Pemanfaatan Teori Graf dalam Mengoptimalisasi Steganografi pada Objek 3D

Fakhri 13510048
Program Sarjana Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13510048@std.stei.itb.ac.id

***Abstrak*—Steganografi adalah bentuk komunikasi rahasia sehingga hanya pengirim dan penerima pesan saja yang tahu akan adanya komunikasi tersebut (Cox, 2008). Komunikasi dilakukan menggunakan media yang memiliki sebuah makna tersendiri yang berbeda dengan pesan rahasia, lalu pesan rahasia dimasukkan ke dalam media tersebut. Di dalam konteks informatika, steganografi diimplementasikan dalam media gambar, audio, video, hingga objek 3D. Pada objek 3D pesan rahasia disisipkan melalui pergeseran *vertex*. Namun terdapat masalah dalam implementasinya, yaitu dalam pemilihan *vertex* dan metode *traversal* di dalam objek 3D tersebut. Teori graf dapat menjadi solusi melalui perepresentasian objek 3D dan pemanipulasiannya.**

***Kata kunci*—Steganografi, *Vertex*, *Edge*, *Face*, Graf.**

I. PENDAHULUAN

Dalam dunia *digital*, komunikasi dilakukan berbasiskan jaringan internet. Bentuk komunikasi secana teknis terjadi melalui pertukaran data dari pihak pertama selaku pengirim pesan kepada pihak kedua selaku penerima pesan. Komunikasi yang terjadi secara umum dilakukan menggunakan perangkat lunak tertentu yang secara tidak langsung dapat memantau komunikasi sehingga dapat dianggap sebagai pihak ketiga.

Secara umum terdapat dua proses terkait privasi yang terdapat dalam komunikasi. Pertama penyedia layanan komunikasi yang menghargai privasi penggunanya dengan secara automatis melakukan enkripsi pesan pada saat pesan dikirim dan didekripsi ketika pesan diterima. Kedua adalah keadaan sebaliknya, yaitu tidak digunakannya proses enkripsi-dekripsi pada transfer pesan sehingga komunikasi apapun yang terjadi dapat dibaca. Dari dua bentuk komunikasi ini, apabila terjadi komunikasi rahasia, data berharga yang dikirimkan baik terenkripsi maupun tidak dapat ditangkap/disadap. Apabila hasil sadap terenkripsi, akan ada usaha pendekripsian untuk memperoleh informasi. Hal ini tentu tidak diinginkan oleh pihak pertama dan kedua, apalagi jika pihak ketiga berhasil mendekripsi dan mendapatkan informasi atau data yang tersimpan didalamnya. Oleh karena itu, sebagai solusi langsung dari masalah ini diperlukan metode pengenkripsian data yang sangat kuat dan terjamin.

Solusi lain yang juga dapat menjawab permasalahan dalam menjaga privasi dan kerahasaian transfer data selain pengenkripsian pesan adalah penggunaan steganografi. Steganografi dilakukan dengan menyisipkan pesan rahasia pada media pembungkus (*cover* object) yang memilikni makna tersendiri menggunakan kunci rahasia yang hanya diketauhi pihak pertama dan kedua. Media pembungkus yang paling umum digunakan adalah gambar. Melalui steganografi pihak ketiga tidak dapat menyadari adanya komunikasi rahasia yang terjadi karena transfer data rahasia dibungkus dengan media yang telah memiliki makna tersendiri, sebagai contohnya adalah gambar objek.

Steganografi terus dikembangkan untuk menjawab kebutuhan privasi akan data yang terus berkembang. Perkembangan dunia *digital* mengakibatkan data yang biasa digunakan semakin besar ukurannya sehingga penggunaan media gambar berpotensi tidak lagi dapat merahasiakan transfer data yang kian membesar ukurannya. Perkembangan steganografi dilakukan untuk objek lain seperti audio dan video. Perkembangan tersebut telah sampai pada pemanfaatan media yang jauh berbeda secara struktur datanya, yaitu objek 3D.

Steganografi menggunakan media gambar, audio, dan video pada dasarnya melakukan modifikasi bit dalam penyisipan bit pesan rahasia. Metode yang popular adalah penyisipan pada *Least Significant Bit* (LSB). Lalu kunci rahasia digunakan sebagai parameter untuk membangkitkan bilangan random dalam memilih posisi pixel ataupun waktu pada audio yang akan diubah bitnya. Sedangkan pada steganografi yang menggunakan objek 3D, metode LSB dan penggunaan kunci tidak dapat digunakan akibat struktur data yang berbeda.

