Penerapan *Scene Graph* dalam Pemodelan Tiga Dimensi

Prisyafandiafif Charifa (13509081)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
prisyafandiafif@students.itb.ac.id

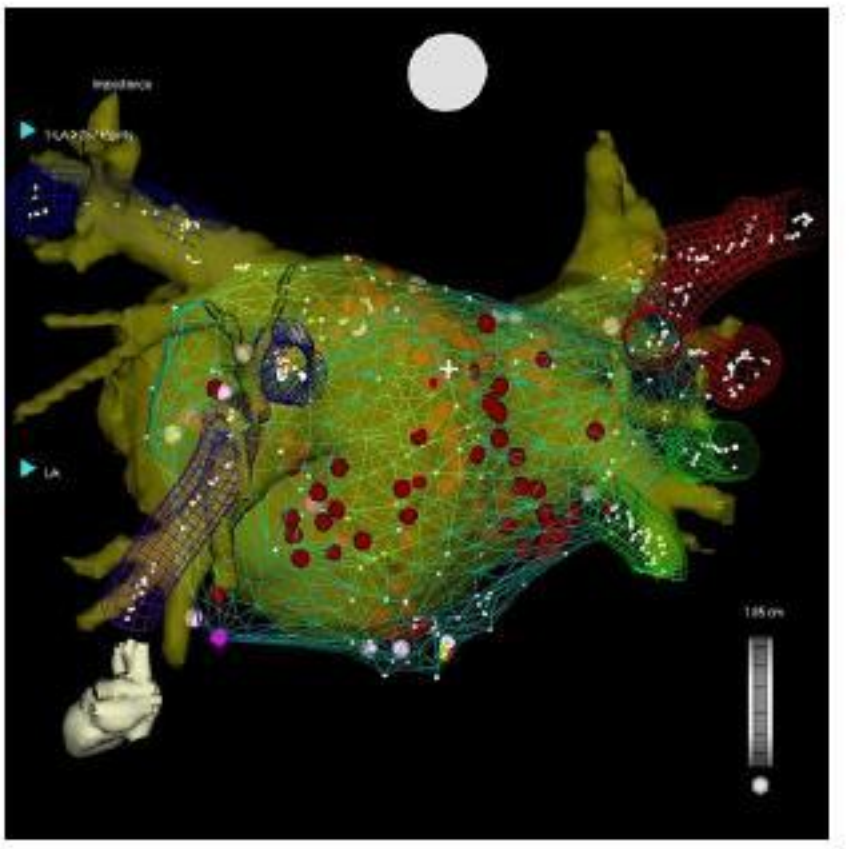
ABSTRAK

Teknologi komputer saat ini sedang berkembang dengan sangat pesat. Komputer yang dahulu fungsinya hanya untuk perhitungan aritmatika, kini sudah mampu melakukan fungsi-fungsi lain yang kompleks dan sulit dilakukan oleh manusia. Salah satu fungsi tersebut adalah kemampuan untuk melakukan pemodelan tiga dimensi. Kemampuan ini membawa perubahan besar pada berbagai sektor kehidupan manusia. Hal ini terbukti dengan dipecahkannya berbagai masalah di banyak bidang keprofesian akibat dari adanya kemampuan ini, seperti pada bidang kedokteran, perminyakan, atau arsitektur. Seiring dengan berkembangnya zaman, maka semakin bertambah pula kecepatan suatu komputer untuk memproses data. Akibatnya, pemodelan tiga dimensi juga menjadi semakin rumit karena adanya tuntutan untuk membuat model tiga dimensi yang pada awalnya sederhana dan tidak nyata menjadi semakin rinci dan mendekati kenyataan. Dapat dibayangkan betapa banyaknya data yang harus diproses komputer jika ingin dihasilkan suatu model tiga dimensi yang seperti itu. Oleh karenanya, dibutuhkan suatu teknik pengelolaan data untuk menangani hal tersebut. Salah satu teknik pengelolaan data ini adalah dengan Scene Graph. Teknik ini memungkinkan komputer untuk menghasilkan suatu model tiga dimensi yang rinci dan mendekati kenyataan dengan pengelolaan data yang sederhana.

Kata Kunci—scene graph, **graf, pohon, struktur data.**

I. PENDAHULUAN

Kemampuan komputer untuk melakukan pemodelan tiga dimensi dari suatu objek telah menjadi suatu terobosan baru di dalam kehidupan manusia. Banyak sekali bidang keprofesian yang mengalami kemajuan karena adanya kemampuan ini. Sebagai contoh adalah bidang kedokteran. Beberapa masalah pengobatan dalam bidang kedokteran telah berhasil dipecahkan dengan pemodelan tiga dimensi. Salah satunya adalah masalah dalam pengobatan penyakit aritme jantung. Penyakit aritme jantung ini adalah suatu kelainan jantung di mana jantung berdetak secara tidak teratur. Fakta mengungkap bahwa dari 1.000 orang terdapat 5 penderita aritme jantung. Penyakit ini dahulu tidak bisa disembuhkan secara total karena tidak adanya alat yang mampu mendeteksi bagian jantung mana yang menyebabkan jantung berdetak tidak beraturan. Namun dengan suatu teknik pemodelan tiga dimensi yang disebut *mapping* (pemetaan) rongga jantung, maka dapat dihasilkan suatu model tiga dimensi dari struktur jantung tersebut secara *real-time* seperti pada Gambar 1 di bawah ini.



Gambar 1. Mapping 3D Rongga Jantung

Pada model tiga dimensi yang dihasilkan dari teknik pemetaan ini, juga disertakan secara sistematis konduksi listrik jantung yang pada gambar di atas ditunjukkan dengan gambar bulatan kecil. Konduksi listrik jantung inilah yang akan digunakan untuk mendeteksi bagian mana pada jantung yang menyebabkan jantung berdetak tidak teratur. Dengan begitu, masalah pengobatan penyakit aritme jantung ini pun terpecahkan. Masih banyak bidang lain yang menggunakan pemodelan tiga dimensi untuk memecahkan berbagai masalah yang ada, seperti bidang arsitektur atau perminyakan.

