PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK UNTUK PENGENALAN OBJEK 3 DIMENSI BERBASIS VEKTOR

TUGAS AKHIR

Oleh :

MAS AGUNG
NRP. 2694 100 016

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2000

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK UNTUK PENGENALAN OBJEK 3 DIMENSI BERBASIS VEKTOR

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer
Pada
Jurusan Teknik Informatika
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya

Mengetahui / Menyetujui,

Dosen Pembimbing I	Dosen Pembimbing II
Ir. ESTHER HANAYA, M.Sc.	RULLY SOELAIMAN, S.Kom
NIP. 130 816 212	NIP. 132 085 802

SURABAYA
Februari, 2000

ABSTRAK

Dalam tugas akhir ini dilakukan pengenalan terhadap objek-objek 3 dimensi dari representasi format file (CAD) khusus. Para pengusaha pabrik dapat menghemat waktu dan biaya dengan kemampuan untuk menggunakan sebuah rancangan prototip untuk mencari produk-produk yang sudah ada untuk perancangan serupa. Pemakaian ulang baik rancangan dan persediaan menjadi lebih praktis dengan teknologi ini, dan pengenalan objek 3D dapat juga dilakukan untuk mengidentifikasi objek-objek yang tidak dikenal. Kinerja dari indeks-indeks berbasis taksonomi dapat ditingkatkan. Masih banyak lagi kegunaan lain dari alat pencarian ini.

Algoritma pengenalan dimulai dengan mengubah file-file CAD menjadi representasi pemukaan (surface), dan mengubahnya lebih lanjut menjadi representasi voxel. Representasi voxel tersebut kemudian dipakai untuk mengkalkulasi ciri-ciri objek, seperti momen-momen geometrik 3D orde nol, pertama, dan kedua.

Kinerja program yang dihasilkan berdasarkan analisis menunjukkan bahwa teknologi ini akurat dalam mendiskriminasi objek-objek berdasarkan geometri.

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa. Seperti yang tertulis dalam kitab Mazmur 127:1 "*Jikalau bukan Tuhan yang membangun rumah, sia-sialah usaha orang yang membangunnya; jikalau bukan Tuhan yang mengawal kota, sia-sialah pengawal berjaga-jaga.*", hanya atas berkat, rahmat, kuasa, dan izin Tuhan-lah tugas akhir ini dapat diselesaikan tepat pada waktunya.

Sehubungan dengan disusunnya tugas akhir ini, penulis mengucapkan terima kasih kepada:

1. Ibu **Esther Hanaya, Ir., MSc.,** selaku dosen pembimbing I tugas akhir saya.
2. Bapak **Rully Soelaiman, SKom.** selaku dosen pembimbing II tugas akhir saya.
3. Bapak **Arif Djunaidi, Ir., MSc., PhD.,** selaku Ketua Jurusan Teknik Informatika ITS Surabaya.
4. Bapak **Khakim Gozali, Ir.,** selaku dosen wali saya.
5. Seluruh Bapak dan Ibu dosen yang telah membimbing saya selama masa perkuliahan.
6. **Ayahanda** dan **Ibunda** tersayang serta seluruh **keluarga** tercinta yang selalu memberikan dorongan moril maupun materiil untuk secepatnya menyelesaikan tugas akhir ini.
7. **Mudi** the Dogol dan **Roni** the Dogoller.
8. Kawan-kawan **C0A**.
9. **Inprise Corp.,** untuk kompiler Borland Delphi 5 yang sangat membantu pembuatan tugas akhir ini.

10. **SquareSoft,** untuk game Final Fantasy VIII yang menemani penulis dalam pembuatan tugas akhir ini.
11. Semua pihak yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari bahwa tugas akhir ini masih jauh dari sempurna. Untuk itu kritik dan saran dari para pembaca sangat penulis harapkan

Semoga tugas akhir ini bermanfaat bagi setiap pembaca.

Surabaya, Februari 2000

Penulis

DAFTAR ISI

halaman

DAFTAR GAMBAR

halaman

BAB I

PENDAHULUAN

1.1 LATAR BELAKANG

Kita semua hidup di alam tiga dimensi. Benda-benda di sekeliling kita memiliki panjang, lebar, dan tinggi. Untuk dapat mengenali benda-benda tersebut, kita sebagai manusia dikaruniai Tuhan panca indera sehingga kita relatif jarang salah dalam menginterpretasikan objek tiga dimensi yang berada di hadapan kita.

Di sisi lain, komputer sebagai benda mati yang belum memiliki kecerdasan tidak dapat serta merta mengenali objek-objek 3 dimensi seperti manusia. Diperlukan adanya suatu program yang menyatakan kriteria-kriteria tertentu sehingga komputer dapat memutuskan "kemiripan" objek-objek 3 dimensi yang sedang diperbandingkan.

1.2 PERMASALAHAN

Objek-objek tiga dimensi yang diperbandingkan direpresentasikan dalam poligon-poligon yang hanya berupa permukaan benda saja, sedangkan untuk dapat menghitung momen-momen yang dipakai untuk memperbandingkan objek-objek tersebut menuntut diketahuinya fungsi densitas dari objek-objek tersebut. Transformasi dari representasi permukaan objek yang kontinyu menjadi representasi densitas objek yang bersifat diskrit (voxel) sangatlah mutlak diperlukan.

Selain itu, pada bagian akhir aplikasi akan dibuktikan mengenai keabsahan dari pemakaian kriteria yang disebutkan dalam dasar teori untuk mengenali objek-objek tiga dimensi tersebut.

1.3 TUJUAN DAN MANFAAT

Tujuan tugas akhir ini adalah mengenali objek-objek 3 dimensi yang diinputkan berdasarkan kemiripan ciri-ciri dengan objek-objek 3 dimensi yang telah diolah sebelumnya dan ada di dalam *library* objek.

Manfaat penulisan tugas akhir ini adalah sebagai berikut:

1. Memberikan informasi secara umum tentang struktur file DXF.
2. Menjelaskan dasar-dasar *volume graphics* yang dipakai sebagai dasar dalam membuat tugas akhir ini, khususnya pada bagian vokselisasi.
3. Memberikan informasi tentang penyelesaian polinomial kubik.
4. Menyediakan algoritma untuk mendiagonalisasi matriks simetrik 3X3. Kontribusi terbesar yang dapat diberikan tugas akhir ini dalam hal ini adalah pada bagian algoritma eliminasi Gauss yang 'tidak lazim'.

1.4 BATASAN MASALAH

Perancangan dan pembuatan perangkat lunak pengenalan objek 3 dimensi berbasis vektor ini direalisasikan dengan menggunakan bahasa pemrograman PASCAL dengan Compiler Borland Delphi 4.0.

Batasan-batasan masalah yang ada sebagai berikut:

1. File-file CAD (Computer Aided Design) yang dipergunakan sebagai input adalah file-file DXF spesifik yang dihasilkan oleh editor 3 dimensi 3DSMax.
2. File-file tersebut merepresentasikan objek-objek 3 dimensi dengan metode POLYLINE yang berupa pendefinisian sisi-sisi objek yang masing-masing berupa hubungan tiga titik yang membentuk segitiga.

1.5 METODOLOGI PENULISAN

Kebutuhan utama dalam penulisan tugas akhir ini adalah tersedianya perangkat lunak (software) bantu dan perangkat keras (hardware) yang memadai, serta literatur-literatur pendukung lainnya. Untuk memperoleh data dan informasi mengenai perancangan dan pembuatan perangkat lunak (software) secara lengkap dan detail dalam penyusunan tugas akhir ini, penulis menempuh beberapa tahap sebagai berikut:

1. Pengumpulan informasi.

   Pengumpulan data dan informasi tentang perangkat lunak (software) bantu yang dapat mendukung penyusunan perancangan dan pembuatan perangkat lunak (software) ini.

2. Studi literatur.

   Penelusuran jurnal-jurnal, buku-buku ilmiah, atau tulisan-tulisan lainnya yang relevan dengan perancangan dan pembuatan software yang dibahas dilakukan untuk mendukung dan mempermudah pelaksanaan perancangan dan pembuatan perangkat lunak.

3. Perencanaan sistem.

Berdasarkan informasi yang didapatkan baik melalui studi literatur maupun konsultasi, perancangan sistem dilakukan.

4. Pembuatan perangkat lunak.

   Perangkat lunak (software) dibuat sesuai dengan perencanaan rancangan.

5. Evaluasi dan perbaikan.

   Pada tahap ini dilakukan pengujian keandalan software dan koreksi pada sistem yang telah dirancang.

1.6 SISTEMATIKA PEMBAHASAN

Uraian sistematika pembahasan pada tugas akhir ini bertujuan supaya perancangan dan pembuatan perangkat lunak yang dibahas menjadi mudah dipahami, jelas, dan sistematis untuk tiap-tiap bab atau sub bahasan. Secara kronologis uraian dalam tugas akhir ini disusun sebagai berikut:

BAB I Pendahuluan

Bab pertama ini membahas mengenai: latar belakang penulisan, tujuan penulisan, batasan masalah, metodologi penulisan, dan sistematika pembahasan.

BAB II Dasar Teori

Dalam bab ini akan diuraikan kerangka kerja sistem perangkat lunak yang akan dibuat. Setiap proses yang akan dibuat dalam sistem perangkat lunak akan dijelaskan secara teoretis dan terperinci berdasarkan jurnal-jurnal dan buku-buku teks yang relevan.

BAB III Perancangan dan Pembuatan Perangkat Lunak

Penguraian dan penjelasan mengenai tahap-tahap perancangan sistem, dasar-dasar pemrograman, pembuatan program, pengenalan bentuk gambar format DXF, serta implementasi algoritma-algoritma yang telah diuraikan pada dasar teori akan dilakukan pada bab ini.

BAB IV Uji dan Evaluasi Aplikasi Program

Perancangan dan pembuatan perangkat lunak untuk pengenalan objek 3 dimensi berbasis vektor diikuti dengan pengujian perangkat lunak yang dibuat tersebut. Di dalam bab ini akan dibahas mengenai hasil-hasil yang diperoleh melalui pengujian program terhadap bermacam-macam sampel data.

BAB V Penutup

Bab ini adalah bab terakhir yang berisi kesimpulan dan ulasan mengenai hal-hal yang telah dibahas pada bab-bab sebelumnya, serta saran untuk pengembangan program ini di masa yang akan datang.

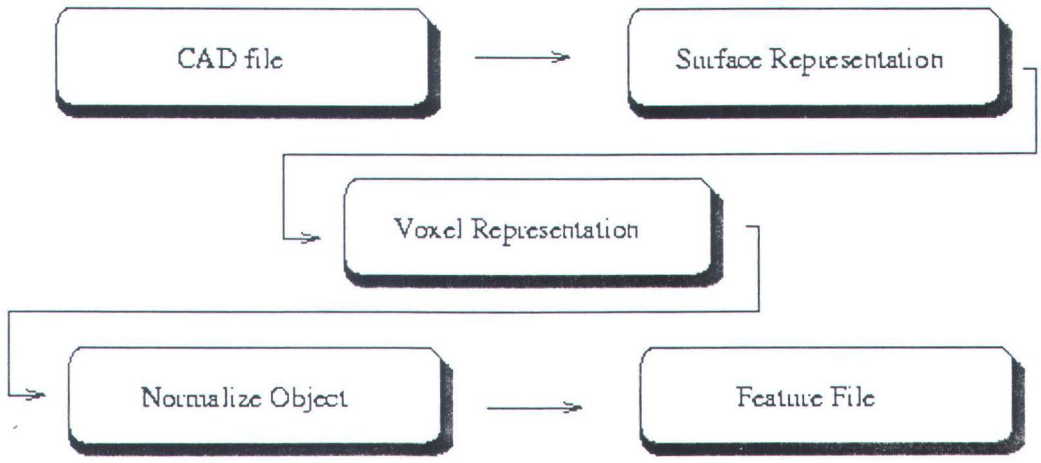
BAB II

DASAR TEORI

Tugas akhir berisi tentang suatu sistem otomatis yang melakukan pengenalan otomatis akan objek 3 dimensi dengan berbasis geometri. Input sistem yang dipakai berupa file CAD. Dimensionalitas dari masalah ini membuatnya lebih kompleks dari masalah pengenalan bentuk 2 dimensi. Masalah pengenalan objek solid untuk masalah pemahaman citra, yang menggunakan pandangan 2D untuk merekonstruksi dan mengenali sebuah objek 3D adalah lebih kompleks dan sulit.