Pada Objek 3D, secara umum penyisipan dilakukan dengan cara mengubah posisi koordinat suatu *vertex* yang dibandingkan terhadap sebuah *edge* pada *face* yang sama. Kunci rahasia digunakan dalam menentukan *vertex* bagian mana yang dipilih untuk diubah posisinya. Terdapat permasalahan yang mendasar, yaitu :

1. Bagaimana penrapan *diffusion* sebagai aspek wajib keamanan dan kaitannya dengan kunci?
2. Bagaimana memilih *vertex* agar *vertex* pada *edge* atau *vertex* yang telah digeser tidak terpilih sehingga mengubah nilai yang telah disisipkan?

Hal ini menjadi permasalahan yang dapat diselesaikan menggunakan teori graf.

II. DASAR TEORI

A. Graf

Permasalahan dalam dunia nyata yang dimengerti oleh manusia dapat dipecahkan menggunakan komputer apabila permasalahan tersebut dimodelkan dengan benar dan dimengerti oleh system komputer. Model ini adalah bentuk representasi permasalahan sebenarnya menjadi pemodelan matematis ataupun pemodelan yang sesuai sistem komputer. Salah satu model representasi pada pemrosesan menggunakan komputer adalah graf.

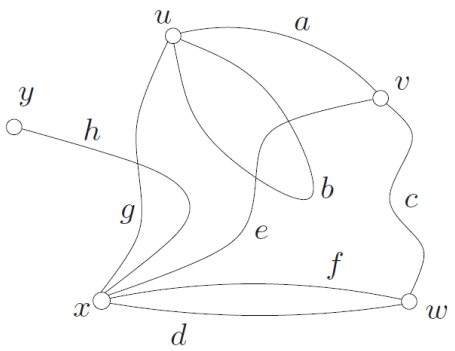
Graf dapat dibayangkan sebagai kumpulan dari titik-titik yang disebut simpul/*node/vertex*, yang sebagiannya terhubung melalui suatu segmen garis yang disebut sisi/*edge* (Levitin, 2011). Secara matematis sebuah graf *G* adalah pasangan dari (*V(G), E(G)*) dengan *V(G)* adalah suatu set *vertex* pada graf *G* dan *E(G)* adalah suatu set *edge* yang menghubungkan *vertex* dalam graf *G*. Sebagai contoh yaitu persamaan 1, 2, dan 3.

$$G = \big(V(G), E(G)\big) \tag{1}$$

dengan

$$V(G) = \{u, v, w, x, y, z\} \tag{2}$$
$$E(G) = \{a, b, c, d, e, f, g\} \tag{3}$$

Penggambaran atau bentuk visual dari graf terdapat pada gambar 1. Pada gambar ini, graf yang tergambar tersusun atas 6 *vertex* bersimbol lingkaran dan 7 *edge* yang menjadi penghubung antar *vertex*.



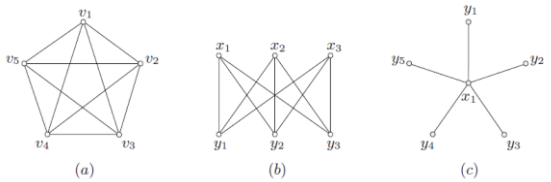
Gambar 1 : Contoh graf sesuai persamaan 1,2, dan 3
Sumber : Cormen (2009)

Penggambaran suatu graf tidak dapat dipastikan ke dalam satu bentuk, karena ada sangat banyak kemungkinan untuk posisi dari suatu *vertex* yang berimbas kepada *edge*. Akan tetapi pola kombinasi *edge* dan *vertex* yang tergambar dapat diklasifikasi. Klasifikasi untuk pola di dalam sutu graf terdiri atas terbentuknya pola berikut:

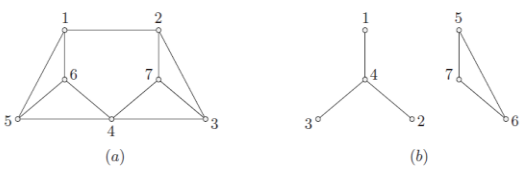
1. *loop*, terbentuk ketika terdapat sebuah *edge* yang menghubungkan dua buah *vertex* yang sama atau *vertex* asal dan *vertex* tujuannya sama. Pada gambar 1 dapat dilihat pada *edge b.*
2. sisi paralel (*parallel edges*), terbentuk ketika ada dua *vertex* yang yang dihubungkan lebih dari satu *edge*. Pada gambar 1 dapat dilihat pada *edge d* dan *edge f.*