Kemampuan komputer dalam memproses data yang terus bertambah seiring dengan berkembangnya zaman dan tuntutan-tuntutan dari berbagai bidang keprofesian yang menginginkan suatu visualisasi tiga dimensi yang lebih rinci akhirnya memaksa para *programmer* untuk terus berusaha agar dapat memaksimalkan kemampuan komputer yang ada. Hasil dari usaha tersebut adalah

dapat divisualisasikannya suatu model tiga dimensi yang sangat rinci bahkan menyerupai model sebenarnya. Model jenis ini sering digunakan dalam bidang arsitektur terutama untuk memvisualisasikan suatu bangunan secara tiga dimensi. Model jenis ini juga sering divisualisasikan dalam skala besar, terutama pada pembuatan *next-gen video game* dan beberapa film besar. Gambar 2 di bawah ini adalah contoh dari penerapan model dengan tingkat kerincian tinggi dalam skala besar.

Gambar 2. *High-Detailed Model in Large Scale*

Model-model bertingkat kerincian tinggi dalam jumlah banyak seperti gambar di atas tentu mempunyai jumlah data yang sangat banyak pula. Data inilah yang kemudian akan diproses oleh komputer sedemikian rupa sehingga dapat dibentuk menjadi suatu model tiga dimensi yang rinci. Namun, jika berhadapan dengan data yang begitu banyak jumlahnya, tentu masalah kemangkusan dalam memproses data tersebut menjadi suatu masalah yang tidak bisa disepelekan. Untuk itu diperlukan suatu struktur data atau pengelolaan data yang benar agar pemrosesan data tersebut dapat berjalan semangkus mungkin. Dalam pemodelan tiga dimensi ini, dipakailah suatu struktur data yang bernama *Scene Graph*. Pada dasarnya, *Scene Graph* ini adalah sebuah metode untuk menggambarkan struktur data dengan menggunakan suatu jenis pohon bernama m-ary. Berbagai hal yang bersangkutan dengan *Scene Graph* akan dibahas pada bagian berikutnya.

II. METODE

Metode penulisan makalah yang digunakan adalah metode studi pustaka dan analisis kasus. Setelah penulis menguraikan penggunaan *Scene Graph* pada pemodelan tiga dimensi, penulis mencoba untuk membentuk suatu algoritma struktur data untuk mengimplementasikan suatu *Scene Graph* yang diberikan sebagai contoh.

III. DASAR TEORI

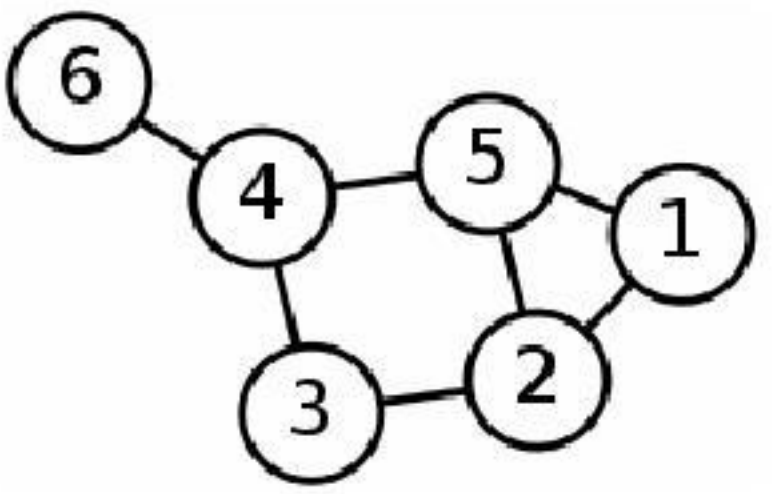
Ada beberapa hal yang perlu diketahui sebelum membahas *Scene Graph* lebih lanjut. Hal tersebut adalah beberapa teori tentang pengertian dari graf dan pohon, dan juga *Scene Graph* itu sendiri.

III.1 Pengertian Graf

Secara ilmu matematika, pengertian dari suatu graf dengan nama G adalah sebagai berikut :

"Suatu pasangan himpunan (V,E) yang ditulis dengan notasi G = (V,E).Ddalam hal ini V adalah himpunan tidak kosong dari simpul-simpul (vertices atau node) dan E adalah himpunan dari sisi-sisi (edges atau arcs) yang menghubungkan sepasang simpul.dan boleh kosong."

Dari pengertian di atas, dapat dinyatakan bahwa V tidak boleh kosong, sedangakan E boleh kosong. Jadi, suatu graf dimungkinkan tidak mempunyai sisi satu buah pun tetapi simpulnya tetap harus ada, minimal satu. Graf yang hanya mempunyai satu simpul tanpa satu buah sisi pun disebut graf trivial. Simpul pada graf dapat ditandai dengan huruf, seperti $a,b,c,...,$ dengan bilangan asli, seperti 1,2,3,..., atau gabungan keduanya. Gambar 3 di bawah ini adalah contoh dari suatu graf secara ilmu matematika.