Karena file CAD dipergunakan untuk mendeskripsikan objek 3 dimensi, penampilan objek dapat dianggap mendekati sempurna. Untuk selanjutnya akan difokuskan kepada bagian analisis pola dari masalah pemahaman citra tanpa pertimbangan merekonstruksi objek 3D tersebut dari pandangan 2D.

Langkah – langkah *preprocessing* yang akan dilakukan dapat dijelaskan oleh bagan berikut ini :



Gambar 2.1. Langkah-langkah *preprocessing* suatu file CAD

Akhir preprocessing adalah diperolehnya suatu *feature file* yang berisi *feature* objek. *Feature* inilah yang nantinya akan disimpan sebagai *library* untuk kemudian dibandingkan dengan objek-objek yang akan dikenali. *Feature* yang dihasilkan oleh *preprocessing* tersebut dapat bergantung pada normalisasi, atau *invariant* terhadap ukuran, posisi, dan orientasi objek.

2.1. STRUKTUR FILE DXF

Secara esensial, sebuah file DXF terdiri dari pasangan-pasangan kode dan nilai yang diasosiasikan. Kode-kode tersebut, yang dikenal sebagai kode kelompok, mengindikasikan tipe nilai yang mengikutinya. Dengan memakai kode kelompok dan pasangan-pasangan nilai ini, sebuah file DXF diorganisasikan menjadi bagian-bagian (*sections*), yang terbentuk dari record-record, yang akhirnya terbentuk dari sebuah kode kelompok dan sebuah item data. Setiap kode kelompok dan nilai terletak pada barisnya sendiri dan file DXF tersebut.

Setiap bagian dimulai dengan sebuah kode kelompok 0 yang diikuti dengan string SECTION, yang selanjutnya diikuti oleh kode kelompok 2 dan sebuah string yang mengindikasikan nama bagian tersebut (misalnya, HEADER) . Masing-masing bagian terdiri dari kode kelompok dan nilai-nilai yang mendefinisikan elemen-elemennya. Sebuah bagian berakhir dengan sebuah 0 yang diikuti dengan string ENDSEC.

File DXF diorganisasikan secara keseluruhan sebagai berikut:

- HEADER *Section*. Informasi umum tentang gambar dapat ditemukan dalam bagian ini. Bagian ini terdiri dari nomor versi database AutoCAD dan sejumlah variabel

sistem. Setiap parameter mengandung sebuah nama variabel dan nilai yang diasosiasikan.

- CLASSES *Section*. Bagian ini mengandung informasi untuk kelas-kelas yang didefinisikan oleh aplikasi. Sebuah definisi kelas ditetapkan secara permanen dalam hirarki kelas.
- TABLES *Section*. Bagian ini mengandung definisi-definisi untuk tabel-tabel simbol sebagai berikut:
  - APPID (tabel identifikasi aplikasi)
  - BLOCK_RECORD (tabel referensi blok)
  - DIMSTYLE (tabel gaya dimensi)
  - LAYER (tabel lapisan)
  - LTYPE (tabel tipe baris)
  - STYLE (tabel gaya teks)
  - UCS (tabel sistem koordinat pemakai)
  - VIEW (tabel tampilan)
  - VPORT (tabel konfigurasi sudut pandang)
- BLOCKS *section*. Mengandung definisi blok dan entitas-entitas gambar yang membuat setiap referensi blok dalam gambar.
- ENTITIES *section*. Bagian ini mengandung objek grafis (entitas-entitas) dalam gambar, termasuk referensi blok.
- OBJECTS *section*. Mengandung objek-objek non-grafis dalam gambar. Seluruh objek yang bukan entitas atau *records* tabel simbol atau tabel simbol disimpan dalam bagian ini.

Kode-kode grup dan nilai-nilai yang diasosiasikan mendefinisikan suatu aspek dari sebuah objek atau entitas. Baris yang langsung mengikuti kode grup adalah nilai yang diasosiasikan. Nilai ini dapat berupa suatu *string*, bilangan bulat, atau sebuah nilai titik-mengambang, seperti koordinat X dari sebuah titik. Baris-baris yang mengikuti baris kedua grup tersebut, jika ada, ditentukan oleh definisi grup dan data yang diasosiasikan dengan grup tersebut. Kode-kode grup khusus dipakai seperti penanda untuk awal dan akhir bagian-bagian (*sections*), tabel-tabel, dan akhir file tersebut.

Entitas, objek, kelas, tabel dan entri tabel, dan pemisah file diperkenalkan dengan sebuah kode grup 0 (nol) yang diikuti oleh sebuah nama yang mendeskripsikan grup tersebut.

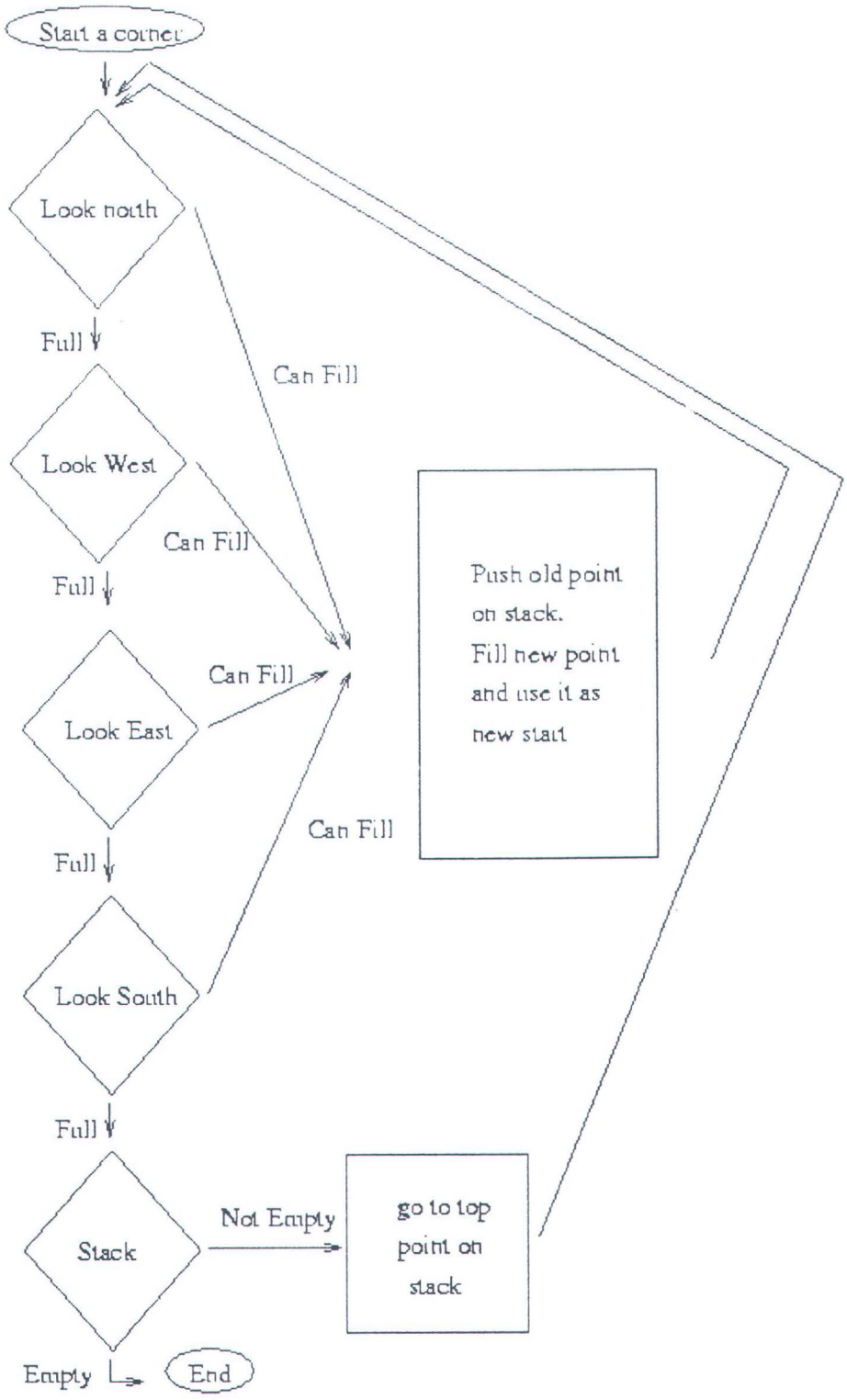
Panjang maksimum *string* file DXF adalah 256 karakter. Jika gambar AutoCAD tersebut mengandung *string* yang melebihi bilangan tersebut, *string* tersebut dipotong selama DXFOUT. DXFIN gagal jika DXF file tersebut mengandung *string* yang melebihi bilangan ini.

2.2. REPRESENTASI VOXEL

Yang dimaksud dengan voxel di sepanjang tugas akhir ini adalah satuan volume (diskrit) terkecil yang dipakai untuk merepresentasikan benda-benda tiga dimensi. Untuk memperjelas maksudnya, voxel dapat dianalogikan dengan pixel untuk objek-objek dua dimensi, di mana pixel merupakan satuan terkecil representasi diskrit dari objek-objek dua dimensi.

Representasi permukaan dikonversi ke representasi voxel dengan suatu proses yang dinamakan *flood-filling*. File yang mendeskripsikan permukaan objek dipindai

lagi arah untuk diisi, bidang tersebut telah selesai diproses. Setelah ruang di luar objek 3D tersebut diisi seluruhnya, bagian yang terisi dikupas sehingga yang tertinggal hanyalah bagian dalam serta perbatasan objek. Sesudah proses *flood filling* selesai, objek tersebut telah berada pada bentuk representasi voxel. Tahap tahap dalam proses ini ditunjukkan pada Gambar 3.



Gambar 2.2. Flowchart Algoritma Flood Filling

2.3. NORMALISASI

Setelah direpresentasikan sebagai voxel, objek menjadi sangat mudah untuk dimanipulasi. Representasi voxel ini merupakan representasi digital dari fungsi densitas objek. Masing-masing lokasi diskrit dalam ruang dipenuhi dengan sebuah densitas digital (nol atau satu). Objek kemudian dinormalisasikan menjadi representasi standar. Dalam sebuah proses normalisasi, suatu objek baru dibuat dengan mempertahankan semua informasi geometris objek aslinya, namun memenuhi sekumpulan kriteria normalisasi. Hal yang perlu diperhatikan adalah bahwa objek kanonikal yang memenuhi kriteria normalisasi ini tidak harus unik. Kriteria normalisasi yang dipakai adalah momen-momen tiga dimensi fungsi densitas tersebut. Untuk sebuah objek yang didefinisikan dengan fungsi densitas $\rho(x,y,z)$, momen tiga dimensi dengan orde l+m+n didefinisikan sebagai:

$$m_{lmn} = \int_{-\infty}^{\infty}\int\int x^l y^m z^n \rho(x,y,z)dxdydz \qquad l,m,n = 1,2,3,\ldots$$

Kriteria normalisasi tersebut meliputi posisi, ukuran, dan orientasi standar.