Selain pola di dalam graf, suatu graf juga dapat diklasifikasikan atas tipe – tipe tertentu. Seperti yang tergambarkan pada gambar 2 dan gambar 3, pengklasifikasian tersebut antara lain :

1. graf sederhana (*simple*), graf yang tidak mengandung *loop* dan sisi paralel di dalamnya
2. graf komplit, graf yang seluruh *vertex* nya terhubung satu sama lain
3. graf bipartite, graf yang dapat dibentuk menjadi dua bagian tanpa adanya keterhubungan antar *vertex* yang berada dalam satu bagian
4. graf bintang, memiliki nama lain graf bipartite komplit karena satu dari dua bagiannya hanya terdiri atas satu *vertex*
5. graf terhubung, graf dengan suatu *vertex* nya dapat mengakses seluruh *vertex* lain pada graf tersebut melalui keterhubungan antar *vertex* melalui *edge*. Sebaliknya adalah graf tidak terhubung.
6. graf planar, graf yang dapat digambarkan tanpa ada *edge* yang beririsan
7. graf berarah, graf dengan *edge* yang memiliki arah sehingga *edge* yang menghubungkan *vertex* 1 ke 2 tidak sama dengan kebalikannya
8. graf berbobot, apabila *edge* memiliki nilai tertentu.



Gambar 2 : (a) Graf Komplit, (b) Graf Bipartite, (c) Graf Bintang
Sumber : Bondy (2008)



Gambar 3 : (a) Graf Terhubung, (b) Graf Tidak Terhubung
Sumber : Bondy (2008)

Lintasan (*path*) adalah graf sederhana dengan *vertex* yang dapat disusun dalam suatu urutan linear sedemikian rupa sehingga urutan tersebut tersusun atas urutan dua *vertex* yang terhubung (Bondy, 2008). Apabila lintasan dapat terbentuk dengan *vertex* awal dan akhir yang sama, maka lintasan tersebut dapat disebut sebagai *cycle*.

Penggambaran graf bukanlah model yang dapat diolah oleh komputer. Perlu dilakukan proses representasi menjadi struktur data computer. Dua bantuk utama representasi graf dapat menggunakan matrix ataupun *list/linked-list* sebagai struktur datanya. Dua representasi itu adalah :

1. Beririsan (*Incidency*)
   Merepresentasikan keterhubungan antar *edge* dan *vertex*. Struktur data yang digunakan hanyalah matriks. Contoh tergambarkan pada gambar 4(a). Index baris matriks berarti *edge* dan kolom matriks berarti *vertex*. Elemen matriks yang bernilai 1 menandakan adanya hubungan antara *edge* dan *vertex*. Apabila nilai lebih besar dari 1, maka terdapat *loop*. Sedangkan jika bernilai 0, menandakan tidak adanya keterhubungan.

2. Ketetanggaan (*Adjecency*)
   Merepresentasikan keterhubungan antar *vertex*. Contoh tergambarkan pada gambar 4(b). Untuk struktur *list*, tiap elemen list yang terhubung menandakan *vertex* dengan nilai yang sama terhubung. Lalu untuk matriks, elemen dengan nilai 1 menandakan keterhubungan antara *vertex* dengan nilai yang sama dengan dua nilai index matriks. Elemen bernilai lebih besar dari 1 menandakan adanya sisi paralel antar dua *vertex*. Sedangkan elemen bernilai 0 menandakan tidak adanya keterhubungan,

	a	b	c	d	e	f	g	h
u	1	2	0	0	0	0	1	0
v	1	0	1	0	1	0	0	0
w	0	0	1	1	0	1	0	0
x	0	0	0	1	1	1	1	1
y	0	0	0	0	0	0	0	1

(a)

	u	v	w	x	y
u	2	1	0	1	0
v	1	0	1	1	0
w	0	1	0	2	0
x	1	1	2	0	1
y	0	0	0	1	0

(b)

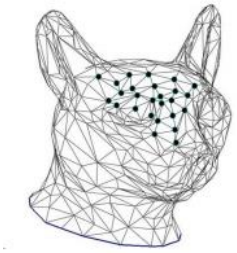
Gambar 4 : Representasi sesuai gambar (1) : (a) Matriks ketetanggaan, (b) Matriks beririsan
Sumber : Bondy (2008)

Representasi yang dibuat digunakan untuk melakukan manipulasi pada graf. Salah satu pemanfaatan yang paling umum digunakan adalah *traversal* atau pengaksesan pada graf dengan pola tertentu, seperti mengakses sesuai keterhubungan antar *vertex*, sesuai nilai *edge* yang terbesar apabila graf tergolong graf berbobot, dan lain-lain. Bentuk *traversal* yang paling umum adalah BFS (Breadth-First-Search) dan DFS (Depth-First-Search).

