian dengan sandi S dan C pada *Edgebreaker*. Metode ini menyandikan kode valensi dari tiap simpul, yaitu jumlah segitiga yang bersesuaian dengannya. Jadi, tiap simbol C diasosiasikan dengan integer valensi, dan simbol S diasosiasikan dengan integer offset. *Decoder* melacak jumlah kecacatan bilangan valensi dari semua simpul di perbatasan. Jika jumlah

kecacatan bilangan valensi di simpul *v* mencapai 1, sebuah segitiga baru akan ditambahkan sehingga *v* menjadi simpul interior (penghubung dua sisi yang baru). Dengan demikian, segitiga L, R, dan E tidak perlu disandikan lagi.

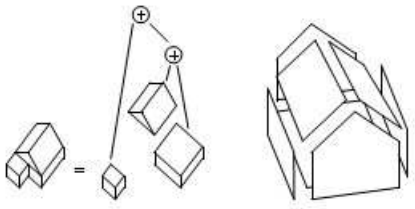
Untuk meminimalisir jumlah kasus *split* (S) yang terjadi, *gate* dipilih dari sisi pembatas yang dihubungkan dengan simpul dengan jumlah sisi bebas yang paling sedikit.

2.7 Analisis Sudut (Angle Analyzer)

Analisi Sudut adalah cara pemilihan *gate* yang lebih baik yang dikembangkan oleh Lee, Alliez, dan Desburn pada tahun 2002. Mereka memilih *gate* menggunakan kriteria geometri dan keterhubungan, mengingat sisi pembatas mempunyai jumlah segitiga bersesuaian yang terbanyak sekaligus membentuk sudut terkecil dengan sisi pembatas yang bersebelahan. Gabungan dari dua kriteria pemilihan *edge* ini secara drastis menurunkan peluang terjadinya kasus S, dan mampu meringkaskan 40% lebih baik daripada metode *valence coding* yang asli.

3. Permodelan Kota 3 Dimensi

Proses membuat model citra komputer 3 dimensi selalu diawali dengan permodelan, yaitu membentuk obyek individu. Bermacam-macam teknik telah dikembangkan, sehingga saat ini terdapat bermacam-macam variasi representasi data 3 dimensi. Representasi data yang paling umum digunakan adalah Constructive Solid Geometry (CSG) dan Boundary Representation (BREP).



Gambar 7 : representasi data 3 dimensi Constructive Solid Geometry / CSG (kiri), dan Boundary Representation / BR (kanan)

CSG menggabungkan obyek-obyek primitif seperti lingkaran, kubus, dan balok menjadi objek kompleks dengan menggunakan operasi boolean. CSG menawarkan cara yang mudah untuk membangun obyek. Akan tetapi, proses membangun model CSG dari data yang sudah ada secara implisit menjadi rumit.

Variasi yang lain, Boundary Representation (BREP) menyimpan informasi bagaimana cara merepresentasikan model 3 dimensi berdasarkan dari bentuk geometri dan pembatas topografis Euler (*Euler Topological Boundaries*). Elemen-elemen dasar dari representasi ini adalah bidang-bidang pembatas, yang disebut *segibanyak*. Pada dasarnya, model 3 dimensi direpresentasikan sebagai kumpulan *segibanyak*. BREP adalah metode representasi yang paling banyak digunakan untuk merepresentasikan model kota 3 dimensi, di mana segibanyak-segibanyak tidak dibuat secara manual, melainkan dibangkitkan secara otomatis dari kumpulan data 2 dimensi, seperti foto dari udara.

Sebuah bentuk 3 dimensi yang khas mempunyai hanya satu atau sejumlah kecil tautan, sedangkan sebuah model kota 3 dimensi dapat mengandung ratusan tautan, yang tersusun dari model permukaan tanah, dan bangunan-bangunan yang berada di atasnya. Model dari permukaan tanah membatasi permukaan bumi dengan udara terbuka di atasnya, yang direpresentasikan dengan segitiga bertaut yang dibatasi luasnya. Model bangunan, termasuk gedung, pohon, dan bentuk-bentuk geometris yang lain yang bukan bagian dari permukaan tanah direpresentasikan dengan segitiga bertaut yang lebih kecil, yang terdiri dari beberapa segitiga dan simpul.