2.3.1. Posisi Kanonikal

Posisi ini dinormalisasikan dengan menyesuaikan titik berat objek dengan pusat sistem koordinat. Titik berat didefinisikan sebagai :

$$(\hat{x},\hat{y},\hat{z}) = \left(\frac{m_{100}}{m_{000}}, \frac{m_{010}}{m_{000}}, \frac{m_{001}}{m_{000}}\right)$$

Fungsi densitas yang ternormalisasi posisi adalah:

$$\hat{\rho}(x,y,z) = \rho(x-\hat{x}, y-\hat{y}, z-\hat{z})$$

Momen-momen dari fungsi densitas tersebut adalah momen-momen pusat.

$$\mu_{lmn} = \int_{-\infty}^{\infty}\int\int (x-\hat{x})^l (y-\hat{y})^m (z-\hat{z})^n \rho(x-\hat{x}, y-\hat{y}, z-\hat{z}) dxdydz \qquad l,m,n = 0,1,2,3,\ldots$$

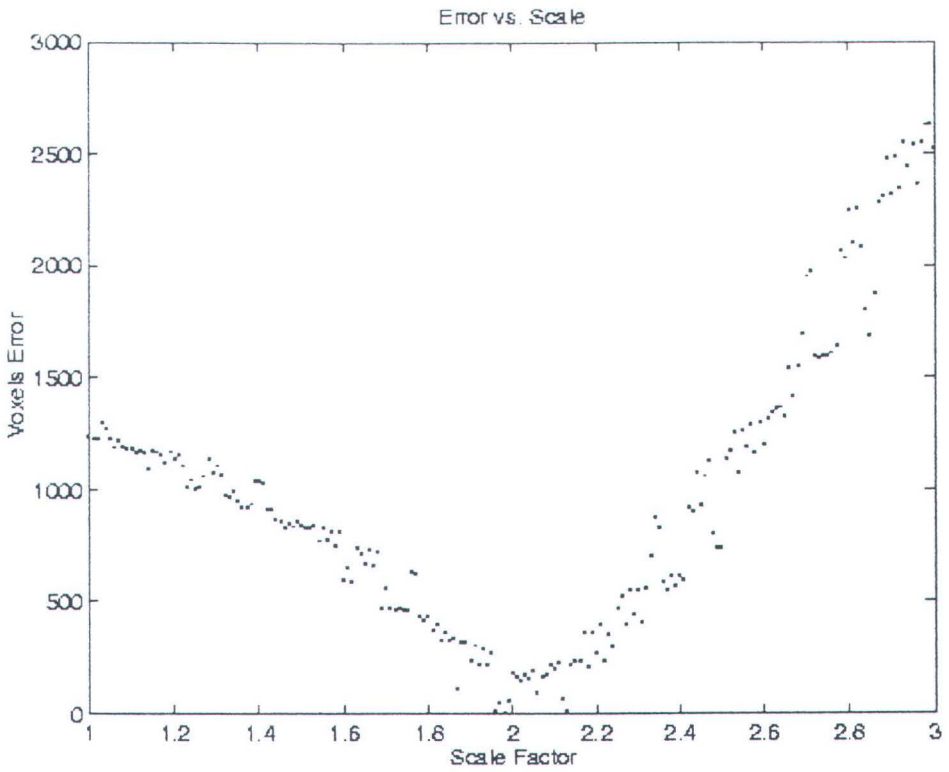
Kondisi tersebut diterapkan pada representasi voxel dengan mengurangi setiap titik dengan $(\hat{x},\hat{y},\hat{z})$. Operasi ini menyeimbangkan objek tersebut sehingga setengah dari densitas objek dapat ditemui dalam setiap setengah ruang koordinat.

2.3.2 Ukuran Kanonikal

Normalisasi ukuran dicapai dengan menyekalakan objek pada volume spesifik, C. Suatu faktor skala α dipergunakan sehingga $m_{000} = \int_{-\infty}^{\infty}\int\int \hat{\rho}(\alpha x, \alpha y, \alpha z) = C$. Kriteria ini diterapkan dengan algoritma iteratif. Pada setiap faktor skala α, objek tersebut memiliki suatu volume V. Tujuan algoritma ini adalah untuk menemukan nilai minimum E, fungsi kesalahan (*error*): $E = |V(\alpha) - C|$. Sebuah algoritma interpolasi parabolik yang dimodifikasi dipakai untuk menemukan nilai minimum tersebut. Karena resolusi objek, yang ditentukan oleh C, tidaklah mungkin untuk mereduksi E menjadi nol; suatu E yang kecil akan ditemukan.

Fungsi kesalahan tersebut mungkin memiliki beberapa titik minimum, dan akan memiliki banyak dataran. Alasan-alasan mengenai hal ini akan dijelaskan lebih lanjut dan ditunjukkan pada Gambar 4, 5, dan 6. Ketika objek diskalakan cukup mendekati ukuran target, algoritma berakhir. Algoritma yang dipakai untuk menyekalakan objek tidak ideal untuk aplikasi ini karena tidak menghasilkan konvergensi menuju nilai minimum global.

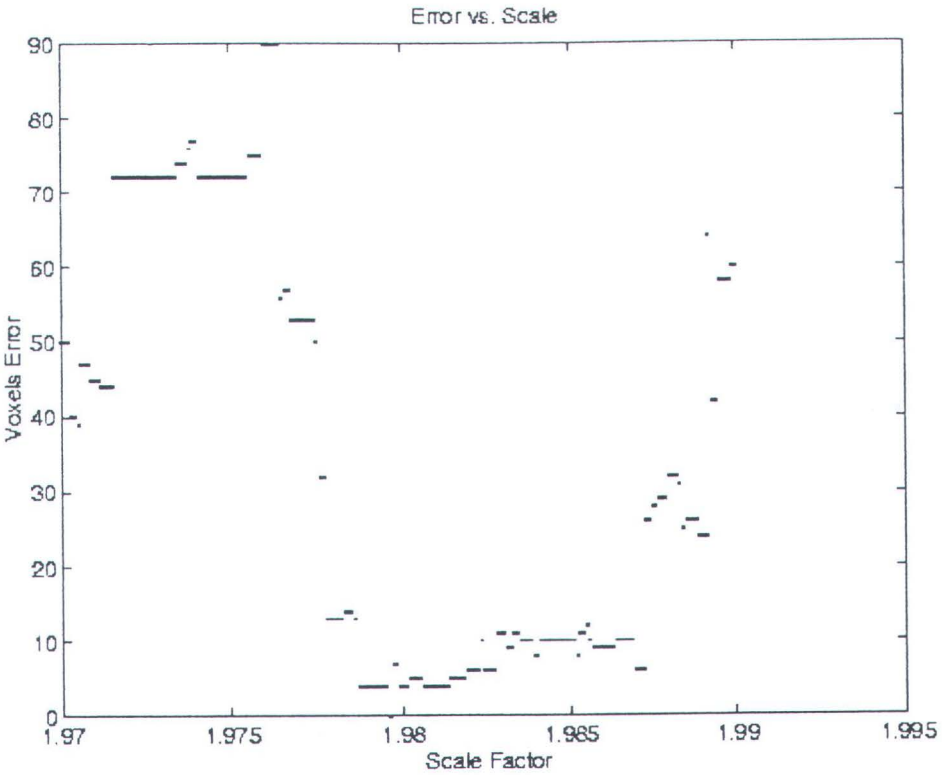
Namun dalam praktiknya, algoritma ini membawa hasil yang berguna, karena suatu titik mula yang cukup beralasan dikalkulasi dalam program tersebut.



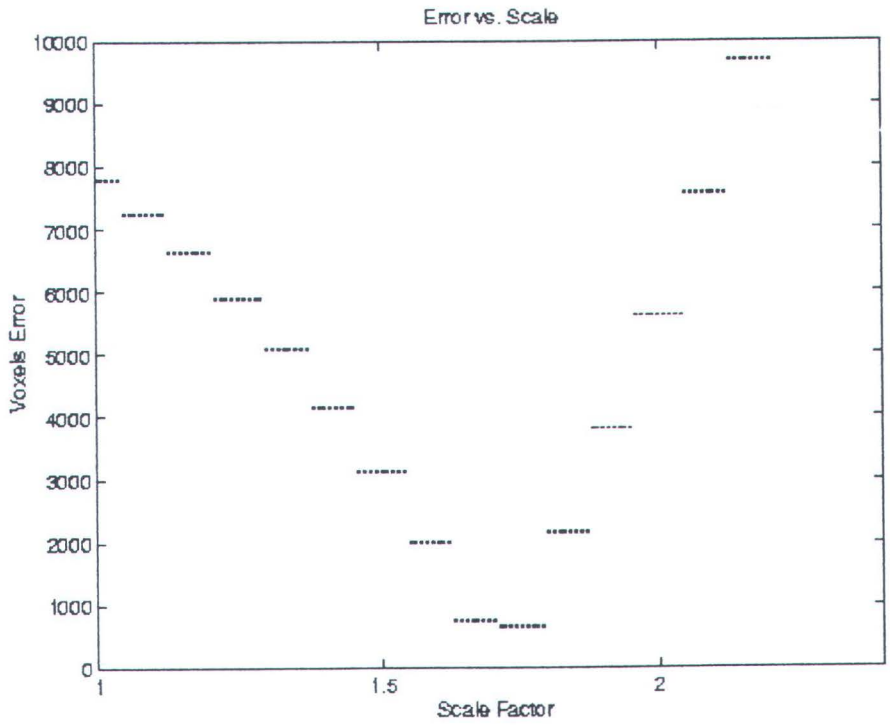
Gambar 2.4. Fungsi Kesalahan p2a

Pada setiap tahap penyekalaan, setiap titik objek semula dikalikan dengan suatu faktor skala. Hasilnya kemudian dibulatkan supaya termuat di dalam ruang diskrit. Pembulatan nilai-nilai ini mengakibatkan terjadinya hasil-hasil yang menarik. Adalah mungkin suatu objek yang dikalikan dengan bilangan besar dibulatkan untuk memiliki volume yang lebih kecil. Hal ini ditunjukkan pada Gambar 4 dan 5. Contoh berikut ini akan menjelaskan bagaimana hal ini dapat terjadi. Misalkan objek tersebut adalah ruas garis dari 1,4 sampai dengan 2,0. Pembulatannya adalah nilai 1 dan 2. Ruas garis tersebut mendapatkan 2 voxel bila diskalakan dengan 1. Ketika ruas garis ini diskalakan dengan 1,075, nilainya menjadi 1,505 dan 2,15. Kedua nilai ini dibulatkan menjadi 2 dan

menempati voxel yang sama. Dengan mengalikan dengan suatu bilangan besar, ruas garis tersebut menjadi lebih kecil setelah pembulatan. Mengalikan dengan bilangan di antara 1,075 dan 1,25 akan mengantarkan kepada hasil yang sama. Hal ini mengantarkan kepada suatu dataran pada fungsi kesalahan. Seluruh penjelasan ini ditunjukkan secara jelas pada Gambar 6. Gambar 4 menunjukkan sebuah plot dari fungsi kesalahan tersebut pada suatu benda. Bentuk benda tersebut memiliki banyak muka, sehingga banyak operasi pembulatan yang mempengaruhi hasil akhirnya. Gambar 5 menunjukkan plot yang lebih detail dari fungsi kesalahan p2a sekitar nilai-nilai minimum. Efek dataran terlihat lebih jelas di sini. Gambar 6 adalah plot fungsi kesalahan pada sebuah kubus. Plot ini menunjukkan bahwa nilai minimum tersebut terjadi pada sebuah dataran dan bukan nol. Adalah tidak mungkin untuk menyekalakan kubus menjadi volume yang diinginkan. Karena ruang yang dipergunakan bersifat diskrit, hanya terdapat volume diskrit yang dapat dicapai oleh volume kubus tersebut.



Gambar 2.5. Fungsi Kesalahan p2a, detail sekitar minimum



Gambar 2.6. Fungsi Kesalahan sebuah Kubus

Langkah pertama dalam meminimasi memakai metode interpolasi parabolik adalah mengurung nilai minimum tersebut, contohnya dengan menemukan suatu faktor penyekalaan yang lebih kecil dan lebih besar dari nilai optimal tersebut. $V_1(\alpha = 1) = m_{000}$. Penyimpangan-penyimpangan pada α dipakai untuk menemukan arah berkurangnya fungsi kesalahan tersebut. Ketika arah tersebut ditemukan, nilai kesalahan yang lebih kecil, E_2, dan $\alpha = \alpha_2$ yang berhubungan disimpan. Langkah-langkah yang lebih besar dengan arah ini diambil sampai kesalahan yang lebih besar, E_3 dan α_3 ditemukan. Kesalahan minimum terletak pada faktor skala antara α_1 dan α_3. Pada setiap tahap dalam algoritma tersebut, ketiga nilai kesalahan tersebut disimpan dalam urutan di mana α yang tengah mengacu pada kesalahan terkecil.

Pada masing-masing tahap, nilai baru ditentukan dengan asumsi bahwa fungsi kesalahan yang sesungguhnya secara kasar berbentuk parabola. Sebuah parabola menempati ketiga nilai yang mengurung nilai minimum. Nilai minimum parabola ini adalah titik yang baru.

Nilai minimum parabola $Ax^2 + Bx + C = y$ terletak pada $x = -\frac{B}{2A}$. A dan B ditentukan dari persamaan:

$$\begin{pmatrix} \alpha_1^2 & \alpha_1 & 1 \\ \alpha_2^2 & \alpha_2 & 1 \\ \alpha_3^2 & \alpha_3 & 1 \end{pmatrix} \begin{pmatrix} A \\ B \\ C \end{pmatrix} = \begin{pmatrix} V_1 \\ V_2 \\ V_3 \end{pmatrix}$$

A dan B dievaluasi dengan aturan Cramer dan disubstitusikan dalam persamaan tersebut untuk kesalahan minimum. Titik yang baru adalah:

$$x = -\frac{1}{2} \frac{\alpha_1^2(V_2 - V_3) + \alpha_2^2(V_3 - V_1) + \alpha_3^2(V_1 - V_2)}{V_1\alpha_2 + V_3\alpha_1 + V_2\alpha_3 - V_3\alpha_2 - V_1\alpha_3 - V_2\alpha_1}$$

$$= \alpha_2 - \frac{1}{2}\frac{(\alpha_2 - \alpha_1)^2(V_2 - V_3) - (\alpha_2 - \alpha_3)^2(V_2 - V_1)}{(\alpha_2 - \alpha_1)(V_2 - V_3) - (\alpha_2 - \alpha_3)(V_2 - V_1)}$$

Jika kesalahan titik baru tersebut sama dengan satu dari dua titik lain yang disimpan, tidak ada parabola yang menempati titik-titik tersebut dan faktor skala di antara dua titik yang menghasilkan kesalahan terkecil tersebut adalah titik yang baru. Ketika ketiga titik yang disimpan menghasilkan kesalahan yang sama, algoritma berakhir.