Gambar 3. Graf dalam Ilmu Matematika

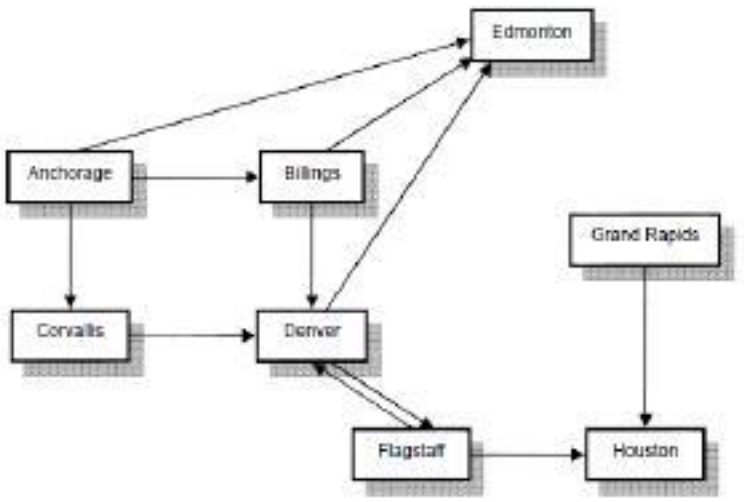
Selain pengertian graf secara ilmu matematika, juga terdapat pengertian graf secara ilmu komputer. Pengertian graf secara ilmu komputer ini merupakan implementasi lanjut dari graf secara ilmu matematika.Di sini, graf digunakan untuk menggambarkan cara pengelolaan data atau biasa disebut struktur data. Graf ini biasa disebut sebagai graf struktur data. Dalam graf struktur data, suatu sisi dapat diberi nilai tertentu, begitu juga dengan simpul. Nilai tersebut dapat berupa sebuah simbol ataupun data-data numerik lainnya seperti panjang, kapasitas, dll. Selain itu simpul tidak harus berbentuk bundar ataupun titik, melainkan dapat berbentuk kotak, elips, dsb.

Berdasarkan orientasi arah pada sisi, graf juga dapat dibedakan menjadi dua jenis.

1. **Graf tak berarah**
   Graf ini sisinya tidak mempunyai orientasi arah.

2. **Graf berarah**
   Graf ini sisinya mempunyai orientasi arah.

Pada Gambar 4 dibawah ini ditunjukkan suatu graf struktur data yang berarah.



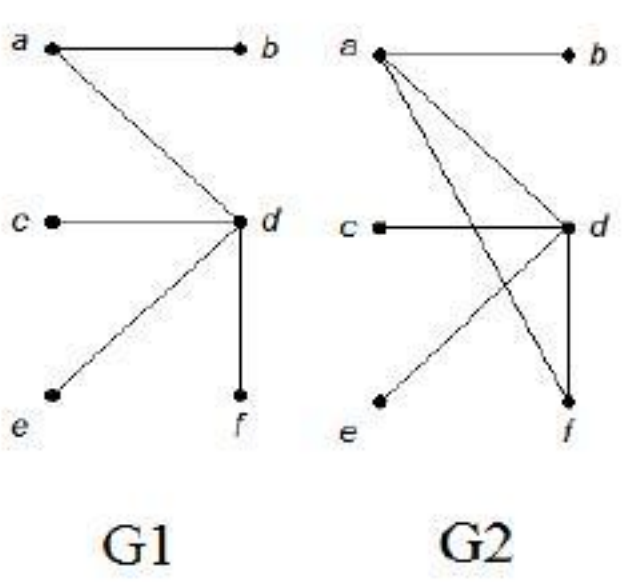
Gambar 4. Graf Struktur Data Berarah

III.2 Pengertian Pohon

Pengertian dari suatu pohon adalah sebagai berikut :

"Suatu graf tak berarah terhubung yang tidak mengandung sirkuit"

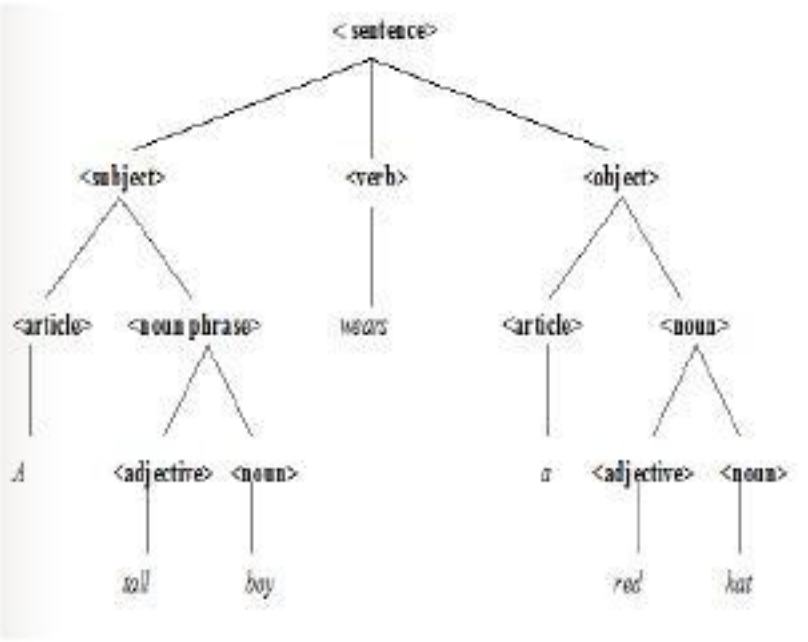
Dari pengertian di atas, dapat diketahui dua sifat penting dari pohon, yaitu terhubung dan tidak mengandung sirkuit. Maksud dari terhubung adalah jika untuk setiap pasang simpul pada satu graf, terdapat lintasan bolak-balik yang menghubungkan kedua simpul tersebut. Sedangkan maksud dari tidak mengandung sirkuit adalah jika pada satu graf terdapat suatu lintasan yang berawal dan berakhir pada simpul yang sama. Pada Gambar 5 dibawah ini ditunjukkan perbedaan antara pohon dan bukan pohon.



Gambar 5. G1 adalah pohon dan G2 bukan pohon

Sama seperti graf, pohon juga mempunyai beberapa jenis. Di antara berbagai jenis tersebut, yang perlu dibahas adalah pohon berakar dan pohon *m-ary*. Pohon berakar adalah pohon yang mana sebuah simpulnya diperlakukan sebagai akar dan sisi-sisinya diberi arah menjauh dari akar. Ada beberapa terminologi yang perlu diketahui pada pohon berakar, yaitu terminologi anak dan orangtua. Suatu simpul *a* dikatakan sebagai anak dari simpul *b* jika ada sisi yang berarah dari simpul *b* ke simpul *a*.