Dalam membuat model kota, terdapat berbagai tingkat keteletian. Terdapat lima tingkat ketelelitian yang telah disepakati, dilambangkan dengan LoD 0 – LoD 4 (LoD = Level of Detail). Semakin tinggi angka di belakang LoD, semakin teliti model yang dibuat. Misalnya, dalam LoD 0 bangunan hanya terdiri dari bentuk geometri putih yang dibatasi dengan garis-garis hitam saja, sedangkan dalam LoD 2 terdapat informasi tambahan seperti model atap dan pewarnaan tekstur.

Selain variasi dalam representasi data 3 dimensi, terdapat variasi dalam format pertukaran data 3 dimensi. Jenis format pertukaran data 3 dimensi yang paling banyak digunakan dalam model kota 3 dimensi adalah Virtual Reality Modelling Languange (VRML) dan Geographic Markup Languange (GML).

VRML didesain untuk kebutuhan internet, di mana model direpresentasikan dengan format teks, di mana simpul dan sisi dari segibanyak dapat dituliskan bersamaan dengan informasi tambahan seperti warna permukaan, tekstur citra, sifat tembus cahaya, kilauan, dan sebagainya.VRML pertama dikembangkan pada tahun 1994, sebagai pengembangan dari format Open Inventor / IRIS Inventor yang dibuat oleh SGI berbasis bahasa pemrograman C++. Versi terbaru dari VRML adalah VRML97 (1997).

Penerus dari VRML adalah Extensible 3D (X3D). Format ini membolehkan komunikasi data 3 dimensi secara *real time* antar aplikasi dan jaringan komunikasi.

Sebagai alternatif dari VRML dan X3D, Geographic Markup Languange (GML) yang dikambangkan oleh Open Geospital Consortium dapat melakukan permodelan, pemindahan, dan penyimpanan dari informasi geografis. GML menggunakan XML (Extensible Markup Languange) sebagai bahasa pemrograman berbasis web untuk menyediakan bermacam-macam obyek yang mendeskripsikan informasi geografis.

4. Meringkaskan Model Kota 3 Dimensi

Dalam bab ini akan dibahas bagaimana sebuah model kota 3 dimensi dapat diringkaskan dengan metode *Edgebreaker*, sebagai metode peringkas yang paling mangkus saat ini. Dalam bagian sebelumnya, telah disebutkan bahwa model kota 3 dimensi secara umum direpresentasikan sebagai Boundary Representation (BREP) dengan beberapa struktur individual. Struktur-struktur tersebut terdiri dari beberapa permukaan segibanyak.

Beberapa jenis algoritma telah tersedia untuk mengubah (memotong) bagian segibanyak dari model BREP untuk menyiapkan data dalam bentuk segitiga bertaut yang merupakan format representasi dasar untuk metode peringkas *Edgebreaker*.

Secara garis besar, tiap struktur dalam model kota 3 dimensi dapat dianggap sebagai obyek-obyek yang terpisah atau sebuah kumpulan segitiga yang terhubung dalam suatu daerah. *Edgebreaker* mempunyai kecenderungan untuk menangani semua struktur model kota dalam satu tautan, sehingga akan menghasilkan sebuah *CLERS string*. Kecenderungan ini mengakibatkan adanya dua jenis pendekatan untuk meringkaskan model kota 3 dimensi dengan beberapa struktur independen yang berbeda-beda.

Dalam pendekatan pertama, obyek-obyek dapat dianggap sebagai satu kesatuan sendiri sehingga dapat diringkaskan satu-satu. Kemudian obyek-obyek tersebut ditransfer (digabungkan) secara independen satu dengan yang lain, sehingga peringkasan sebuah model kota dilakukan dengan meringkaskan strukturnya satu-satu. Akan tetapi, pendekatan ini mempunyai satu kelemahan besar. Penyandi entropi yang berfungsi untuk meringkaskan vektor-vektor koreksidan atribut-atribut tautan seperti warna dan sebagainya akan dijalankan tiap kali meringkaskan struktur, sehingga kemangkusannya berkurang. Sebagai contoh, sebuah penyandi aritmatik misalnya dapat menyandikan informasi geometri dari sebuah kubus dengan 8 simpul ke dalam 54 byte data. Jika lima buah kubus tersebut diringkaskan secara bersamaan, maka jumlah data yang dihasilkan adalah 180 byte. Hasil ini lebih kecil dibandingkan dengan meringkaskan kubus tersebut satu-satu (5 * 54 = 270 byte), karena sisi-sisi dan simpul-simpul yang digunakan bersama hanya diringkaskan sekali saja. Dari contoh sederhana ini, penyandi aritmatik bekerje 150% lebih banyak untuk rangkaian yang mempunyai elemen 5 kali lebih banyak daripada rangkaian yang lain.