2.3.3. Orientasi Kanonikal

Kriteria untuk normalisasi orientasi diterapkan pada momen-momen pusat objek, μ_{lmn}. Kondisi pertama dipenuhi dengan melakukan rotasi pada objek sehingga matriks momen pusat orde kedua, M, menjadi diagonal.

$$M = \begin{pmatrix} \mu_{xx} & \mu_{xy} & \mu_{xz} \\ \mu_{xy} & \mu_{yy} & \mu_{yz} \\ \mu_{xz} & \mu_{yz} & \mu_{zz} \end{pmatrix} \qquad \mu_{xy} = \mu_{xz} = \mu_{yz} = 0$$

Kriteria ini menyesuaikan objek sehingga sumbu-sumbu pusat terletak sepanjang sumbu-sumbu koordinat. Terdapat delapan representasi objek yang memenuhi kriteria ini. Untuk menghilangkan ambiguitas tersebut, kondisi-kondisi lebih lanjut diterapkan pada objek. Sebuah permutasi diterapkan pada objek sehingga $\mu_{xx} > \mu_{yy} > \mu_{zz}$ dan perluasan maksimum objek adalah lebih besar dalam masing-masing setengah ruang positif.

Matriks momen, *M*, adalah simetrik. Matriks ini dapat difaktorkan menjadi matriks diagonal Λ dan matriks unitary *U*.

$$M = U\Lambda U^* = \int_{-\infty}^{\infty}\iint \begin{pmatrix} x \\ y \\ z \end{pmatrix} (x\,y\,z)\rho \begin{pmatrix} x \\ y \\ z \end{pmatrix} dx\,dy\,dz$$

$U^*U = UU^* = \mathrm{I}$, sehingga $U^*MU = UU^*\ \Lambda\ UU^* = \Lambda$

$$\Lambda = \int_{-\infty}^{\infty}\int\int U^* \begin{pmatrix} x \\ y \\ z \end{pmatrix} (x\,y\,z) U\rho \begin{pmatrix} x \\ y \\ z \end{pmatrix} dx\,dy\,dz$$

Dengan perubahan koordinat ini, $U^* \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$:

$$\Lambda = \int_{-\infty}^{\infty}\int\int \begin{pmatrix} a \\ b \\ c \end{pmatrix} (a\,b\,c)\rho \left(U \begin{pmatrix} a \\ b \\ c \end{pmatrix} \right) da\,db\,dc$$

Dengan U diterapkan pada fungsi densitas, ρ, matriks tersebut menjadi terdiagonalisasi. U adalah rotasi yang diterapkan.

Setelah rotasi diterapkan, yang bukan elemen diagonal dari matriks momen tersebut adalah nol. Setelah rotasi ini, orientasinya masih belum dinormalisasikan. Elemen-elemen diagonal dari matriks Λ, yang merupakan nilai-nilai Eigen dari matriks momen tersebut, M harus terurut terlebih dahulu. Tanda dan urutan nilai-nilai Eigen ini dinormalisasikan. Urutan tersebut dinormalisasikan dengan mempermutasi objek sehingga $\mu_{xx} > \mu_{yy} > \mu_{zz}$. Setelah pengurutan ini, masih terdapat delapan orientasi yang mungkin memenuhi kriteria tersebut. Sebuah permutasi akhir untuk menormalisasikan objek dilakukan. Sepanjang sumbu koordinat, setengah objek dengan perluasan lebih besar diposisikan pada arah positif.

2.3.4 Degenerasi

Ketika dua atau lebih momen-momen pusat diagonal, atau perluasan-perluasan dalam arah positif dan negatif adalah sama, metode ini tidak akan menghasilkan objek ternormalisasi yang unik. Dalam praktik, hal ini merupakan masalah yang timbul bukan hanya ketika kuantitasnya seimbang, namun juga bilamana mereka terdapat di dalam resolusi objek yang terdapat gangguan (noise). Objek-objek semacam ini bersifat simetrik pada setidaknya satu sumbu, yang dapat ditentukan. Objek-objek yang simetrik secara rotasional dua dimensi dapat dinormalisasikan. Beberapa metode yang dipakai untuk menormalisasikan objek-objek dua dimensi dapat diperluas untuk objek-objek tiga dimensi, namun komputasi deskriptor yang diperlukan sangat rumit dan mungkin tidak akurat karena komputasi numerik. Ketika degenerasi merupakan masalah, berbagai rotasi dan permutasi objek dapat dibandingkan dengan pola tersebut, atau ciri-ciri yang lain dapat dipergunakan. Dalam kasus degenerasi, ciri-ciri yang invarian terhadap rotasi, penyekalaan, dan translasi, *RST invariants*, dapat dipakai.

2.4 CIRI-CIRI

Setelah objek dinormalisasi, tahap berikutnya adalah membentuk file ciri-ciri (*feature file*) objek yang dinormalisasi tersebut. File ciri-ciri ini mengandung daftar atribut numerik objek yang membedakan objek dengan objek yang lain sehingga mudah diperbandingkan. Ciri-ciri dapat mengandalkan normalisasi objek, atau dapat invarian terhadap ukuran, posisi, dan orientasi objek. Ciri-ciri invarian ini lebih berguna daripada ciri-ciri lainnya dalam membandingkan objek-objek ketika objek yang diperbandingkan

$$= \int_{-\infty}^{\infty}\int\int \frac{(\sigma x)^l (\sigma y)^m (\sigma z)^n}{\sigma^{l+m+n}} \rho(\sigma x, \sigma y, \sigma z) \frac{d(\sigma x) d(\sigma y) d(\sigma z)}{\sigma^3}$$

$$= \frac{1}{\sigma^{l+m+n+3}} \int_{-\infty}^{\infty}\int\int (\sigma x)^l (\sigma y)^m (\sigma z)^n \rho(\sigma x, \sigma y, \sigma z) d(\sigma x) d(\sigma y) d(\sigma z)$$

$$= \frac{1}{\sigma^{l+m+n+3}} m_{lmn}$$

Faktor normalisasi ditentukan oleh volume terskala, $\frac{m_{000}}{\sigma^3}$. Momen invarian tersebut harus mengandung semua informasi tentang momen tersebut, m_{lmn}, namun harus invarian terhadap faktor skala σ. Invarians ini dapat direalisasikan dengan mengkombinasikan momen-momen dan volumenya secara aljabar untuk mengeliminasi σ.

$$\frac{\frac{m_{lmn}}{\sigma^{l+m+n+3}}}{\left(\frac{m_{000}}{\sigma^3}\right)^{\alpha}} = \frac{m_{lmn}}{m_{000}^{\alpha}} \frac{\sigma^{3\alpha}}{\sigma^{l+m+n+3}}$$

Dengan memilih $\alpha = 1 + \frac{l+m+n}{3}$, kuantitasnya menjadi invarian terhadap penyekalaan. Untuk momen-momen orde kedua $\alpha = \frac{5}{3}$. Perlu diperhatikan pula bahwa momen ke-0 yang disentralisasikan, m_{000}, momen-momen yang invarian terhadap skala, κ_{lmn}, dikalkulasi dengan persamaan berikut ini:

$$\kappa_{lmn} = \frac{\int_{-\infty}^{\infty}\int\int x^l y^m z^n \rho(x,y,z)\, dx\, dy\, dz}{\mu_{000}^{\frac{5}{3}}}$$

$$= \frac{\int_{-\infty}^{\infty}\int\int x^l y^m z^n \rho(\sigma x, \sigma y, \sigma z)\, dx\, dy\, dz}{\mu_{000}^{\frac{5}{3}}} \qquad l+m+n = 2$$

Karena fungsi karakteristik tersebut invarian terhadap rotasi, persamaan karakteristik matriks translasi dan momen-momen kedua yang invarian terhadap skala (κ_{lmn}), adalah invarian RST.

$$\begin{vmatrix} \kappa_{200}-\Lambda & \kappa_{110} & \kappa_{101} \\ \kappa_{110} & \kappa_{020}-\Lambda & \kappa_{011} \\ \kappa_{101} & \kappa_{011} & \kappa_{002}-\Lambda \end{vmatrix}$$

$$= -\Lambda^3 + \Lambda^2(\kappa_{200}+\kappa_{020}+\kappa_{002}) - \Lambda(\kappa_{002}\kappa_{020}+\kappa_{002}\kappa_{200}+\kappa_{020}\kappa_{200}-\kappa_{011}^2-\kappa_{101}^2-\kappa_{110}^2)$$

$$+(\kappa_{002}\kappa_{020}\kappa_{200}+2\kappa_{011}\kappa_{101}\kappa_{110}-\kappa_{101}^2\kappa_{020}-\kappa_{011}^2\kappa_{200}-\kappa_{110}^2\kappa_{002})$$

Koefisien pangkat Λ adalah ciri-ciri yang dipergunakan.

$$F_1 = \kappa_{200}+\kappa_{020}+\kappa_{002}$$

$$F_2 = \kappa_{002}\kappa_{020}+\kappa_{002}\kappa_{200}+\kappa_{020}\kappa_{200}-\kappa_{011}^2-\kappa_{101}^2-\kappa_{110}^2$$

$$F_3 = \kappa_{002}\kappa_{020}\kappa_{200}+2\kappa_{011}\kappa_{101}\kappa_{110}-\kappa_{101}^2\kappa_{020}-\kappa_{011}^2\kappa_{200}-\kappa_{110}^2\kappa_{002}$$

Invarian-invarian momen orde yang lebih tinggi mungkin dapat dihitung, namun hal ini rawan terkena degradasi noise. Untuk alasan ini, invarian-invarian hanya dikonstruksi pada momen-momen orde kedua.

2.4.2 Invarian Momen Kernel Sferis

Momen-momen selain momen geometrik dapat dikalkulasi. Contoh-contohnya adalah Legendre, Zernike, pseudo-Zernike, rotasional, dan kompleks. Kernel dalam setiap momen yang disebutkan ini menentukan tipenya. Seperti halnya invarian momen orde kedua, invarian-invarian momen kernel sferis memakai momen-momen pusat untuk

mendapatkan invarians translasi. Titik pusat objek adalah pusat sebuah bulatan S_i yang dipakai sebagai kernel momen tersebut. Ciri-ciri bulatan digenerasi dengan persamaan:

$$S f_i = \int_{-\infty}^{\infty} \iint S_i \rho(x, y, z)\, dx\, dy\, dz \qquad S_i = \begin{cases} 1 & r <= i \\ 0 & r > i \end{cases}$$

Sebagai tambahan selain invarian terhadap translasi, ciri-ciri ini juga invarian terhadap rotasi karena bulatan tersebut invarian terhadap rotasi. Invarian skala dicapai dengan menormalisasikan ciri-ciri tersebut terhadap suatu volume spesifik dan dengan mempergunakan fungsi densitas sebuah objek yang telah diskalakan menjadi volume standar.

Dua himpunan invarian momen kernel sferis telah diuji. Himpunan yang pertama digenerasi dengan sekumpulan bulatan dengan jari-jari yang meningkat secara linier dan dinormalisasikan menjadi nilai maksimum yang dapat dicapai ciri-ciri bulatan, $S f_i$.

$$S1_i = \frac{S f_i}{\int_0^{2\pi} \int_0^{\pi} \int_0^{i} d\phi\, d\theta\, dr}$$

Ciri-ciri ini menimbang semua jari-jari secara seimbang tanpa memberikan perhatian khusus pada daerah manapun dari objek tersebut. Ciri-ciri ini khususnya sesuai untuk mendeteksi variasi objek yang mengandung lubang-lubang, seperti mur. Jika objeknya solid, ciri-ciri ini mengandung sedikit informasi sampai radius bulatan tersebut sebagian terletak di luar objek.