Sedangkan pohon *m-ary* adalah pohon berakar yang setiap simpul cabangnya mempunyai paling banyak *m* buah anak. Gambar 6 di bawah ini adalah salah satu conto dari pohon *m-ary* dengan m=3.

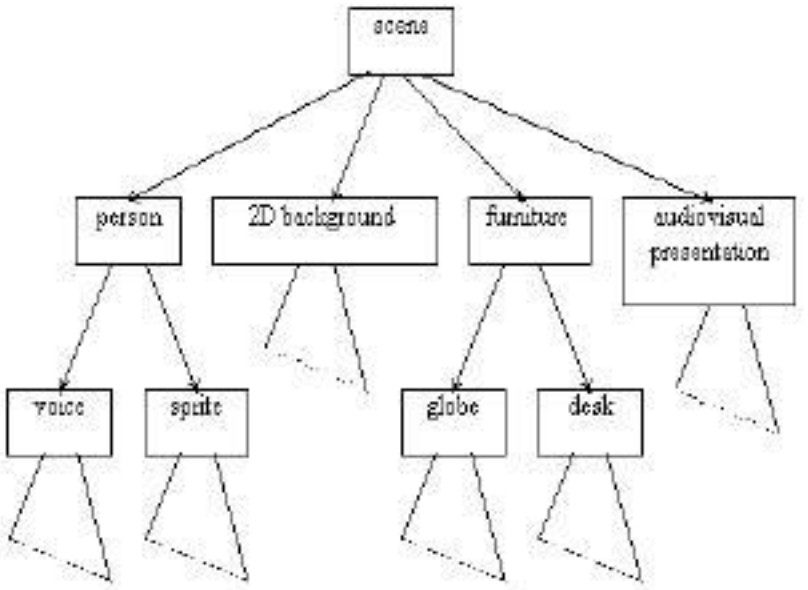


Gambar 6. Pohon 3-*ary*

Pada suatu pohon berakar, tanda arah yang biasanya disertakan pada sisi-sisi pohon boleh tidak dicantumkan seperti pada Gambar 6 di atas. Dengan begitu, ada suatu persetujuan bahwa arah pada pohon tersebut berlangsung dari simpul yang letaknya lebih atas ke simpul yang letaknya lebih bawah.

III.3 Pengertian Scene Graph

Secara umum, *Scene Graph* adalah suatu teknik pengelolaan data atau struktur data yang biasanya dipakai pada suatu aplikasi rekayasa grafis berbasis vektor dan pemodelan tiga dimensi baik itu untuk keperluan *video game* ataupun yang lainnya dengan menggunakan konsep graf berarah dan pohon m-*ary*. Beberapa program yang menggunakan *Scene Graph* ini adalah AutoCad, Adobe Illustrator, Corel Draw, dll. Pengertian teoritis dari *Scene Graph* ini sebenarnya tidak jelas atau kabur dikarenakan para *programmer* yang menggunakan *Scene Graph* pada sebuah aplikasi hanya mengambil prinsip dasarnya lalu mengimplementasikannya pada aplikasi tersebut sesuai fungsi dari aplikasi tersebut. Namun yang pasti adalah bahwa *Scene Graph* mengatur logika dari suatu adegan bergambar. Pada setiap simpul di dalam satu *Scene Graph*, terdapat sebuah aksi yang harus dilakukan sebelum alur berlanjut pada simpul anaknya. Hal inilah yang membedakan *Scene Graph* dengan graf biasa. Pada Gambar di bawah ini ditunjukkan contoh dari suatu *Scene Graph* dari pemrosesan format video MPEG-4.



Gambar 7. *Scene Graph* dalam MPEG-4

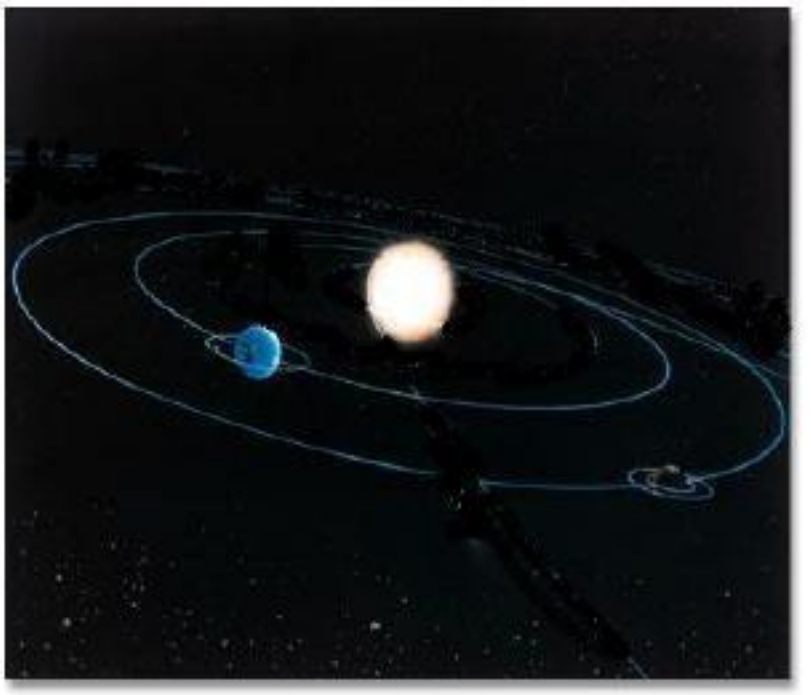
IV. PEMBAHASAN

Setelah membahas tentang pengertian dari graf, pohon, dan *Scene Graph* itu sendiri, berikut ini akan dibahas tentang penerapam *Scene Graph* pada pemodelan tiga dimensi dengan lebih lanjut. *Scene Graph* sebenarnya tidak hanya digunakan untuk pemodelan tiga dimensi saja, tapi juga untuk pemodelan dua dimensi. Namun di sini penulis membatasi bagian pembahasan ini agar yang dibahas hanya penggunaan *Scene Graph* pada pemodelan tiga dimensi saja.