Algoritma BFS bergerak dengan pola *first-in, first-out* sehingga memproses seluruh *vertex* terdekat yang terhubung dengan *vertex* yang sedang menjadi fokus proses. Sedangkan algoritma DFS mengnakan *backtrack*, sehingga memroses tiap satu *vertex* yang terhubung dengan *vertex* yang menjadi fokus proses, mengubah fokus proses ke *vertex* yang terhubung tersebut, dan terus mengakses *vertex* lainnya yang terhubung hingga mencapai titik sentinel seperti *vertex* yang terakhir diakses hanya memiliki satu keterhubungan atau *vertex* tersebut terhubung dengan *vertex* – *vertex* yang telah diproses. Ketika kondisi ini terpenuhi, terjadilah *backtrack* hingga ditemukan *vertex* terhubung yang belum diproses.

B. Objek 3D

Objek 3D secara literal memliki makna yaitu sebuah objek yang memiliki panjang, lebar, dan tinggi/ketebalan. Pada koteks informatika, objek 3D terbentuk atas kumpulan *vertex*, dan *edge* yang membentuk *face*. *Face* adalah suatu polygon yang terdiri atas *vertex* dan *edge* yang menghubungkan *vertex* dengan pola *cycle*. Satuan penyusun minimal suatu objek 3D adalah *face* yang dapat berbentuk triangular, quad, atau polygon bersisi banyak. Objek 3D digunakan pada banyak bidang, mulai dari hiburan seperti film animasi, permainan, hingga bidang professional seperti desain bangunan, produk, dan lain-lain. Contoh objek 3D dapat dilihat pada gambar



Gambar 5 : Contoh Objek 3D yang membentuk kerangka kepala kucing
Sumber : Gotler (2011)

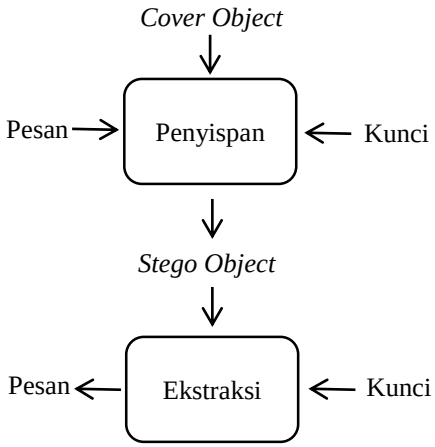
Konsep penyusun objek 3D memilik prinsip bisa dibilang bagian dari teori graf. Perbedaannyaterletak pada penyusun terkecil dan syarat – syarat yang harus dipenuhi suatu graf untuk dapat dikategorikan sebagai bantuk yang dapat dijadikan sebagai objek 3D.

Salah satu format *file* objek 3D yang popular adalah format *wavefront* dengan ekstensi *.obj. Format ini menggunakan

C. Steganografi

Steganografi adalah bentuk komunikasi rahasia sehingga hanya pengirim dan penerima pesan saja yang tahu akan adanya komunikasi tersebut (Cox, 2008). Contoh

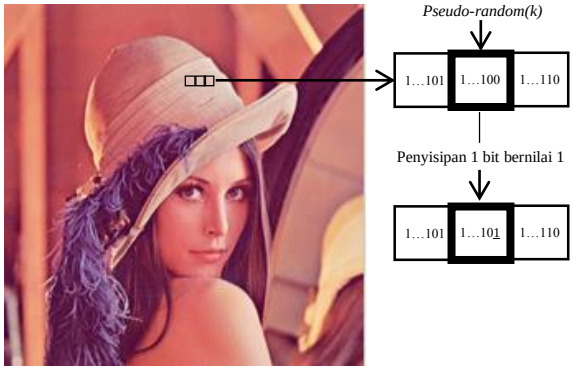
kasus yang menjadi kasus sederhana pemanfaatan steganografi adalah kasus pada sistem pengiriman pesan antar tahanan penjara. Pada kasus ini, pengiriman pesan dari tahanan A ke B harus melalui sipir penjara C. C akan membaca pesan terlebih dahulu, menentukan pesan tersebut berisi pesan yang diizinkan atau tidak, lalu meneruskan atau tidak pesan tersebut sesuai penentuan. Tahanan penjara A ingin mengirimkan pesan terkait usaha untuk kabur dari penjara kepada tahanan B. Agar pesan yang disampaikan tidak diketahui oleh sipir C, A melakukan steganografi pesan rahasia dengan cara menjadikan pesan rahasia sebagai akronim, dengan diawali persetujuan tahanan B. Dengan begitu, sipir penjara yang tidak mengetahui akronim pesan sebagai pesan utama hanya dapat membaca pesan seperti biasa dan akronim menjadi pesan rahasia yang hanya diketahui tahanan A dan tahanan B.