Himpunan kedua dari invarian momen kernel sferis memberi perhatian lebih pada ujung terluar objek di mana objek-objek yang mirip mungkin berbeda. Ciri-ciri ini digenerasi dengan sebuah himpunan bulatan dengan jari-jari meningkat sebagai r^3, seperti yang dilakukan volume di dalam objek. Ciri-ciri ini dinormalisasi sesuai dengan volume objek.

$$S2_i = \frac{Sf_i}{m_{000}}$$

2.4.3 Ciri-ciri lainnya

Bermacam-macam ciri-ciri heuristik dikalkulasi dalam proses mengkalkulasi ciri-ciri tersebut di atas. Ciri-ciri semacam ini memasukkan dimensi kotak persegi terkecil yang disesuaikan dengan sumbu-sumbu pokok yang mengandung objek. Titik berat objek adalah yang lain dari ciri-ciri ini, dan area permukaan objek adalah ciri-ciri terakhir yang dikalkulasi.

2.5 PEMBANDINGAN

Setelah menormalisasi objek, sistem menempatkan bentuk yang dinormalisasi ke dalam database. Objek tersebut dideskripsikan secara lengkap dalam database ini dengan tiga buah file deskriptif :

1. Daftar permukaan segitiga yang mendefinisikan objek tersebut.
2. Ciri-ciri invarian RST yang dikalkulasi untuk objek tersebut.
3. Representasi voxel digital dari objek tersebut.

Permukaan segitiga tersebut dapat dipergunakan untuk membentuk ulang objek, atau untuk menampilkan objek, namun tidak dipergunakan untuk pembandingan. File invarian-invarian RST dapat dipergunakan untuk mengkalkulasi kemiripan objek berdasarkan kemiripan invarian-invarian RST tersebut. Karena ciri-ciri ini telah dipreproses, pencarian dapat dilakukan secara cepat dan efisien. File yang mengandung

representasi voxel digital dapat dipergunakan untuk menampilkan representasi voxel objek, untuk mengkalkulasi ciri-ciri baru, atau untuk membandingkan objek-objek yang telah dinormalisasikan dengan cara serupa. Pembandingan file-file ini bersifat intensif dan mungkin memakan banyak waktu. Karena hal itu, pembandingan ini sebaiknya hanya dilakukan untuk objek-objek yang mirip, berdasarkan invarian-invarian RST yang dikalkulasi.

Database tersebut pertama diperiksa dengan melakukan suatu query cosinus pada ciri-ciri relevan dalam file ciri-ciri. Ciri-ciri yang diekstraksi dari objek-objek dinormalisasi lebih lanjut sehingga memiliki perbandingan yang sama pada setiap elemen vektornya. Hal ini dilakukan supaya tidak ada ciri-ciri yang memiliki pengaruh lebih besar terhadap yang lain. Ciri-ciri dapat diberi bobot supaya memiliki pengaruh lebih kecil atau lebih besar. Vektor-vektor ciri-ciri dari invarian-invarian RST dibandingkan berdasarkan sudut di antara mereka. Ukuran perbandingan yang dipakai di sini adalah:

$$C_{xy} = 1 - 2\frac{\arccos\left(\frac{x.y}{|x||y|}\right)}{\pi}$$

Objek-objek serupa memiliki koefisien pembanding mendekati 1, sementara objek-objek yang sangat berbeda memiliki koefisien-koefisien mendekati 0. Prosedur ini sangat cepat, dan efisien untuk pencarian pada database yang berukuran besar.

Pencarian yang lebih intensif, namun kurang efisien, dapat dilakukan antara dua objek dengan membandingkan voxel-untuk-voxel.

Pembandingan Voxel-Voxel

Pencocokan pola adalah suatu pendekatan dalam membandingkan dua objek 3D. Ketidaksesuaian antara dua objek ternormalisasi dengan fungsi densitas f dan g dapat dikomputasi sebagai berikut:

$$\int_{-\infty}^{\infty}\int\int(f-g)^2\,dv = \int_{-\infty}^{\infty}\int\int f^2\,dv + \int_{-\infty}^{\infty}\int\int g^2\,dv - 2\int_{-\infty}^{\infty}\int\int f\,g\,dv$$

Untuk setiap objek, nilai $\int_{-\infty}^{\infty}\int\int f^2\,dv$ dan $\int_{-\infty}^{\infty}\int\int g^2\,dv$ adalah tetap. Ketidaksesuaian yang kecil terjadi hanya jika kesesuaian $\int_{-\infty}^{\infty}\int\int f\,g\,dv$ besar. Untuk representasi digital objek-objek, kesesuaian voxel-voxel M_{fg} diukur secara analog.

$$M_{fg} = \frac{\sum_i\sum_j\sum_k f(i,j,k)g(i,j,k)}{Max\left(\sum_i\sum_j\sum_k f(i,j,k), \sum_i\sum_j\sum_k g(i,j,k)\right)}$$

Rumus ini mencocokkan f dan g voxel-untuk-voxel, sehingga juga mengukur kecocokan bentuk maupun perbedaan orientasi. Jika kedua objek dimanipulasi menjadi posisi, ukuran, dan orientasi kanonikal, kesesuaian voxel-voxel mengukur kemiripan geometrik.

Perbedaan simetrik voxel tidak selalu merupakan pembandingan yang lebih baik daripada invarian-invarian RST. Meskipun pembandingan voxel lebih intensif dan biasanya menghasilkan pembandingan yang lebih baik, pembandingan ini tidak berarti untuk objek-objek yang simetris secara rotasional jika mereka objek-objek ini tidak dijajarkan pada suatu orientasi yang unik. Jika objek tersebut simetrik secara rotasional, misalnya momen-momen orde keduanya hampir sama, perbedaan voxel simetrik tidak boleh dipergunakan sebagai pembandingan kecuali kriteria lebih jauh dipakai untuk

menjajarkan objek-objek tersebut. Metode-metode ini dapat melibatkan penghitungan momen-momen kompleks, atau deskriptor Fourier yang dimodifikasi. Keduanya memerlukan banyak komputasi dan telah dikembangkan untuk objek-objek dua dimensi.

2.6 ANALISIS KESALAHAN

Representasi digital dipakai untuk merepresentasikan objek-objek yang mulus (*smooth*). Hal ini melibatkan kuantisasi dan aproksimasi yang memperkenalkan kesalahan-kesalahan yang mungkin terjadi dalam pencocokan objek-objek. Kesalahan-kesalahan ini harus dipahami dan dikuantifikasi sehingga dapat diminimasi dan sehingga suatu tingkat kepastian dapat diterapkan pada hasil-hasil yang dicapai. Di bawah kondisi tertentu, beberapa ciri-ciri lebih baik dari yang lain dan beberapa metode pembandingan bekerja lebih baik dari yang lain. Contohnya, pembandingan voxel-voxel tidak akan dapat diandalkan untuk membandingkan objek-objek yang simetrik secara rotasional jika objek-objek ini tidak dijajarkan dalam suatu orientasi yang unik. Jika rasio momen-momen orde keduanya mendekati satu, kepastian pembandingan voxelnya rendah.

Karena titik pusat suatu objek menempati suatu voxel, dan tidak dapat terletak di antara voxel-voxel, titik pusat yang dikalkulasi akan terletak dalam sebuah voxel pada setiap arah dari titik pusat yang sebenarnya. Sebuah kesalahan dalam mengkalkulasi titik pusat tersebut akan mempengaruhi kalkulasi lainnya. Semua momen pusat akan dikalkulasi pada suatu pendekatan titik pusat. Sebagai tambahan, objek tersebut akan dirotasikan pada pusat ini. Kesalahan-kesalahan ini dapat dideteksi dalam matriks momen representasi kanonikal objek. Jika pusat dan momen-momen tersebut dikalkulasi secara tepat, dan efek-efek pembulatan permukaan dapat diabaikan, sumbu-sumbu pokok objek

dapat dijajarkan secara tepat dengan sumbu-sumbu koordinat. Entri-entri diagonal off dalam matriks derajat kedua akan sama dengan nol. Deviasi dari nilai-nilai ini dari nol akan mengkuantifikasikan kesalahan yang terjadi dalam penjajaran objek yang dapat disebabkan oleh kesalahan yang terjadi akibat kesalahan digitasi titik pusat. Kesalahan-kesalahan ini dapat diperkecil dengan memakai rasio kuantisasi yang lebih tinggi. Pengorbanan yang harus dilakukan adalah kecepatan pemrosesan dan ruang penyimpanan. Semakin banyak rasio sampling, lebih kecil kesalahan yang terjadi, namun karena lebih banyak voxel per objek, masing-masing objek memerlukan lebih banyak memori untuk menyimpannya dan komputasi yang lebih banyak diperlukan dalam normalisasi dan pembandingan objek-objek tersebut.

2.6.1 Komputasi Momen-ınomen

Momen-momen tiga dimensi dikalkulasi dengan persamaan pada awal sub bab 2.3. Integralnya dikonversi menjadi bentuk penjumlahan untuk kalkulasi komputer.

$$m_{lmn} = \int_{-\infty}^{\infty}\int\int x^l y^m z^n \rho(x,y,z)\,dx\,dy\,dz \qquad l,m,n = 1,2,3,\ldots$$

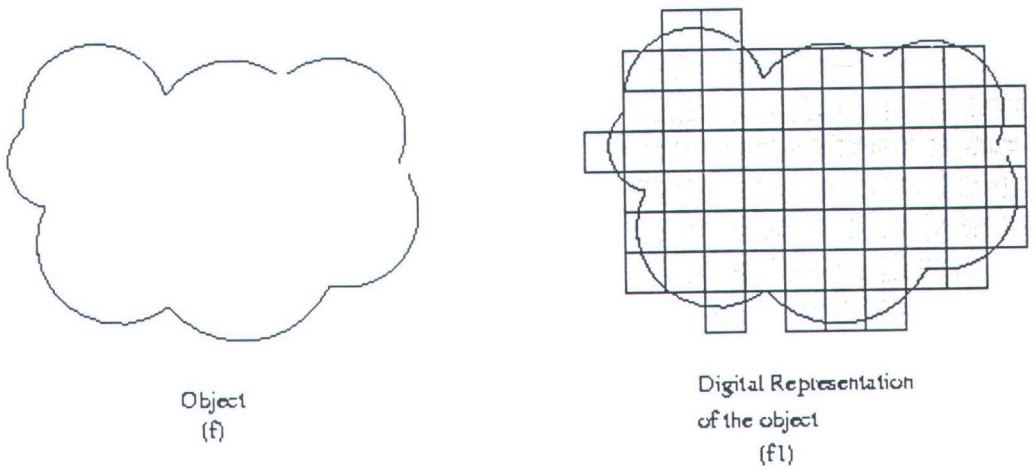
$$\approx \sum_{-\infty}^{\infty}\sum\sum \int_{z-\frac{1}{2}}^{z+\frac{1}{2}}\int_{y-\frac{1}{2}}^{y+\frac{1}{2}}\int_{x-\frac{1}{2}}^{x+\frac{1}{2}} x^l y^m z^n \rho(x,y,z) \qquad l,m,n = 1,2,3,\ldots$$

Penjumlahan tersebut dipakai untuk menambahkan kontribusi setiap voxel kepada momen tersebut. Integral dalam persamaan tersebut dipakai untuk mengkalkulasi kontribusi yang dimiliki setiap voxel terhadap momen. Adalah mungkin untuk mengkalkulasi nilai tersebut dengan integral ini dan mendapatkan sebuah polinom

sebagai hasilnya. Hasil tersebut kemudian dipergunakan dalam program untuk mengkalkulasi nilai momen. Misalnya, hasil dari integral ini untuk m_{200} adalah $x^2 + 1/12$. Kalkulasi ini, yang dilakukan ketika pemrograman kalkulasi momen-momen mengurangi kesalahan kalkulasi selama komputasi.

2.6.2 Efek Permukaan

Kesalahan-kesalahan digital juga mempengaruhi permukaan batas setiap objek. Kecuali objek tersebut memiliki permukaan rata yang terletak secara tepat pada batas-batas voxel, akan terjadi beberapa kesalahan pada voxel-voxel yang terletak pada permukaan.Voxel-voxel yang terletak di dalam objek tidak memiliki kesalahan serupa. Kesalahan pada permukaan memiliki batas atas yang tergantung kepada besarnya permukaan objek. Gambar 7 menunjukkan suatu objek f dan aproksimasi digitalnya f_1.