IV.1 Scene Graph pada Pemodelan Tiga Dimensi

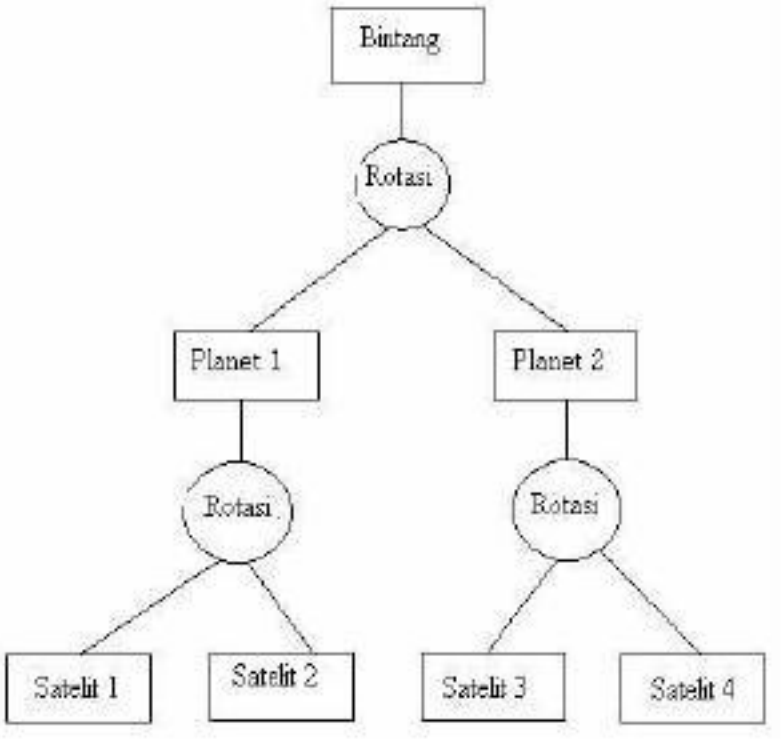
Pemodelan tiga dimensi memang sudah banyak diterapkan pada berbagai bidang di kehidupan manusia. Untuk melakukan pemodelan tiga dimensi itu diperlukan sebuah aplikasi yang menunjang pembuatan model tiga dimensi. Sudah banyak aplikasi penunjang pembuatan model tiga dimensi yang beredar di pasaran, sebut saja Autodesk 3DS Max, Autodesk Maya, AutoCad, dll. Cara kerja aplikasi ini pada dasarnya adalah mengolah berbagai data yang ada seperti koordinat posisi, vektor, gambar, suara, dll. sehingga dapat ditampilkan menjadi suatu model tiga dimensi yang interaktif dan *real-time*. Model tiga dimensi tersebut diproses sedemikian rupa sehingga dapat bergerak dan bertingkah layaknya benda sebenarnya. Hal inilah yang nantinya disebut sebagai animasi tiga dimensi.

Scene Graph diterapkan dalam aplikasi pengolah model tiga dimensi sebagai suatu teknik pengelolaan data sehingga struktur data yang akan dibuat menjadi mangkus dan mudah dimengerti. Agar penjelasan mengenai *Scene Graph* pada pemodelan tiga dimensi menjadi lebih jelas, maka penulis akan mengambil suatu contoh model yang akan divisualisasikan secara tiga dimensi kemudian menampilkan contoh *Scene Graph* -nya. Pada penjelasan ini penulis akan menggunakan sistem tata surya sebagai contoh. Pada sistem tata surya yang akan divisualisasikan, terdapat sebuah bintang sebagai pusatnya dan dua planet yang mengitari bintang tersebut. Setiap planet mempunyai dua satelit yang mengitari planet tersebut. Gambar 8 di bawah ini adalah contoh sistem tata surya yang akan dibuat.



Gambar 8. Sistem tata surya sederhana

Untuk memvisualisasikan model ini, sebenarnya ada dua cara. Cara pertama adalah dengan membuat fungsi-fungsi dengan algoritma yang rumit pada setiap objek yang ada pada sistem tata surya tersebut. Hanya saja, jika suatu saat seorang *designer* yang merancang model ini ingin mengubah posisi suatu planet, maka diperlukan juga suatu perubahan pada algortima fungsi pengatur posisi satelit yang mengitari planet itu. Selain itu jika objek yang ada bertambah banyak, maka pembuatan fungsi juga akan menjadi banyak sehingga pemodelan menjadi tidak mangkus. Karena itu dilakukanlah cara kedua. Cara kedua adalah dengan membuat suatu *Scene Graph* yang akan menyederhanakan segalanya. Gambar 9 di bawah ini adalah diagram *Scene Graph* yang akan merepresentasikan keadaan di atas.