Untuk mengatasi masalah ini, model kota dapat dianggap sebagai sebuah tautan yang besar, di mana tiap struktur ditetapkan sebagai sebuah daerah sendiri yang tersambung, sehingga dapat diringkaskan oleh *Edgebreaker* sebagai sebuah tautan. Akan tetapi, batas pemisah antar struktur menjadi tidak jelas karena sebuah struktur dapat terdiri dari beberapa daerah yang tersambung. Akibatnya, algoritma pemulih-ringkas *Edgebreaker* tidak akan dapat menerjemahkan daerah dari struktur yang bersesuaian tanpa informasi tambahan, seperti jumlah dari daerah yang tidak terhubung atau jumlah segitiga tiap strukturnya.

Untuk pendekatan yang lain, hanya informasi selain informasi penyambungan yang dapat diringkaskan secara bersamaan. *CLERS string* mengandung semua informasi yang dibutuhkan

untuk membagi ulang data yang diringkaskan bersamaan tadi pada saat proses pulih-ringkas dan menambahkan tiap struktur dengan nilai informasi yang cocok. Hal ini menghapus kebutuhan untuk untuk memberikan informasi tambahan seperti jumlah segitiga tiap strukturnya atau daerah-daerah yang tidak terhubung. Ilustrasi dari pendekatan ini diberikan di paragraf selanjutnya.

Pada langkah pertama, peringkas *Edgebreaker* menyandikan posisi 3 simpul dari semua segitiga awal yang dihampiri untuk setiap daerah Ri dari tautan. Jika sebuah simpul ditemukan saat pelintasan tautan, sandi C ditambahkan ke *CLERS string.* Tiap sandi S menandakan split, dan diakhiri dengan sandi E. Sandi E tambahan ditambahkan untuk menandai akhir dari daerah yang sedang dilintasi. Jika algoritma peringkas menemukan simpul baru yang belum dikunjungi setelah melalui akhir dari daerahnya, simpul itu diasumsikan sebagai daerah lain yang tidak tersambung, lalu menyandikan tiga simpul segitiga asal baru, dan melanjutkan lintasan. Hasilnya, walaupun tidak ada sandi baru dalam *CLERS string* selain C, L, E, R, dan S, masih dapat ditentukan akhir dari daerah yang terhubung dan awal dari daerah berikutnya, cukup dengan memperhatikan lokasi sandi S dan E. Pendekatan ini juga mampu mengitung jumlah simpul dari tiap struktur dengan membaca *CLERS string.* Dimulai dari indeks simpul di 3 (jumlah simpul segitiga awal), jika dibaca sandi C (simpul baru), indeks simpul bertambah satu. Jika dibaca sandi S(split), maka keberadaannya dicatat hingga ditemukan sandi E(akhir dari split). Jika sandi E dibaca tanpa keberadaan S yang bersesuaian (akhir dari daerah, dan awal daerah baru), maka indeks simpul ditambah 3 karena diciptakan segitiga awal baru.

Dari jumlah simpul dalam struktur yang didapat dari proses pengolahan *CLERS string* di atas, simpul-simpul yang bersesuaian akan diletakkan dalam strukturya dan proses pulih-ringkas dapat dilakukan.

Dengan cara yang mirip, jumlah segitiga dapat dihitung dengan menelusuri *CLERS string*, dan atribut tambahan seperti warna dapat diringkaskan dengan cara yang sama seperti saat meringkaskan simpul.