Gambar 7: Suatu Objek dengan Aproksimasi Digitalnya

Berikut ini akan menunjukkan batas-batas pada kesalahan tersebut, E, karena efek permukaan untuk sebuah kalkulasi momen.

$$E = \left| \int f\, g\, dv - \int f_1\, g\, dv \right| \leq \int |f - f_1| |g|\, dv$$

$|f - f_1|$ bernilai nol kecuali pada posisi dekat permukaan. Jika volume semua voxel pada permukaan adalah S, maka:

$$\int |f - f_1| \, dv \leq S$$

$$\int |f - f_1||g| \, dv \leq S \int |g| \, dv$$

Kalkulasi ini menghitung momen yang benar untuk representasi digital objek, namun representasi digital tersebut mengandung kesalahan permukaan.

2.6.3 Kesalahan dalam Normalisasi

Dengan beranggapan bahwa objek digital tersebut benar dengan mengabaikan efek permukaan, masih terdapat kesalahan-kesalahan dalam normalisasi. Posisi kanonikal secara esensial diperoleh di dalam 1 voxel dari pusat, jadi terdapat sedikit kesalahan dalam fase normalisasi tersebut. Normalisasi ukuran objek juga akan memiliki beberapa kesalahan. Seperti yang telah dibahas sebelumnya, tidaklah mungkin untuk mencapai ukuran tujuan suatu objek. Setiap perbedaan dalam volume sesungguhnya dan volume tujuan adalah kesalahan.

Kesalahan lain dalam normalisasi adalah dalam orientasi. Jika sumbu pusat objek secara tepat dijajarkan pada sumbu koordinat, matriks M akan menjadi diagonal. Setiap deviasi dari elemen-elemen diagonal-off dari nol adalah kesalahan dalam normalisasi. Secara teoretis, istilah diagonal-off haruslah nol jika objek tersebut diorientasikan secara tepat. Tiga potong informasi diperoleh dari elemen-elemen diagonal matriks. Dalam praktiknya, diagonal-off tersebut mendekati, namun tidak secara tepat sama dengan nol.Matriks tersebut mengandung informasi lebih dari yang terdapat pada ketiga elemen

diagonal dalam kasus ini. Inilah mengapa ciri-ciri dalam persamaan 19 dipergunakan. Ciri-ciri ini kebal terhadap kesalahan-kesalahan dalam orientasi. Pembandingan lain seperti pembandingan voxel tidak sedemikian tangguh. Kesalahan-kesalahan dalam orientasi akan mempengaruhi pembandingan ini. Nilai elemen-elemen diagonal-off akan dapat dipergunakan untuk mengkuantisasi seberapa baik objek diorientasikan sebagai dasar untuk kepastian pembandingan voxel tersebut.

BAB III

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK

Dalam perancangan dan pembuatan perangkat lunak, terdapat beberapa hal yang berbeda dengan teori yang telah dijelaskan dalam bab 2. Perbedaan-perbedaan ini antara lain disebabkan karena kurang lengkapnya penjelasan yang disebutkan dalam bab 2 dan karena tidak diimplementasikannya beberapa teori yang telah disebutkan.

3.1. PEMBACAAN FILE DXF

3.1.1. Pemindaian File DXF

Secara garis besar, modul pembaca file DXF ini berupa pemindai (*scanner*) yang melakukan *scanning* terhadap file DXF yang dihasilkan oleh 3DSMAX. Modul ini terdiri dari kelas-kelas yang berfungsi untuk mengakses file DXF, untuk merepresentasikan objek-objek 3 dimensi (dalam file tersebut), pemindai (*scanner*), pengenal *token-token* yang dipindai, serta *stack* untuk menampung *token-token* tersebut untuk keperluan pemindaian.

Pembacaan diawali dengan membuka file DXF yang diinginkan, lalu pemindaian dilakukan mulai dari awal file sampai dengan *end-of-file*. Pemindaian dilakukan secara baris per baris, di mana informasi yang terdapat pada satu baris tersebut diperlakukan

sebagai *token*. *Token* ini dicocokkan nilainya pada tabel *token*, untuk kemudian dipakai dalam pembacaan nilai yang diasosiasikan dengan *token* tersebut pada baris berikutnya.

Pemindaian file DXF ini mengabaikan sejumlah informasi yang terdapat pada file DXF tersebut. Bagian yang diabaikan adalah HEADER dan TABLES sehingga pemindaian dapat langsung mengakses bagian ENTITIES. Pada bagian ENTITIES, dilakukan pembacaan terhadap setiap POLYLINE (objek 3 dimensi).

Penyimpanan data pada file DXF ini merepresentasikan objek-objek 3 dimensi menjadi blok-blok POLYLINE sehingga struktur data internal pada modul pembaca file DXF ini pun disesuaikan dengan metode penyimpanan tersebut. Metode representasi yang dipakai adalah dengan terlebih dahulu mendefinisikan dan memberi indeks kepada seluruh titik yang merupakan anggota dari *polyline* tersebut, setelah itu mendefinisikan seluruh bidang / poligon (segitiga) yang merupakan sisi – sisi objek 3 dimensi tersebut. Definisi poligon-poligon tersebut dilakukan dengan cara memakai indeks titik-titik yang telah didefinisikan dan diindeks sebelumnya.

Pada setiap polyline, baik pendefinisian titik maupun poligon memakai token yang sama, yaitu VERTEX. Untuk membedakan kedua hal ini, yang menandakan adalah nilai yang diasosiasikan pada token '70'. Nilai 192 sesudah token tersebut menunjukkan bahwa *vertex* tersebut adalah titik, sedangkan nilai 128 sesudah token tersebut menunjukkan bahwa *vertex* tersebut adalah poligon.

Dengan demikian, hasil modul ini adalah larik (*array*) objek 3 dimensi, di mana setiap objek 3 dimensi direpresentasikan dengan larik yang memuat titik-titik sudut dalam objek tersebut, serta sebuah larik yang memuat poligon-poligon sebagai sisi-sisi objek tersebut.

3.1.2. Penampilan Objek 3 Dimensi

Penampilan objek-objek 3 dimensi yang telah dipindai dilakukan dengan memakai OpenGL. Karena tujuan utama tugas akhir ini bukanlah untuk menampilkan objek-objek tersebut sebaik mungkin melainkan untuk melakukan pengenalan terhadap objek-objek tersebut, maka penampilan objek-objek tersebut dilakukan dengan cara yang sangat sederhana, yaitu dengan mengasosiasikan nilai warna yang dipilih secara acak pada setiap titik yang merupakan bagian dari objek 3 dimensi.

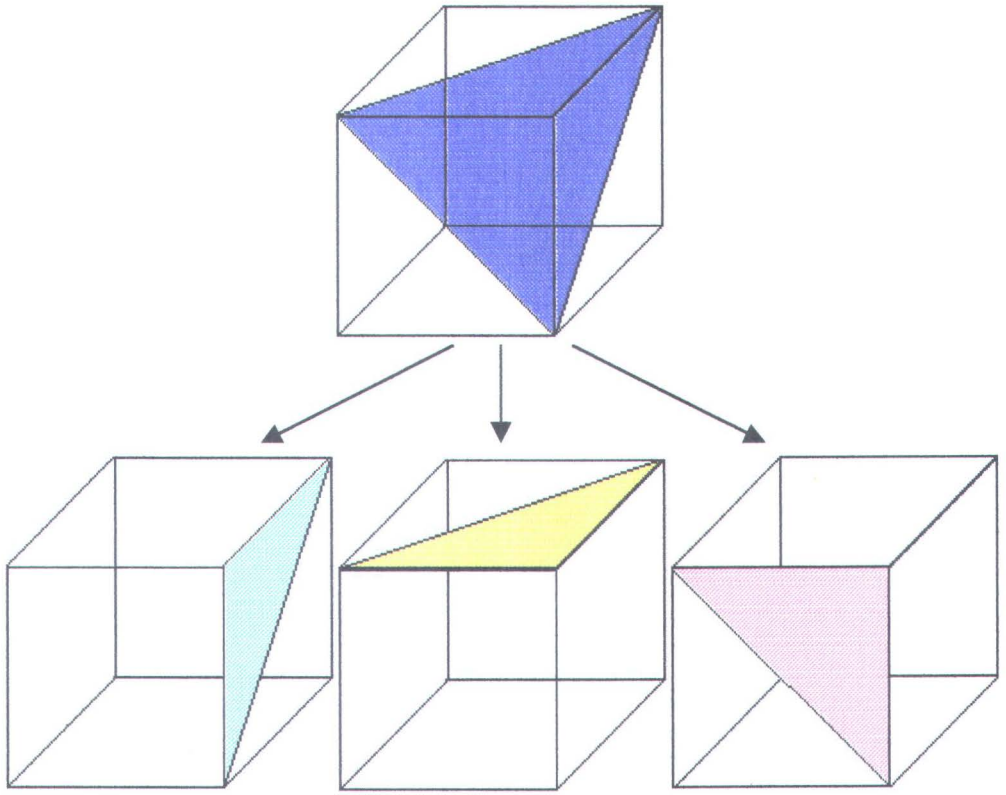
3.2. TRANSFORMASI OBJEK MENJADI REPRESENTASI VOXEL

Tujuan modul ini adalah untuk membentuk representasi voxel dari representasi objek 3 dimensi yang dihasilkan oleh modul sebelumnya. Representasi voxel yang dimaksud adalah larik (*array*) tiga dimensi yang bernilai *Boolean* (*True / False*) yang merupakan fungsi okupansi ($\rho(x,y,z)$) di mana *True* berarti bahwa sel tersebut merupakan bagian dari objek (*solid*) sedangkan *False* berarti kosong (*void*).

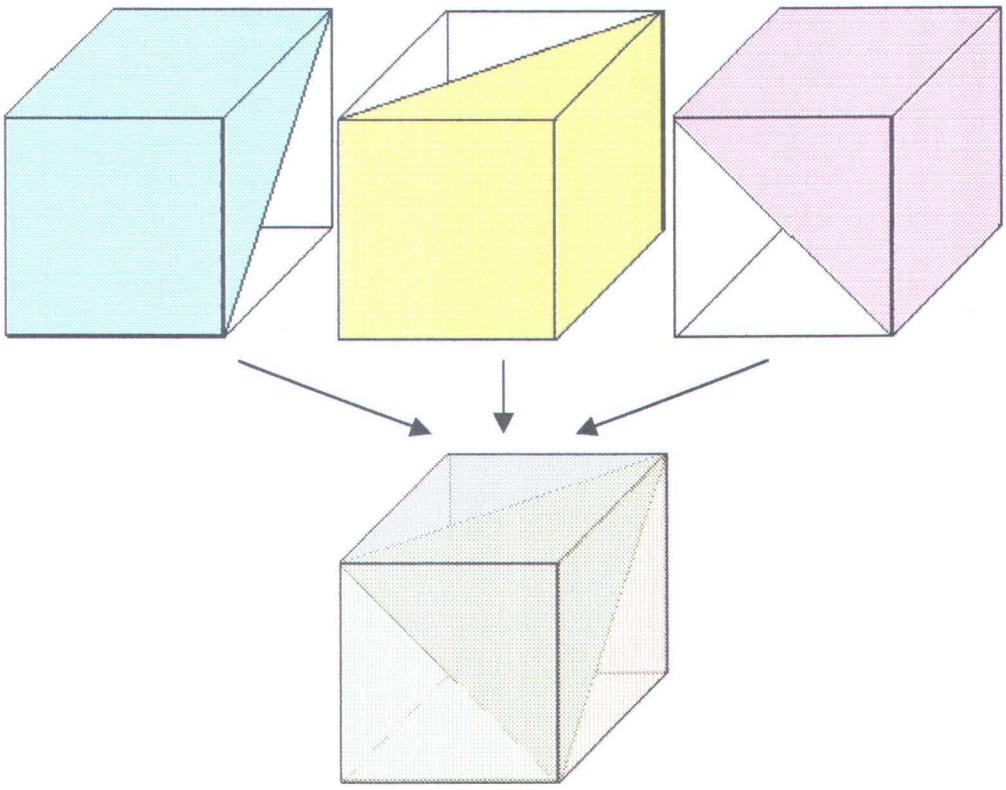
Proses transformasi diawali dengan mendiskritisasi poligon-poligon (segitiga-segitiga) ke dalam ruang diskrit.

3.2.1. Diskritisasi Segitiga

Di dalam proses ini, poligon – poligon (segitiga – segitiga) yang melambangkan sisi – sisi setiap objek 3 dimensi didiskritisasi dengan cara memproyeksikannya ke masing – masing bidang datar yang menjadi dasar koordinat Cartesian (X=0 atau bidang YOZ, Y=0 atau bidang XOZ, dan Z=0 atau bidang XOY) sehingga segitiga tersebut menjadi "tiga" buah segitiga hasil proyeksi.