Gambar 6 : Diagram Steganografi Umum

Media yang menjadi teempat penyisipan disebut *cover object*, lalu setelah disisipkan pesan media tersebut menjadi *stego object* yang kemudian menjadi inputan pada proses ekstraksi pesan. Prosesnya adapat diperhatikan pada gambar 6.

Pada konteks informatika, steganografi secara umum dilakukan menggunakan gambar. Pada Ilustrasi steganografi pada objek gambar dapat diperhatikan pada gambar 7. Pada gambar 7 ini, dilakukan pemilihan posisi pixel gambar secara *pseudo-random* menggunakan kunci sebagai pembangkit. Pada pixel yang terpilih dilakukan sebuah penyisipan pesan. Pesan yang ingin disisipkan secara umum adalah teks, namun pada penerapannya pesan bisa berupa data dengan format apapun. Hal ini disebabkan oleh pesan yang akan disisipkan terlebih dahulu diubah sehingga direpresentasikan dengan bit. Bit ini lah yang satu demi satu akan disisipkan ke dalam pixel gambar. Metode yang populer atau umum digunakan adalah manipulasi *least significant bit* (LSB), yaitu mengubah nilai bit dengan pengaruh perubahan pada gambar terkecil, yaitu bit pada satuan terkecil. Pada contoh di gambar 7, penyisipan pesan dilakukan per 1 bit pesan di tiap LSB pada suatu pixel terpilih. Apabila LSB pada pixel terpilih memiliki nilai yang sama dnegan pesan yang ingin disisipkan, tidak dilakukan perubahan, sedangkan jika berbeda dilakukan perubahan seperti yang terdapat pada gambar 7 tersebut. Setelah semua pesan tersisipkan, selesailah proses penyisipan dan dihasilkanlah sebuah *stego image*.



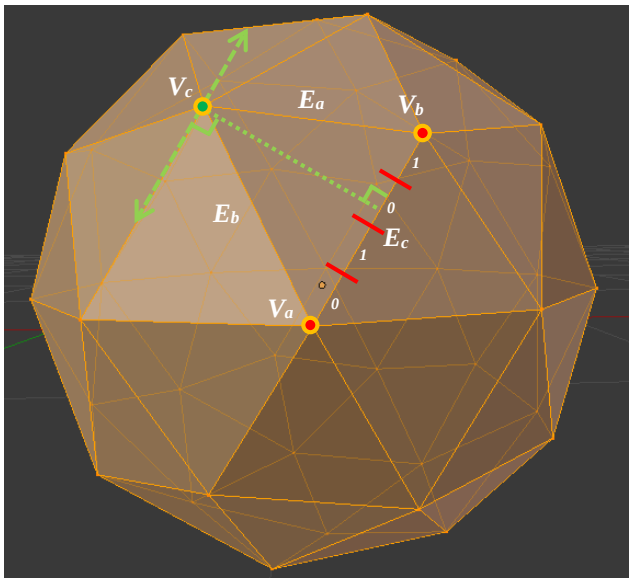
Gambar 7 : Contoh penyisipan 1 bit pesan pada objek gambar

Untuk proses ekstraksi pesan, dilakukan dengan cara mengakses pixe dengan posisi *pseudo-random* sesuai nilai yang dibangkitkan menggunakan kunci. Untuk tiap pixel yang diakses dilakukan pembacaan nilai LSB. Bit – bit yang dibaca digabungkan satu sama lain sehingga membentuk data pesan layaknya sebelum pengiriman.