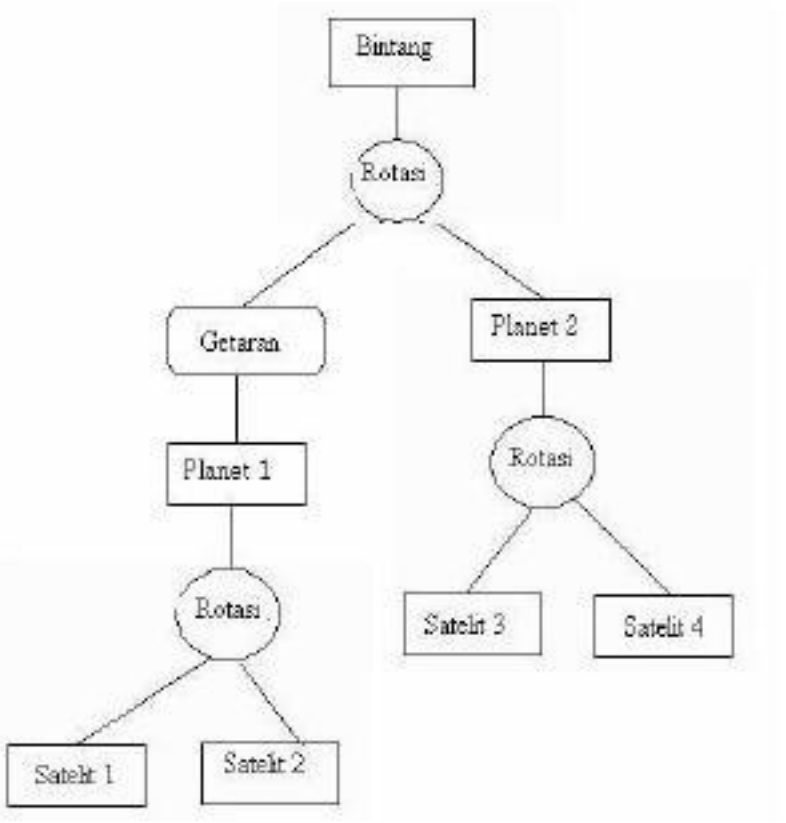
Gambar 9. *Scene Graph* 1

Dari *Scene Graph* 1 diasumsikan bahwa simpul bernama "Rotasi" adalah simpul yang melakukan penyimpanan koordinat suatu objek ke dalam matriks dan mengalikannya sedemikian rupa sehingga objek tersebut berputar mengitari bintang yang di sini disebut rotasi. Setiap simpul melakukan suatu aksi dahulu sebelum akhirnya berlanjut ke simpul anaknya. Dengan begitu, dapat dilihat bahwa setelah simpul "Bintang" yang melakukan penggambaran atau pemunculan bintang di layar, dilakukanlah proses rotasi pada planet 1 dan 2 dengan kecepatan yang sama baru kemudian menggambarkannya ke layar. Setelah itu dilakukan lagi proses rotasi pada satelit 1,2,3, dan 4 dengan kecepatan yang berbeda, dan digambarkan lagi ke layar. *Scene Graph* 1 di atas bila ditulis secara logika dengan kata-kata adalah sebagai berikut :

- Gambarkan objek bintang ke layar
- Simpan koordinat ke dalam matriks
    - Lakukan proses rotasi planet 1 terhadap bintang
    - Gambarkan objek planet 1 ke layar
    - Simpan koordinat ke dalam matriks
        - Lakukan proses rotasi satelit 1 dan 2 terhadap planet
        - Gambarkan objek satelit 1 ke layar
        - Gambarkan objek satelit 2 ke layar
    - Hapus matriks yang telah disimpan
    - Gambarkan objek planet 2 ke layar
    - Simpan koordinat ke dalam matriks
        - Lakukan proses rotasi satelit 3 dan 4 terhadap planet
        - Gambarkan objek satelit 3 ke layar
        - Gambarkan objek satelit 4 ke layar
    - Hapus matriks yang telah disimpan

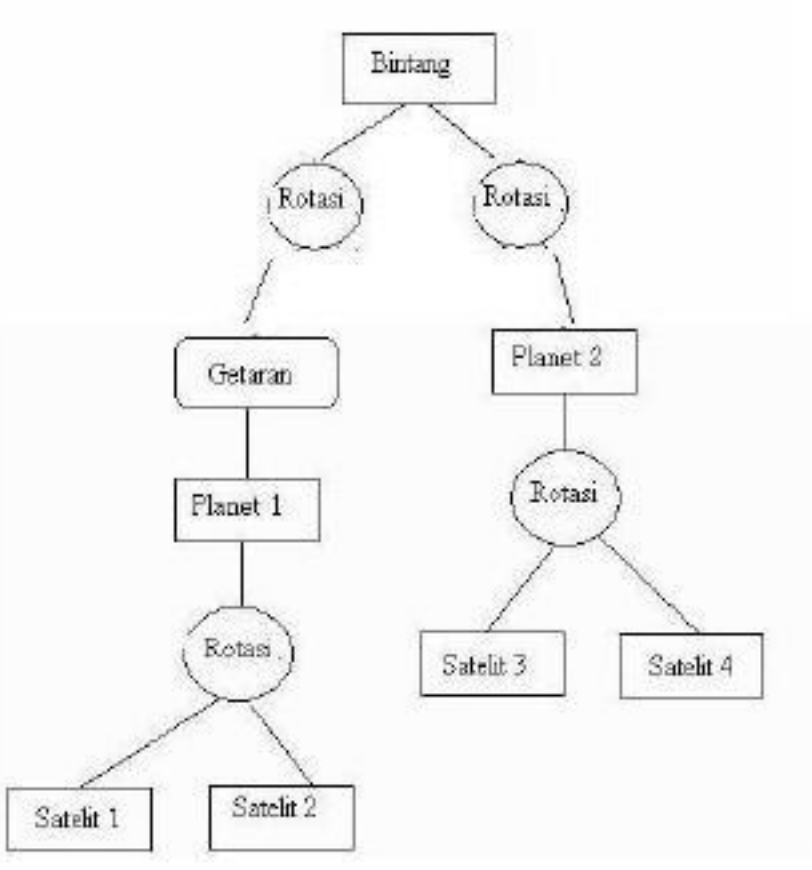
Scene Graph yang telah dibuat seperti di atas masih bersifat sederhana. Kelebihan dari *Scene Graph* ini adalah tidak diperlukannya sintaks-sintaks rumit untuk memvisualisasikan adegan-adegan yang diinginkan seperti yang biasa digunakan dalam membuat suatu aplikasi besar. Di dalam *Scene Graph* ini, dapat disertakan banyak aksi seperti rotasi dan getaran tapi tidak dibahas bagaimana cara melakukan rotasi dan getaran tersebut. Untuk itu, bagian berikutnya akan membahas suatu adegan visualisasi yang lebih rumit lagi.