Dari hasil pengamatan tentang cara peringkasan model kota 3 dimensi dengan menggunakan metode *Edgebreaker* di atas, dapat ditentukan atribut (byte) yang dibutuhkan untuk merepresentasikan sebuah model kota yang lengkap dengan informasi geometri dan penyambungannya tanpa atribut tambahan, yaitu senarai yang berisi representasi *CLERS string* dari tiap struktur dalam bit dan sebuah senarai bit tunggal untuk semua sandi entropi dari vektor-vektor koreksi model yang disandikan. Jika kuantisasi dan prediksi geometri digunakan pewarnaan ikut diringkaskan, maka atribut lain harus digunakan. Tabel 1 menggambarkan sebuah Abstract Data Type (ADT) yang digunakan untuk merepresentasikan model kota 3 dimensi lengkap dengan struktur-struktur yang dapat dibedakan dan informasi warna untuk digunakan oleh peringkas *Edgebreaker* bersamaan dengan peringkas geometris (*prediction, quantization, dan arithmatic coding*). ADT ini akan digunakan dalam bagian berikutnya, yang membahas eksperimen peringkasan model kota 3 dimensi.

Model kota 3 dimensi yang diringkaskan	
Prediction scheme	Int
*CLERS string*s	Byte[ ][ ]
Vektor koreksi	Byte [ ]
Informasi pewarnaan	Byte [ ]
Informasi kuantisasi	Byte [ ]

Tabel 1 : ADT dari model kota 3 dimensi yang diringkaskan dengan metode *Edgebreaker* dengan prediksi geometri dan kuantisasi, disertai vektor koreksi. Jumlah segitiga dan simpul dari tiap struktur dapat ditentukan dari *CLERS string*, sehingga tidak perlu disertakan.

5. Eksperimen Peringkasan Model Kota 3D

Di bagian ini akan dibahas hasil eksperimen yang dilakukan oleh F. Eppinger dan V. Coors, dosen di Stuttgart University of Applied Sciences, Departement of Geometries, Computer Science and Mathematics. Mereka melakukan eksperimen peringkasan yang menggunakan bagian dari model kota 3 dimensi dari kota Stuttgart, Jerman yang telah tersedia dalam LoD 1 dengan bentuk atap yang dapat dibedakan.

Model kota yang digunakan dalam eksperimen tersedia dalam format VRML, dengan tiap

struktrur didefinisikan sebagai "kumpulan permukaan yang diberi indeks" yang berbeda-beda tersusun atas beberapa permukaan segibanyak pembatas (*Boundary Representation*). Informasi pewarnaan diletakkan dalam tiap permukaan, yang berarti atribut warna disisipkan sebagai informasi dari tiap segibanyak.

Agar model kota 3 dimensi dapat diringkaskan dengan metode *Edgebreaker*, maka mode VRML dengan representasi segibanyak tersebut diubah terlebih dahulu menjadi VRML dengan representasi segitiga bertaut dengan menggunakan implementasi *Earcutting* (ElGindy, 1993), dengan tidak mengubah informasi pewarnaan yang digunakan.

Sebuah hasil implementasi dari metode *Edgebreaker* telah tersedia dalam bentuk Java di bawah lisensi dari GNU Lesser General Public License (dipublikasikan oleh Free Software Foundation). Implementasi ini dikembangkan agar dapat menangani pendekatan yang telah dijelaskan dalam bagian sebelumnya dari makalah ini, yang menggunakan lingkungan peringkasan seperti yang digambarkan pada Diagram 1.