Steganografi menggunakan objek gambar, audio, ataupun video dapat dilakukan menggunakan manipulasi LSB. Ketiga objek tersebut menggunakan pola dan pengaruh bit yang serupa terhadap kondisi masing – masingnya. Apabila LSB pada posisi tertentu di dalam badan dari data diubah, efek yang timbl adalah perubahan output seperti perubahan warna ataupun suara yang dihasilkan. Akan tetapi untuk data pada objek 3D, konsep dan pola data yang serupa tidak ada. Perubahan bit dapat mengakibatkan tidak terbaca nya data tersebut. Sehingga dilakukan perubahan

Metode dasar dalam steganografi pada objek 3D berasalah dari algoritma yang dibuat oleh Cayre (2003). Secara mendasar, melalui metode ini konsep penyisipan dan pembacaan pesan pada saat ekstraksi dilakukan dengan cara mengubah posisi *vertex* dengan perbandingan terhadap sebuah *edge*. Pesan yang disisipkan tetap diubah menjadi bit – bit terlebih dahulu karena penyisipan dilakukan per bit. Tiap *Edge* yang diproses dan menjadi objek pambanding dipartisi dengan metode yang seragam menjadi zona – zona nilai 1 dan 0. Melalui partisi ini, ketika proyeksi *vertex* terhadap *edge* berada pada suatu rentang jarak partisi, maka dapat diartikan bahwa nilai partisi tersebut adalah nilai bit pesan rahasia. Apabila posisi proyeksi *vertex* terhadap *edge* tidak berada pada partisi dengan nilai yang tepat, posisi *vertex* diubah agar masuk ke zona partisi dengan nilai yang sesuai bit pesan dan dengan jarak yang terdekat. Agar memperoleh jarak terdekat, selain dilakukan perhitungan jarak, perubahan

posisi *vertex* harus sejajar terhadap *edge* pembanding. Dengan cara ini, tiap *vertex* pada objek 3D dapat menyimpan 1 bit besa rahasia. Contoh penerapan ini dapat diperhatikan seperti yang terdapat pada gambar 8. Di gambar 8 ini, dilakukan perubahan posisi *vertex* V_c terhadap *edge* E_c yang telah dipartisi menjadi 4 zona. Apabila nilai bit yang d V_c pada zona E_c tidak sesuai bit pesan yang ingin disisipkan, posisi V_c harus diubah sejajar E_c sehingga zona proyeksinya telah sesuai dengan bit pesan.



Gambar 8 : Steganografi pada Objek 3D secara umum

III. PERANCANGAN SOLUSI

A. Diffusion dan Kunci

Menerapkan aspek *diffusion* dapat dilakukan dengan menggunakan kunci rahasia. Pada steganografi dengan objek gambar, *diffusion* terdapat pada pemilihan pixel secara *pseudo-random* menggunakan kunci sebagai pembangkitnya. Pada steganografi dengan objek 3D, pembangkitan bilangan *pseudo-random* dapat digunakan dalam pemilihan *vertex* yang akan disisipkan pesan rahasia.

```
begin
   make_matriks_vertex(vertex)
   make_matriks_edge(edge)
   bit_pesan <- convert_to_bit(pesan)
   i <- 0
   while (i < length(bit_pesan))
         r <- random(kunci)
         z <- zona_vertex(vertex[r],edge[r]
         b <- is_pernah_ubah(vertex[r])
         if ((z <> bit_pesan[i]) & (b = true)
                  ubah_posisi(vertex[acak], edge[r])
end
```

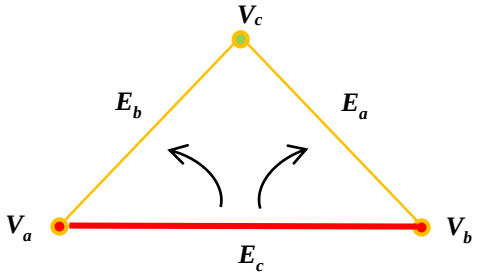
Gambar 9 : Pseudocode steganografi pada objek 3D sederhana

Seperti yang terdapat pada gambar 9, memilih *vertex* secara *pseudo-random* dapat dilakukan dengan tetap memperhatikan dan menyimpan log *vertex* yang telah disipkan dan *edge* yang menjadi pembandingnya. Dengan begitu ketika suatu *vertex* telah disisipkan pesan dan terjadi pengaksesan ulang akibat bilangan acak yang dibangkitkan pernah terbangkitkan, *vertex* tersebut tidak diubah kembali posisinya, begitu pula dengan *edge* yang menjadi pembandingnya. Melalui cara ini, pesan rahasia tidak dapat disisipkan

Pemilihan *vertex* secara *pseudo-random* memiliki kelemahan, yaitu sangat tidak efisiennya pemanfaatan *vertex* dalam penyisipan pesan rahasia. Apabila yang diharapkan adalah 1 bit pesan rahasia pada 1 *vertex*, dengan cara ini penyisipan 1 bit pesan rahasia untuk kasus terburuk memerlukan 3 *vertex*. 3 *vertex* ini adalah satu *vertex* sebagai *vertex* yang disisipkan pesan lalu 2 *vertex* sebagai penyusun *edge* pembandingnya. Kasus terbaiknya terjadi apabila dua *vertex* penyusun *edge* pembanding yang terpilih tersusun dari *vertex* yang telah disisipkan pesan. Dari kasus ini, perbaikan yang diharapkan adalah selalu terjadinya kasus terbaik ini.