Diskritisasi



Gambar 3.1. Diskritisasi Segitiga

Dalam gambar tersebut terlihat penguraian segitiga 3 dimensi menjadi tiga segitiga 2 dimensi. Segitiga-segitiga 2 dimensi ini didiskritisasi dengan cara dipetakan menjadi *bitmap*. *Bitmap* ini besarnya disesuaikan dengan ukuran sampling objek-objek 3 dimensi dalam file tersebut. Selanjutnya, dibuat sebuah larik (*array*) 3 dimensi seukuran bitmap-bitmap tersebut untuk memperluas segitiga – segitiga diskrit tersebut menjadi tiga prisma diskrit. Ketiga buah prisma segitiga ini diiriskan menjadi sebuah limas diskrit segitiga. Kemudian seharusnya limas ini diproses lebih lanjut sehingga bentuknya menjadi "segitiga" diskrit tiga dimensi. Namun sampai penulis menyelesaikan tugas akhir ini, penulis masih belum menemukan metode yang tepat untuk mengimplementasikannya.

3.2.2. Penentuan Bagian Luar-Dalam Objek-objek 3 Dimensi

Setelah segitiga-segitiga tersebut didiskritisasi, maka larik tiga dimensi tersebut telah berisi informasi mengenai voxel-voxel mana saja yang merupakan representasi sisi-sisi objek 3 dimensi tersebut. Proses dilanjutkan dengan melakukan proses *flood-filling* seperti yang telah disebutkan di bab 2, di mana kisaran (*range*) larik tiga dimensi tersebut diperluas sebanyak 2 voxel pada setiap sumbunya (x,y,z). *Flood-filling* ini memakai tiga larik sementara untuk menandai voxel-voxel mana saja yang termasuk 'di dalam objek' untuk masing-masing orientasi sumbu. Hasil ketiga larik sementara ini dipakai untuk menandai larik tiga dimensi yang semula dengan cara menandai sel-sel dalam larik tersebut yang pada ketiga larik sementara bernilai *True* (di dalam benda).

3.3. NORMALISASI

Normalisasi dilakukan supaya objek – objek yang dibandingkan nantinya memiliki kriteria yang sama sehingga meminimalisasi distorsi informasi yang disebabkan karena perbedaan ukuran (volume), posisi (relatif terhadap pusat koordinat), serta orientasi.

Untuk mengimplementasikan hal ini, penghitungan momen-momen dilakukan dengan pendekatan numeris sebagai berikut :

$$m_{lmn} = \sum_{-\infty}^{\infty}\sum\sum \int_{z-\frac{1}{2}}^{z+\frac{1}{2}} \int_{y-\frac{1}{2}}^{y+\frac{1}{2}} \int_{x-\frac{1}{2}}^{x+\frac{1}{2}} x^l y^m z^n \rho(x,y,z)$$

Pendekatan ini dapat menghasilkan bentuk polinomial. Untuk contoh m_{200}, penyederhanaan analitis rumus di atas dapat dilakukan sebagai berikut:

$$m_{200} = \sum_{-\infty}^{\infty}\sum\sum \int_{z-\frac{1}{2}}^{z+\frac{1}{2}} \int_{y-\frac{1}{2}}^{y+\frac{1}{2}} \int_{x-\frac{1}{2}}^{x+\frac{1}{2}} x^2 y^0 z^0 \rho(x,y,z)$$

$$m_{200} = \sum_{-\infty}^{\infty}\sum\sum \int_{z-\frac{1}{2}}^{z+\frac{1}{2}} \int_{y-\frac{1}{2}}^{y+\frac{1}{2}} \left[\frac{1}{3}x^3\right]_{x-\frac{1}{2}}^{x+\frac{1}{2}} \rho(x,y,z)$$

$$m_{200} = \sum_{-\infty}^{\infty}\sum\sum \int_{z-\frac{1}{2}}^{z+\frac{1}{2}} \int_{y-\frac{1}{2}}^{y+\frac{1}{2}} \frac{1}{3}\left\{\left(x+\frac{1}{2}\right)^3 - \left(x-\frac{1}{2}\right)^3\right\} \rho(x,y,z)$$

$$m_{200} = \sum_{-\infty}^{\infty}\sum\sum \int_{z-\frac{1}{2}}^{z+\frac{1}{2}} \int_{y-\frac{1}{2}}^{y+\frac{1}{2}} \frac{1}{3}\left\{\left(x^3+\frac{3}{2}x^2+\frac{3}{4}x+\frac{1}{8}\right) - \left(x^3-\frac{3}{2}x^2+\frac{3}{4}x-\frac{1}{8}\right)\right\} \rho(x,y,z)$$

$$m_{200} = \sum_{-\infty}^{\infty}\sum\sum \int_{z-\frac{1}{2}}^{z+\frac{1}{2}} \int_{y-\frac{1}{2}}^{y+\frac{1}{2}} \frac{1}{3}\left\{3x^2 + \frac{1}{4}\right\}\rho(x,y,z)$$

$$m_{200} = \sum_{-\infty}^{\infty}\sum\sum \int_{z-\frac{1}{2}}^{z+\frac{1}{2}} \int_{y-\frac{1}{2}}^{y+\frac{1}{2}} x^2 + \frac{1}{12}\rho(x,y,z)$$

$$m_{200} = \sum_{-\infty}^{\infty}\sum\sum \int_{z-\frac{1}{2}}^{z+\frac{1}{2}} \left(x^2 + \frac{1}{12}\right)[y]_{y-\frac{1}{2}}^{y+\frac{1}{2}}\rho(x,y,z)$$

$$m_{200} = \sum_{-\infty}^{\infty}\sum\sum \int_{z-\frac{1}{2}}^{z+\frac{1}{2}} \left(x^2 + \frac{1}{12}\right)\left(\left(y+\frac{1}{2}\right)-\left(y-\frac{1}{2}\right)\right)\rho(x,y,z)$$

$$m_{200} = \sum_{-\infty}^{\infty}\sum\sum \int_{z-\frac{1}{2}}^{z+\frac{1}{2}} \left(x^2 + \frac{1}{12}\right)(1)\rho(x,y,z)$$

$$m_{200} = \sum_{-\infty}^{\infty}\sum\sum \left(x^2 + \frac{1}{12}\right)[z]_{z-\frac{1}{2}}^{z+\frac{1}{2}}\rho(x,y,z)$$

$$m_{200} = \sum_{-\infty}^{\infty}\sum\sum \left(x^2 + \frac{1}{12}\right)\left(\left(z+\frac{1}{2}\right)-\left(z-\frac{1}{2}\right)\right)\rho(x,y,z)$$

$$m_{200} = \sum_{-\infty}^{\infty}\sum\sum \left(x^2 + \frac{1}{12}\right)(1)\rho(x,y,z)$$

$$m_{200} = \sum_{-\infty}^{\infty}\sum\sum \left(x^2 + \frac{1}{12}\right)\rho(x,y,z)$$

Karena yang menjadi perhatian di sini hanyalah momen orde kedua, maka rumus umum tersebut dapat dijabarkan secara spesifik untuk langsung dikalkulasi guna meningkatkan kecepatan penghitungan sebagai berikut :

$$m_{xx} = m_{200} = \sum_{-\infty}^{\infty}\sum\sum\left(x^2 + \frac{1}{12}\right)\rho(x,y,z)$$

$$m_{yy} = m_{020} = \sum_{-\infty}^{\infty}\sum\sum\left(y^2 + \frac{1}{12}\right)\rho(x,y,z)$$

$$m_{zz} = m_{002} = \sum_{-\infty}^{\infty}\sum\sum\left(z^2 + \frac{1}{12}\right)\rho(x,y,z)$$

$$m_{xy} = m_{110} = \sum_{-\infty}^{\infty}\sum\sum xy\rho(x,y,z)$$

$$m_{xz} = m_{101} = \sum_{-\infty}^{\infty}\sum\sum xz\rho(x,y,z)$$

$$m_{yz} = m_{011} = \sum_{-\infty}^{\infty}\sum\sum yz\rho(x,y,z)$$

3.3.1. Normalisasi Posisi

Implementasi untuk normalisasi posisi dilakukan tepat seperti teori yang telah dipaparkan dalam bab 2. Untuk menormalisasi posisi, yang perlu dilakukan adalah menghitung titik berat (*centroid*) objek (atau objek-objek, bila terdapat lebih dari satu objek dalam sistem tersebut) 3 dimensi.

Seperti yang telah disebutkan dalam teori di bab 2, titik berat didefinisikan sebagai:

$$(\hat{x},\hat{y},\hat{z}) = \left(\frac{m_{100}}{m_{000}}, \frac{m_{010}}{m_{000}}, \frac{m_{001}}{m_{000}}\right)$$

Fungsi densitas yang ternormalisasi posisi adalah:

$$\hat{\rho}(x,y,z) = \rho(x-\hat{x}, y-\hat{y}, z-\hat{z})$$

Momen-momen dari fungsi densitas tersebut adalah momen-momen pusat.

$$\mu_{lmn} = \int_{-\infty}^{\infty}\int\int (x-\hat{x})^l (y-\hat{y})^m (z-\hat{z})^n \rho(x-\hat{x}, y-\hat{y}, z-\hat{z}) dxdydz \qquad l,m,n = 0,1,2,3,\ldots$$

3.3.2. Normalisasi Ukuran[1]

Normalisasi ukuran yang disebutkan dalam sub bab 2.4.1 dapat dilakukan dengan cara membagi nilai momen dengan $\mu_{000}^{\frac{5}{3}}$; di mana μ_{000} adalah volume hasil diskritisasi benda tersebut. Dengan demikian, momen-momen yang ternormalisasi terhadap ukuran dapat dihitung dengan cara sebagai berikut :

$$\kappa_{lmn} = \frac{\int_{-\infty}^{\infty}\int\int x^l y^m z^n \rho(x,y,z)\, dx\, dy\, dz}{\mu_{000}^{\frac{5}{3}}} \; ; \; l+m+n = 2.$$

3.3.3. Normalisasi Orientasi

Dalam implementasinya, normalisasi orientasi tidak diimplementasikan seperti yang telah disebutkan dalam bab 2 (2.3.3.), karena pada tahap selanjutnya, ciri-ciri yang dihitung sudah bersifat invarian terhadap orientasi.

---

[1] Untuk menormalisasi ukuran, penulis membuktikan bahwa teori yang disebutkan di bab 2 (2.3.2) tidaklah berperan dalam penghitungan untuk tahap selanjutnya.