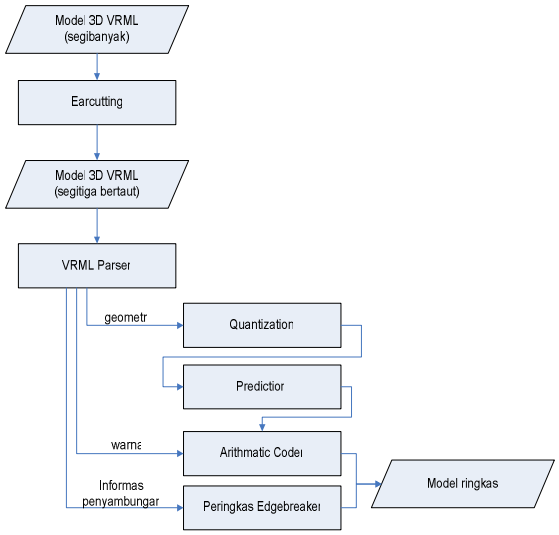
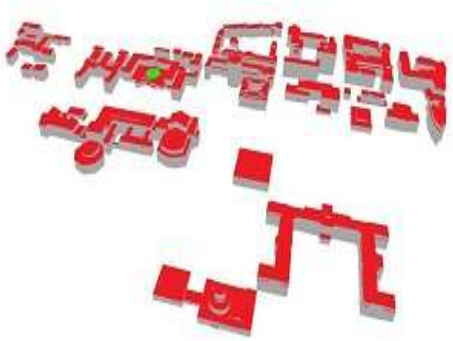


Diagram 1 : Lingkungan eksperimen peringkasan - informasi geometri dan pewarnaan diringkaskan dengan kombinasi dari *quantization*, *parallelogram prediction*, dan *arithmatic coder*, dengan kapasitas tertentu sehingga tidak ada pengurangan ketelitian yang terlihat. Sebuah aplikasi *Edgebreaker* digunakan untuk meringkaskan informasi penghubungan (penyambungan) dari model.

Dua daerah dari model berwarna kota 3 dimensi dari Stuttgart telah dipilih untuk menjadi sampel eksperimen, yaitu sebuah model dari bagian kecil dari jantung kota Stuttgart dengan 111 struktur dan ukuran rata-rata per strukturnya sebesar 3.93 kb (gambar 8) dan sebuah model daerah permukiman kota yang lebih besar dengan 6771 struktur dan ukuran rata-rata per stukturnya sebesar 2.02 kb (gambar 9). Model jantung kota memerlukan 435.6 kb dengan format VRML, dan model daerah pemukiman memerlukan 13.694,5 kb.



Gambar 8 : model 3 dimensi dari bagian jantung kota Stuttgart, mengandung 111 stuktur dengan total 5.959 simpul dan 11.474 segitiga. Ukuran dalam format VRML : 435,6 kb

Gambar 9 : model 3 dimensi dari tempat permukiman di Stuttgart, mengandung 6.771 struktur dengan total 153.697 simpul dan 280.306 segitiga. Ukuran dalam format VRML : 13.694 kb

Dengan menggunakan alat peringkas komersial Gzip, model jantung kota diringkaskan hingga 20,2% (87,8 kb) dan model daerah pemukiman diringkaskan hingga 13,6% (1.859,7 kb) dari ukuran aslinya.

Dengan menggunakan metode *Edgebreaker*, model jantung kota yang diringkaskan hingga 7,2% (31,03 kb) dan model tempat pemukiman diringkaskan hingga 8,1% (1.108,04 kb) dari

ukuran file aslinya. Tabel 2 menunjukkan perbandingan ukuran dari total ukuran file hasil ringkasan menggunakan metode *Edgebreaker* dibandingkan dengan peringkasan menggunakan Gzip.

Keterangan	Jantung kota	Pemukiman
Jumlah struktur	111	6.771
Jumlah simpul	5.959	153.697
Jumlah segitiga	11.474	280.306
Rata-rata kb / struktur	3,93	2,02
Ukuran asli (VRML)	435,6 kb	13.694 kb
Ukuran ringkasan Gzip (% file asli)	87,8 kb (20,2%)	1.859,7 kb (13,6%)
Ukuran ringkasan *Edgebreaker* (% file asli)	31,03 kb (7,2%)	1.108,04 kb (8,1%)
Selisih ukuran ringkasan	56,77 kb	751,66 kb

Tabel 2 : perbandingan ukuran file hasil ringkasan menggunakan metode Edgebreaker dan GZip

Dalam implementasi *Edgebreaker* ini, *CLERS string* disandikan dengan ketentuan sebagai berikut:

- ‘C’ : 0
- ‘L’ : 110
- ‘E’ : 111
- ‘R’ : 101
- ‘S’ : 100

Seperti yang telah disebutkan dalam bagian 2, hasil penyandian yang dihasilkan akan bervariasi sesuai dengan metode penyandian yang digunakan. Dalam kasus ini, digunakan metode penyandian asli dari *Edgebreaker*.