B. Pemilihan dan Pengaksesan Vertex

Agar *edge* pembanding yang terpilih dalam tiap proses penyisipan adalah dua *vertex* yang telah disisipkan pesan, pembangkitan bilangan acak tidak dapat digunakan. Solusi untuk mencapai hal ini dapat diperoleh dengan hanya memilih *edge* yang terdapat pada *face* yang telah diubah posisi salah satu *vertex* nya. Seperti yang terdapat pada gambar 10, pada *face* tersebut V_c disisipkan pesan rahasia dengan E_c sebagai *edge* yang menjadi pembandingnya. Langkah selanjutnya adalah memilih *face* yang bertetanggaan, yaitu *face* lain yang memiliki E_b atau E_c sebagai penyusunnya. Karena V_a , V_b , dan V_c yang telah direkap sebagai *vertex* yang telah diakses, *vertex* ini tidak boleh lagi diubah. *Face* tetangga yang dipilih akan diproses dengan *edge* terhubung (E_a atau E_b) sebagai *edge* pembanding dengan *vertex* dihadapannya untuk proses lanjutan. Metode ini digunakan sebagai dasar pengembangan dalam banyak algoritma steganografi 3D, seperti algoritma yang dibuat oleh Cheng dan Wang (2006), Cheng dan Wang (2007), Chao (2009), dan banyak lagi,



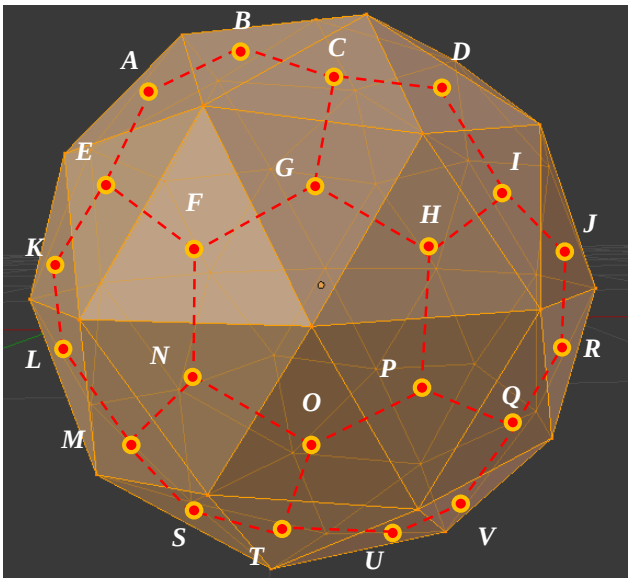
Gambar 10 : Dua pilihan edge untuk proses selanjutnya

Hal vital yang diperlukan untuk melakukan hal ini adalah pembuatan model representasi yang baik sehingga pengaksesan atau *traversal* antar *face* jelas. Representasi ini dapat dibentuk menggunakan teori graf. Bentuk representasi yang dapat digunakan adalah matriks atau senarai ketetanggaan, lalu matriks bersisian.

C. Pembentukan Representasi Graf

Representasi graf yang dibentuk sebagai solusi bukanlah representasi graf yang secara langsung menerjemahkan *vertex* dan *edge* penyusun objek 3D. Seperti yang tertera pada gambar 11, graf yang dibentuk adalah representasi dari graf yang mendeskripsikan ketetanggaan antar *face*. *Face* dianggap sebagai seuatu *vertex* dan hubungan ketetanggaan antar *face* sebagai *edge* nya.

Bentuk matriks ketetanggaan yang dapat dibentuk untuk sebagian keterhubungan *face* objek 3D pada gambar 11 dapat dilihat pada table 1.



Gambar 11 : Visualisasi graf dari face yang bertetangga

Tabel 1 : Matriks Ketetanggan Graf pada gambar 11 untuk edge A hingga J

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
A	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
E	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
F	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
G	0	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0
I	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
J	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0

Dengan menggunakan representasi ini penentuan *face* selanjutnya akan lebih mudah. Bentuk representasi ini dapat dipersederhana sesuai dengan jumlah sisi terbesar dari suatu *face* penyusun graf karena jumlah *face* tetangga sama dengan jumlah sisi suatu *face*.