Pada keadaan ini kurang lebih sama dengan keadaan sebelumnya, hanya saja planet 1 akan sedikit bergetar. Asumsikan planet 1 bergetar karena adanya hantaman dari asteroid. Yang perlu dilakukan untuk memvisualisasikan keadaan ini adalah menambah simpul "Getaran" pada *Scene Graph* yang telah dibuat sebelumnya tepat sebelum simpul "Planet 1". Asumsikan simpul "Getaran" melakukan suatu proses kepada objek tertentu sehingga objek tersebut terlihat bergetar. Gambar 10 di bawah ini merepresentasikan keadaan tersebut.



Gambar 10. *Scene Graph* 2

Adegan di atas masih dapat dibuat menjadi lebih kompleks lagi. Berikutnya diinginkan agar planet 1 dan planet 2 berputar mengitari bintang dengan kecepatan yang berbeda. Gambar 11 di bawah ini merepresentasikan keadaan tersebut.



Gambar 11. *Scene Graph* 3

Pembacaan alur dari *Scene Graph* 3di atas sama dengan pembacaan alur pada *Scene Graph* 1, hanya saja terdapat dua simpul rotasi setelah simpul bintang. Hal ini

menandakan bahwa nantinya kecepatan dan arah rotasi dari planet 1 dan 2 dapat diatur sesuai keinginan. *Scene Graph* 3 di atas bila ditulis secara logika dengan kata-kata adalah sebagai berikut :

- Gambarkan objek bintang ke layar
- Simpan koordinat ke dalam matriks
  - Lakukan proses rotasi planet 1 pada bintang
    - Simpan koordinat ke dalam matriks
      - Lakukan proses getaran pada planet 1
      - Gambarkan objek planet 1 ke layar
        - Simpan koordinat ke dalam matriks
          - Lakukan proses rotasi satelit 1 dan 2 terhadap planet
            - Gambarkan objek satelit 1 ke layar
            - Gambarkan objek satelit 2 ke layar
          - Hapus matriks yang telah disimpan
        - Hapus matriks yang telah disimpan
      - Hapus matriks yang telah disimpan
    - Hapus matriks yang telah disimpan
  - Simpan koordinat ke dalam matriks
    - Lakukan proses rotasi planet 2 pada bintang
      - Gambarkan objek planet 2 ke layar
      - Simpan koordinat ke dalam matriks
        - Lakukan proses rotasi satelit 3 dan 4 terhadap planet
        - Gambarkan objek satelit 3 ke layar
        - Gambarkan objek satelit 4 ke layar
      - Hapus matriks yang telah disimpan
    - Hapus matriks yang telah disimpan
  - Hapus matriks yang telah disimpan

Dari *Scene Graph* 1,2, dan 3 di atas, dapat dilihat bahwa banyak variasi dari suatu keadaan dalam pemodelan tiga dimensi yang dapat diolah dan diatur dengan metode ini. Masih banyak implementasi *Scene Graph* yang lebih rumit daripada yang telah dijelaskan pada makalah ini. Contohnya apabila ingin memvisualisasikan sebuah kota dengan penduduk yang padat. Jika ingin memvisualisasikan keadaan tersebut, dibutuhkan banyak simpul-simpul yang mengatur segala kegiatan dari objek yang ada seperti manusia, tumbuhan, atau bangunan. Pada bagian berikutnya akan dibahas mengenai implementasi struktur data yang telah dibuat dengan *Scene Graph* menjadi struktur data dalam bahasa pemrograman tertentu.

IV.2 Analisis dengan Pemrograman

Setelah membahas tentang penerapan struktur data visualisasi tiga dimensi sistem tata surya dengan *Scene* Graph, maka berikutnya akan dibahas mengenai penerapan struktur data tersebut pada bahasa pemrograman sebenarnya. Di sini, penulis tidak menggunakan notasi algoritma atau *pseudo-code* dikarenakan algoritma yang akan digunakan berorientasi objek. Karena itu, penulis berasumsi bahwa bahasa pemrograman berorientasi objek yang umum digunakan dan mudah dimengerti adalah bahasa pemrograman C++. Gambar 12 berikut ini adalah penerapan struktur data visualisasi tiga dimensi sistem tata surya dengan C++.

```
class CSceneNode
{
public:
  // Konstruktor
  CSceneNode() { }

  // Destruktor
  virtual ~CSceneNode() { Destroy(); }

  // Menghapus objek dari memori
  void Release() { delete this; }

  // Mengganti simpul-simpul yang ada
  virtual void Update()
  {
    // Melakukan looping sesuai dengan daftar yang ada dan mengganti simpul anak
    for( std::list<CSceneNode*>::iterator i = m_lstChildren.begin();
         i != m_lstChildren.end(); i++ )
    {
      (*i)->Update();
    }
  }

  // Menghapus semua simpul anak
  void Destroy()
  {
    for( std::list<CSceneNode*>::iterator i = m_lstChildren.begin();
         i != m_lstChildren.end(); i++ )
      (*i)->Release();

    m_lstChildren.clear();
  }

  // Menambah simpul anak pada suatu simpul lainnya
  void AddChild( CSceneNode* pNode )
  {
    m_lstChildren.push_back(pNode);
  }

protected:
  // Daftar dari simpul anak yang ada
  std::list<CSceneNode*> m_lstChildren;
};
```