CLERS string dari ringkasan model jantung kota tersusun atas 49%‘C’, 9%‘L’, 7%‘E’, 29%’R’, dan 6%’S’, menghasilkan kurang lebih 2,01 bit untuk tiap segitiga atau 2,81 kb total dari ukuran informasi penghubungan, yaitu hanya 9,1% dari total keseluruhan ukuran file ringkasan. Distribusi sandi untuk model tempat pemukiman tidak jauh berbeda dengan yang didapat dari model jantung kota. Pada kasus tempat pemukiman, total ukuran yang dihasilkan dari informasi penghubungan adalah 68,69 kb, yaitu 6,2% dari total file hasil ringkasan. Hal ini dapat dijelaskan dengan melihat banyaknya jumlah data yang digunakan untuk menyimpan informasi kuantisasi dari tiap struktur, yang membutuhkan 13,3% dari total ukuran file hasil ringkasan model jantung kota dan 22,9% dari total ukuran file hasil ringkasan model tempat pemukiman. Jika vektor-vektor koreksi tidak dimasukkan dalam gambar, maka ukuran informasi geometri akan menempati urutan pertama dari ukuran file hasil ringkasan model.

Penggunaan memori yang lain yang masuk dalam ukuran file hasil ringkasan berasal dari string pengalamatan dari struktur yang diringkaskan dan beberapa slot (seperti skema prediksi yang digunakan dalam peringkasan) yang diperlukan untuk proses pulih-ringkas. Walaupun hanya meyusun 2,3 – 3,3% dari keseluruhan ukuran file ringkasan, informasi-informasi ini dapat dihilangkan dari file ringkasan jika pengalamatan dianggap tidak penting dan skema prediksi akan diimplementasikan dengan manual pada saat proses pulih-ringkas. Tabel 3 menunjukkan distribusi ukuran file hasil ringkasan model jantung kota dan tempat pemukiman yang menggunakan metode *Edgebreaker*.

Distribusi (% file ringkasan)	Jantung kota (31,03 kb)	Pemukiman (1.108,04 kb)
CLERS string	2,81 kb (9,1%)	69,05 kb (6,2%)
Vektor koreksi	18,74 kb (60,4%)	572,26 kb (51,7%)
Informasi kuantisasi	4,13 kb (13,3%)	253,76 kb (22,9%)
Informasi pewarnaan	4,62 kb (14,9%)	176,0 kb (15,9%)
Lain-lain	0,73 kb (2,3%)	36,97 kb (3,3%)

Tabel 3 : distribusi ukuran file hasil ringkasan model kota 3 dimensi (dalam kilobyte). Skema sederhana yang digunakan untuk menyandikan *CLERS string* membuat ukuran penyimpanan dari tiap segitiga menjadi sekitar 2 bit saja.

Dengan menggunakan laptop IBM Thinkpad T41p 1700 MHz 1 GB RAM dalam lingkungan peringkasan *Eclipse Development*, proses peringkasan membutuhkan waktu 309 ms untuk model jantung kota dan 8.078 ms untuk model tempat pemukiman, sedangkan proses pulih-ringkas membutuhkan waktu 245 ms untuk model jantung kota dan 6.588 ms untuk model pemukiman.

Hasil tersebut mendemonstrasikan bahwa jumlah dari struktur berpengaruh pada distribusi penyimpanan data di file hasil peringkasan model. Hal ini disebabkan oleh banyaknya informasi kuantisasi yang harus disimpan untuk tiap struktur. Akan tetapi, bagaimanapun juga metode *Edgebreaker* mampu mengungguli peringkas komersial berbasis tekstual sebanyak 35% hingga 60%, tergantung jumlah dan ukuran dari struktur dalam model.

Untuk model-model dengan dengan koordinat integer, yang tidak membutuhkan kuantisasi, hasil yang didapat akan lebih baik.