Diffusion dapat dilakukan dengan menentukan *face* pertama. Selain itu juga menentukan *edge* yang akan dipilih selanjutnya. Kemungkinan *face* yang dipilih setelah sastu proses penyisipan hanya ada dua. Oleh karena itu kunci dapat digunakan dengan cara membangkitkan bilangan random dengan kunci sebagai parameter untuk menentukan sisi kanan atau kiri yang akan dipilih.

Melalui gambar 12, dapat dilihat algoritma steganografi dengan memilih *face* tetangga untuk proses lanjutan. Algoritma ini dapat dituliskan dengan cara rekursif dengan basis adalah tidak ada lagi pesan yang dapat disisipkan. Ketika tidak ada lagi *face* yang dapat diakses, akan diakukan *backtrack* hingga mencapai *face* dengan tetangga yang belum diakses.

```
begin
        bit_pesan <- convert_to_bit(pesan)
        i <- 0
        r <- random(kunci)
        f <- face[r]
        stegano3D(f, bit_pesan, i)
end

procedure stegano3D(input f : face; input bp : bit;
                    input/output i : integer)
        begin
                v <- vertex[face[f]]
                e <- edge[face[f]
                z <- zona_vertex(v, e]
                if (z <> bit_pesan[i])
                     ubah_posisi(v, e)
                r <- random(kunci)
                n <- r mod jumah_sisi_face(f)
                j <- 0
                do
                     p <- face_tetangga(face[f], r)
                     j <- j + 1
                while (is_pernah_ubah(vertex[p])
                           & (j < n))
                if ((j <> n) & (i < length(bp))
                     f <- p
                     stegano3D(f, bp, i)
        end
```

Gambar 12 : Pseudocode steganografi dengan pemilihan face tetangga

IV. KESIMPULAN

Pada steganografi dengan objek 3D, pembentuk *diffusion* menggunakan kunci dapat diterapkan dengan cara pembangkitan bilangan *pseudo-random* yang bertujuan menentukan 2 kemungkinan output. Output ini menentukan pola *face* tetangga yang diakses pada proses penyisipan bit selanjutnya. Penentuan *face* yang dipilih juga mengandung *backtrack* apabila *face* yang terakhir diakses buntu. Ini menambah aspek *diffusion* pada steganografi dengan objek 3D.

Pemilihan *vertex* dengan pola tetangga dan *pseudo-random* dapat dilakukan dengan menerapkan teori graf dalam merepresentasikan *face* pada objek 3D sebagai *vertex* dalam graf dan relasi keterhubungan antar *face* pada objek 3D sebagai *edge* pada graf. Dengan representasi ini, pengaksesan atau *traversal* yang dilakukan pada objek 3D dapat dengan lebih cepat dan lebih mudah dikontrol pergerakannya. Oleh karena itu steganografi pada objek 3D dapat lebih optimal terutama dari sisi kapasitas bit yang dapat disisipkan dan *diffusion* dimilikinya.

REFERENSI

[1] Cox, I.J., Miller, M.L., Bloom, J.A., Fridrich J, Kalker T. (2008). *Digital watermarking and steganography, 2nd ed.* Mogran Kaufmann, Burlington

[2] Levitin, Anany. (2011). *Introduction to the Design and Analysis of Algorithms 3rded.* Addison Wesley, New Jearsey.

[3] Bondy, Adrian., Murty, U.S.R. (2008). *Graph Theory (Graduate Texts in Mathematics)*. Springer-Verlag, New York.

[4] Corman, Thomas H., Leiserson, Charles E., Rivest, Ronald L., Stein, Clifford. (2009). *Introduction to Algorithms 3rd ed*. MIT Pres, Massachusetts.

[5] Cayre, F., Macq, *B.* (2003). *Data hiding on 3-D triangle meshes*. IEEE Trans, Signal Process 51(4) : 939-949

[6] Cheng, Y.M., Wang, *C*.M. (2006). *A high-capacity steganographic spproach for 3D polygonal meshes.* Vis Comput 22:845 -855

[7] Cheng, Y.M., Wang, *C*.M. (2007). *An adaptive steganographic algorithm for 3d polygonal mehses.* Vis Comput 23:721-732

[8] Chao, M.W., Lin, *C*.H., Yu, *C*.W., Lee, T.Y. (2009). *A high capacity 3D steganography Algorithm.* IEEE Transaction on Visualization & Computer Graphics Vol 20 No. 3

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, 9 Desember 2014