Gambar 12. Struktur data sistem tata surya dalam C++

Selain struktur data di atas, diperlukan struktur data lain yang merupakan implementasi dari setiap simpul yang ada di dalam *Scene Graph*. Struktur data tersebut adalah untuk menggambarkan suatu objek ke layar, menyimpan koordinat ke dalam matriks, dan melakukan beberapa pergerakan. Gambar 13 berikut akan menampilkan struktur data untuk menggambarkan suatu objek ke layar.

```
class CGeometryNode: public CSceneNode
{
public:
  CGeometryNode() { }
  ~CGeometryNode() { }

  void Update()
  {
  // Gambarkan objek yang ingin ditampilkan ke layar di sini.

  CSceneNode::Update();
  }
};
```

Gambar 13. Struktur data untuk menggambarkan objek tiga dimensi ke layar dalam C++

Setelah struktur data untuk menggambarkan objek tiga dimensi, berikut ini juga ditampilkan pada Gambar 14 yaitu struktur data untuk menyimpan koordinat objek ke matriks dalam C++.

```
class CDOFNode: public CSceneNode
{
public:
  CDOFNode() { }
  ~CDOFNode() { }

  void Initialize( float m[4][4] )
  {
    for( int i = 0; i < 4; i++ )
      for( int j = 0; j < 4; j++ )
        m_fvMatrix[i][j] = m[i][j];
  }

  void Update()
  {
    glPushMatrix();
    glLoadMatrix( (float*)m_fvMatrix );

    CSceneNode::Update();

    glPopMatrix();
  }

private:
  float m_fvMatrix[4][4];
};
```

Gambar 14. Struktur data untuk menyimpan koordinat objek ke matriks dalam C++

Struktur data yang terakhir adalah struktur data utnuk melakukan pergerakan pada suatu objek tiga dimensi. Untuk struktur data ini, penulis tidak menampilkan implementasinya dalam bahasa pemrograman C++ dikarenakan panjang dan banyaknyanya kode tersebut. Selain itu juga terdapat banyak rumus matematika yang akan membuanya makin rumit jika ditampilkan dalam makalah ini. Namun intisari dalam struktur data untuk menggerakkan objek tiga dimensi adalah mencatat waktu pergerakan dan perpindahan posisi dari objek tersebut.

V. KESIMPULAN

Dari berbagai pembahasan di atas mengenai penerapan *Scene Graph* pada pemodelan tiga dimensi, dapat diambil beberapa kesimpulan sebagai berikut :

a. Pemodelan tiga dimensi telah menjadi suatu terobosan baru dalam berbagai bidang kehidupan manusia, terbukti dengan ditemukannya teknik pengobatan baru untuk penyakit aritme jantung melalui Mapping 3D Rongga Jantung.
b. Telah ditemukan suatu metode pengelolaan data yang memungkinkan para *programmer* untuk membuat suatu struktur data yang mangkus untuk pemodelan tiga dimensi. Metode ini disebut *Scene Graph*.
c. *Scene Graph* adalah suatu metode pengelolaan data yang sering dipakai pada suatu aplikasi rekayasa grafis berbasis vektor dan pemodelan tiga dimensi dengan menggunakan konsep graf berarah dan pohon m-*ary*
d. Kelebihan dari metode *Scene Graph* dalam menentukan struktur data adalah keterhubungan antara suatu keadaan dengan suatu objek sehingga untuk mengubah keadaan suatu objek seperti adanya pergerakan menjadi lebih mudah,
e. *Scene Graph* hanya merupakan suatu diagram yang memberikan alur keadaan suatu objek sehingga menjadi suatu adegan. Untuk penerapan sesungguhnya, maka *Scene Graph* tersebut harus diterjemahkan ke dalam bahasa pemrograman berorientasi objek.

VI. DAFTAR PUSTAKA

[1] Munir, Rinaldi, "Matematika Diskrit", 2nd ed. Bandung: Informatika Bandung, 2009, hal. 353–486.
[2] Rosen, Kenneth H, *"Discrete Mathematics and Its Applications"*, 6th ed. New York: McGraw-Hill, 2007, hal. 589–743.
[3] http://en.wikipedia.org/wiki/Scene_graph.
Tanggal Akses : 13 Desember 2010
[4] http://en.wikipedia.org/wiki/Graph_theory.
Tanggal Akses : 13 Desember 2010
[5] http://en.wikipedia.org/wiki/Tree_(data_structure).
Tanggal Akses : 12 Desember 2010
[6] http://retno-ayu-sp.blogspot.com/2010/05/membuat-sebuah-actor-menggunakan-delta.html.
Tanggal Akses : 15 Desember 2010
[7] http://developer.techsoft3d.com/documentation/GettingStarted/DefiningTheScenegraph_C.html.
Tanggal Akses : 12 Desember 2010
[8] http://hellforge.gameriot.com/blogs/Hellforge/CryEngine-2-Scenes-From-A-Movie
Tanggal Akses : 12 Desember 2010
[9] http://www.lhsc.on.ca/About_Us/LHSC/Media_Room/Media_Releases/2006/september5.htm
Tanggal Akses : 12 Desember 2010
[10] http://www.montana.edu/wwwmor/education/NASAtrunks/planets.html
Tanggal Akses : 13 Desember 2010
[11] http://en.wikipedia.org/wiki/Animation
Tanggal Akses : 13 Desember 2010
[12] http://en.wikipedia.org/wiki/K-ary_tree
Tanggal Akses : 12 Desember 2010
[13] http://ride.chiariglione.org/inside_MPEG-4_part_A/inside_MPEG-4_part_A.htm
Tanggal Akses : 13 Desember 2010
[14] http://id.wikipedia.org/wiki/Sejarah_komputer|
Tanggal Akses : 13 Desember 2010
[15] http://hamilton.bell.ac.uk/swdev2/notes/notes_18.pdf
Tanggal Akses : 13 Desember 2010

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, 17 Desember 2010

Prisyafandiafif Charifa
(13509081)