6. Kesimpulan

Tanpa memandang kecepatan bandwidth dari jaringan telekomunukasi yang terus meningkat, mengakses model 3 dimensi lewat internet seringkali dibatasi karena besarnya jumlah data yang diperlukan untuk representasi 3 dimensi tersebut. Akan tetapi, perkembangan terakhir dari metode peringkasan model 3 dimensi berbasis segitiga bertaut telah berhasil menemukan cara untuk mengurangi ukuran file dengan representasi 3 dimensi secara drastis tanpa mengurangi detail dari obyek yang diringkas secara nyata.

Walaupun segitiga bertaut bukanlah representasi yang paling umum untuk merepresentasikan model kota 3 dimensi, format ini dapat dihasilkan dengan melakukan konversi dari format-format lain, sehingga dapat digunakan untuk diringkas dengan algoritma peringkasan termutakhir saat ini, *Edgebreaker*.

Algoritma *Edgebreaker* merupakan algoritma yang paling mangkus dan sangkil saat ini untuk meringkaskan model 3 dimensi dengan representasi segitiga bertaut. Bahkan untuk model kota yang besar ukurannya dapat diringkaskan menjadi sepersepuluh dari ukuran aslinya jika algoritma ini diterapkan pada informasi penghubungannya.

Pendekatan dengan algoritma *Edgebreaker* dalam makalah ini mengurangi ukuran dari model kota 3 dimensi dari 435,6 kb menjadi kurang dari 31 kb tanpa merusak kualitas visualnya secara nyata. Hal ini merupakan penurunan sebesar 92% dan merupakan peningkatan sebesar 35% dari hasil yang dicapai dengan meggunakan kakas peringkas umum

Walaupun *CLERS string* dari file hasil ringkasan hanya menyusun sekitar 8,5% dari total ukuran file, ukuran dari *CLERS string* ini dapat lebih diringkas lagi dengan menggunakan metode-metode lain seperti *Delphi Compression* yang menggunakan prediksi penghubungan berbasis geometri digunakan untuk menringkaskan model 3 dimensi yang bersangkutan.

Keuntungan lain yang didapatkan dari meringkas model 3 dimensi berdasarkan dari isi file representasinya dibandingkan dengan meringkas berdasarkan representasi tekstualnya adalah tingkat ketelitian dan detail dari obyek yang diringkasnya, sehingga ukuran penyimpanan data 3 dimensi dapat disesuaikan sesuai dengan kriteria tertentu seperti kecepatan bandwidth yang tersedia.

Selain memiliki keunggulan-keunggulan, metode *Edgebreaker* juga memiliki kerugian-kerugian. Salah satu kerugian dari metode *Edgebreaker* adalah penggunaan VRML sebagai satu-satunya informsi geometri akan disandikan. Akan tetapi, teknik peringkasan *Edgebreaker* jika dikombinasikan dengan penyandi biner XML akan dapat dengan mudah digunakan untuk menghasilkan model kota padat yang sesuai dengan format CityGML.

Akhir kata, ukuran data yang kecil berarti akses yang cepat dan download yang cepat pula, yang pada akhirnya membuka peluang baru bagi aplikasi-aplikasi online yang baru dan pengembangan-pengembangan lain yang bermanfaat di semua bidang, tidak hanya di bidang perencanaan kota.

Daftar Pustaka

Coors, V & Rossignac, J. (2004). Delphi : Geometric Based Connectivity Prediction in Triangle Mesh Compression. *The Visual Computer*

Eppinger & Coors, V. (2006). Compressing 3 Dimensional Urban Models. Stuttgart University of Applied Sciences, Department of Geomatics, Computer Science and Mathematics

Ewald, K & Coors, V. (2005). Composed 3D Urban Models in Internet Based E-Planning. *1st International Workshop on Next Generation 3D City Models, Bonn, Germany*

Lee, H & Alliez, P. (2002). Angle Analyzer : A Triangle-Quad Mesh Codec

Munir, Rinaldi. (2006). Diktat Kuliah IF2152 – Matematika Diskrit edisi IV Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung

Rossignac, J. (1999). *Edgebreaker* : Connectivity Compression for Triangle Meshes. *IEEE Transactions on Visualization and Computer Graphic*

X3D. (2005). Extensible 3D (X3D). *http://www.web3d.org/x3d/specifications. Tanggal akses : 26 Desember 2006, 15.00*