3.4. PENGHITUNGAN CIRI-CIRI OBJEK

Yang dimaksudkan dengan ciri-ciri objek dalam hal ini adalah serupa dengan yang disebutkan dalam bab 2 (2.4.1.). Ciri-ciri tersebut diformulasikan sebagai berikut:

$$F_1 = \eta_{200} + \eta_{020} + \eta_{002}$$

$$F_2 = \eta_{002}\eta_{020} + \eta_{002}\eta_{200} + \eta_{020}\eta_{200} - \eta_{011}^2 - \eta_{101}^2 - \eta_{110}^2$$

$$F_3 = \eta_{002}\eta_{020}\eta_{200} + 2\eta_{011}\eta_{101}\eta_{110} - \eta_{101}^2\eta_{020} - \eta_{011}^2\eta_{200} - \eta_{110}^2\eta_{002}$$

η_{lmn} yang disebutkan dalam rumus-rumus di atas adalah momen-momen yang telah dinormalisasikan terhadap translasi dan ukuran (*size*). η_{lmn} memiliki formulasi matematis:

$$\eta_{lmn} = \frac{\mu_{lmn}}{\mu_{000}^{\frac{5}{3}}}$$

Karena $\mu_{lmn} = \int_{-\infty}^{\infty}\int\int (x-\hat{x})^l (y-\hat{y})^m (z-\hat{z})^n \rho(x-\hat{x}, y-\hat{y}, z-\hat{z}) dxdydz$, maka rumus di atas dapat dijabarkan lebih lanjut berupa:

$$\eta_{lmn} = \frac{\int_{-\infty}^{\infty}\int\int (x-\hat{x})^l (y-\hat{y})^m (z-\hat{z})^n \rho(x-\hat{x}, y-\hat{y}, z-\hat{z}) dxdydz}{\mu_{000}^{\frac{5}{3}}}$$

Dalam tugas akhir ini, momen-momen yang dipergunakan hanyalah momen-momen orde kedua (untuk menghitung ciri-ciri), orde pertama (untuk menghitung titik pusat atau *centroid*), serta orde nol (volume). Maka dari itu, secara numerik rumus-rumus yang dipergunakan untuk menghitung momen-momen orde kedua yang telah dijelaskan pada awal sub bab ini (3.3) dapat diperjelas berdasarkan rumus umum di atas sebagai berikut:

$$\mu_{000} = \sum_{-\infty}^{\infty}\sum\sum \rho(x-\hat{x}, y-\hat{y}, z-\hat{z}) = m_{000} = \sum_{-\infty}^{\infty}\sum\sum \rho(x,y,z)$$

$$\eta_{xx} = \eta_{200} = \frac{\sum_{-\infty}^{\infty}\sum\sum\left((x-\hat{x})^2 + \frac{1}{12}\right)\rho(x-\hat{x}, y-\hat{y}, z-\hat{z})}{\mu_{000}^{\frac{5}{3}}}$$

$$\eta_{yy} = \eta_{020} = \frac{\sum_{-\infty}^{\infty}\sum\sum\left((y-\hat{y})^2 + \frac{1}{12}\right)\rho(x-\hat{x}, y-\hat{y}, z-\hat{z})}{\mu_{000}^{\frac{5}{3}}}$$

$$\eta_{zz} = \eta_{002} = \frac{\sum_{-\infty}^{\infty}\sum\sum\left((z-\hat{z})^2 + \frac{1}{12}\right)\rho(x-\hat{x}, y-\hat{y}, z-\hat{z})}{\mu_{000}^{\frac{5}{3}}}$$

$$\eta_{xy} = \eta_{110} = \frac{\sum_{-\infty}^{\infty}\sum\sum(x-\hat{x})(y-\hat{y})\rho(x-\hat{x}, y-\hat{y}, z-\hat{z})}{\mu_{000}^{\frac{5}{3}}}$$

$$\eta_{xz} = \eta_{101} = \frac{\sum_{-\infty}^{\infty}\sum\sum(x-\hat{x})(z-\hat{z})\rho(x-\hat{x}, y-\hat{y}, z-\hat{z})}{\mu_{000}^{\frac{5}{3}}}$$

$$\eta_{yz} = \eta_{011} = \frac{\sum_{-\infty}^{\infty}\sum\sum(y-\hat{y})(z-\hat{z})\rho(x-\hat{x}, y-\hat{y}, z-\hat{z})}{\mu_{000}^{\frac{5}{3}}}$$

3.5. PEMBANDINGAN OBJEK-OBJEK

Pembandingan objek-objek dilakukan tepat seperti yang telah disebutkan dalam dasar teori (2.5.), di mana rumus yang dipergunakan adalah:

$$C_{xy} = 1 - 2\frac{\arccos\left(\frac{x.y}{|x||y|}\right)}{\pi}$$

di mana x dan y adalah vektor – vektor ciri-ciri yang diperbandingkan. Baik x maupun y merupakan vektor yang terdiri atas 3 elemen, berupa ciri-ciri yang dihasilkan oleh proses penghitungan ciri-ciri yang telah dilakukan sebelumnya.

Karena normalisasi terhadap elemen–elemen vektor–vektor ini perlu dilakukan lebih lanjut supaya tidak ada suatu elemen vektor yang lebih dominan terhadap elemen vektor yang lain dalam menentukan similaritas kedua vektor (objek), maka rumus di atas diimplementasikan dengan cara sebagai berikut:

$$C_{xy} = 1 - 2\frac{\left(\arccos\left(\frac{\begin{bmatrix}\frac{x_1}{x_1}\\ \frac{x_2}{x_2}\\ \frac{x_3}{x_3}\end{bmatrix} \bullet \begin{bmatrix}\frac{y_1}{x_1}\\ \frac{y_2}{x_2}\\ \frac{y_3}{x_3}\end{bmatrix}}{\sqrt{\left(\frac{x_1}{x_1}\right)^2 + \left(\frac{x_2}{x_2}\right)^2 + \left(\frac{x_3}{x_3}\right)^2}\sqrt{\left(\frac{y_1}{x_1}\right)^2 + \left(\frac{y_2}{x_2}\right)^2 + \left(\frac{y_3}{x_3}\right)^2}}\right)\right)}{\pi}$$

$$C_{xy} = 1 - 2\frac{\arccos\left(\dfrac{\begin{bmatrix}1\\1\\1\end{bmatrix} \bullet \begin{bmatrix}\frac{y_1}{x_1}\\ \frac{y_2}{x_2}\\ \frac{y_3}{x_3}\end{bmatrix}}{\sqrt{3\left(\left(\frac{y_1}{x_1}\right)^2 + \left(\frac{y_2}{x_2}\right)^2 + \left(\frac{y_3}{x_3}\right)^2\right)}}\right)}{\pi}$$

$$C_{xy} = 1 - 2\frac{\arccos\left(\dfrac{\frac{y_1}{x_1} + \frac{y_2}{x_2} + \frac{y_3}{x_3}}{\sqrt{3\left(\left(\frac{y_1}{x_1}\right)^2 + \left(\frac{y_2}{x_2}\right)^2 + \left(\frac{y_3}{x_3}\right)^2\right)}}\right)}{\pi}$$

3.6. STRUKTUR DATA

Struktur data yang dipergunakan dalam implementasi tugas akhir ini adalah sebagai berikut:

- Untuk merepresentasikan informasi yang diambil dari suatu file DXF, dipakai struktur data sebagai berikut:

```
T3DObjectList = class
public
  PolyLineArray : array of TFaceList;
  ExtremeValues : array [0..5] of Extended;
  constructor Create;
  function ReOrientation(var M : TMatrix3X3) : TMinMax;
  function NumberOfPolylines : Integer;
  procedure AddNewPolyline;
end;
```

PolyLineArray adalah larik yang merupakan bangun-bangun 3 dimensi. Larik ini dipakai karena dalam suatu file DXF mungkin terdiri dari lebih dari satu objek 3 dimensi. TFaceList tersebut adalah representasi sebuah bangun 3 dimensi yang akan dijelaskan dalam poin selanjutnya.

ExtremeValues adalah larik yang memuat nilai-nilai maksimum dan minimum dari sistem / file tersebut pada masing-masing sumbu koordinatnya.

- Untuk merepresentasikan masing-masing bangun 3 dimensi, dipakai struktur data sebagai berikut:

```
TFaceList = class
private
  ThePoints : array of TVect;
  TheFaces : array of array [0..2] of Integer;
public
  constructor Create;
  procedure assign_last_face(a, b, c : Integer);
  procedure assign_last_point(x, y, z : Single);
  procedure get_new_face;
  procedure get_new_point;
  function ReOrientation(var M : TMatrix3X3) : TMinMax ;
  function NumberOfFaces : Integer;
  function NumberOfPoints : Integer;
  function GetFaceInfo(FaceIndex, PointIndex : Integer) : TVect;
  function GetThePoint(FaceIndex, PointIndex : Integer) :
Integer;
end;
```

Sebuah bangun direpresentasikan sebagai sekumpulan titik-titik sudut (ThePoints) beserta sekumpulan poligon / segitiga (TheFaces) yang menghubungkan tiga titik dari seluruh titik yang ada pada larik titik-titik (ThePoints) yang melambangkan sisi-sisi bangun tersebut.

- Untuk merepresentasikan suatu titik, dilakukan secara sederhana berupa:

```
TVect = array [0..2] of Extended;
```

di mana X = TVect[0], Y = TVect[1], dan Z = TVect[2].

- Representasi voxel dari sistem objek tersebut diimplementasikan dengan struktur data:

```
VoxelArray : array [0..Sample+1, 0..Sample+1, 0..Sample+1] of
Boolean;
```

Perlu diketahui bahwa 'Sample' adalah suatu bilangan yang dipakai secara konsisten untuk mendiskritisasikan sistem-sistem benda objek 3 dimensi untuk diperbandingkan. Atas dasar itulah 'Sample' diperlakukan sebagai bilangan konstan dan bukan variabel sehingga tidak dapat didefinisikan oleh pemakai (bukan *user-defined*). Program harus dikompilasi ulang bila ingin mengubah bilangan tersebut. *Default* untuk bilangan 'Sample' adalah 50.

3.7. IMPLEMENTASI FUNGSI

Algoritma-algoritma maupun rumus-rumus yang telah disebutkan sebelumnya diimplementasikan dalam bentuk *pseudo-code* dan potongan program sebagai berikut:

3.7.1. Pemindaian File DXF

Fungsi utama :

1. Abaikan bagian HEADER dan TABLES. Cari bagian ENTITIES.
2. Cari bagian POLYLINE.
3. Perpanjang larik objek 3 dimensi sebanyak 1.
4. Ulangi langkah-langkah berikut sampai diketemukan ENDSEC:
    a. Cari awal bagian VERTEX.
    b. - Bila diketemukan token 10, catat nilai sesudahnya sebagai koordinat X.

- Bila diketemukan token 20, catat nilai sesudahnya sebagai koordinat Y.
- Bila diketemukan token 30, catat nilai sesudahnya sebagai koordinat Z.
- Bila diketemukan token 70, lihat nilai sesudahnya. Bila nilai sesudah token 70 adalah 192, maka bagian ini berisi informasi tentang posisi suatu titik sudut 3 dimensi; namun bila bernilai 128, maka bagian ini berisi informasi tentang tiga buah indeks titik sudut suatu sisi (segitiga).
- Bila diketemukan token 71, catat nilai sesudahnya sebagai indeks titik pertama suatu sisi.
- Bila diketemukan token 72, catat nilai sesudahnya sebagai indeks titik kedua suatu sisi.
- Bila diketemukan token 73, catat nilai sesudahnya sebagai indeks titik ketiga suatu sisi.

c. Cari awal bagian VERTEX lainnya; bila ketemu, ulangi bagian b.

5. Cari bagian POLYLINE lainnya; bila ada, ulangi mulai nomor 3.

3.7.2. Diskritisasi Segitiga.

Untuk setiap segitiga yang ada, lakukan:

1. Pembacaan.
2. Proyeksikan segitiga tersebut menjadi ke bidang XOY, XOZ, dan YOZ
3. Temukan nilai minimum dan maksimum untuk ketiga titik dalam segitiga tersebut.
4. Tentukan *Bounding Box* untuk segitiga tersebut (relatif terhadap keseluruhan sistem 3 dimensi yang sedang divoxelisasi) dalam bentuk larik 3 dimensi.
5. Buat bitmap-bitmap yang sesuai untuk ketiga hasil proyeksi.
6. Petakan bitmap-bitmap tersebut ke dalam *Bounding Box*.

7. Perluas bitmap-bitmap tersebut ke dimensi yang ketiga (contoh: segitiga hasil proyeksi XOY diperluas ke sepanjang sumbu Z).
8. Untuk setiap sel dalam Bounding Box yang merupakan perpotongan dari perluasan ketiga bitmap, tandai sel tersebut ke dalam larik utama (VoxelArray).

3.7.3. Penentuan Bagian Luar-Dalam Objek 3 Dimensi.

Bagian luar-dalam objek 3 dimensi ditentukan dengan cara:

1. Lakukan proses *flood-filling* pada ketiga arah irisan terhadap VoxelArray (irisan pertama searah dengan bidang XOY, irisan kedua searah dengan bidang XOY, irisan ketiga searah dengan bidang YOZ).
2. Tandai setiap sel dalam VoxelArray yang tidak berhasil di-*fill* oleh proses di atas sebagai 'bagian dalam objek'.

Proses *flood-filling* dilakukan dengan algoritma yang digambarkan pada gambar 2.

3.7.4. Penghitungan Ciri-ciri Objek.

Ciri-ciri objek dihitung berdasarkan nilai-nilai n (momen-momen orde kedua yang telah dinormalisasi), yang dapat dicari dengan cara sebagai berikut (dalam bentuk potongan program Pascal):

```
n200 := 0;
n020 := 0;
n002 := 0;
n110 := 0;
n101 := 0;
n011 := 0;
for i := 1 to Sample do
  for j := 1 to Sample do
    for k := 1 to Sample do
      if VoxelArray[i,j,k] then begin
        n200 := n200 + ((I-XCenter)*(I-XCenter) + 1/12);
        n020 := n020 + ((J-YCenter)*(J-YCenter) + 1/12);
        n002 := n002 + ((K-ZCenter)*(K-ZCenter) + 1/12);
        n110 := n110 + (I-XCenter)*(J-YCenter);
```

```
        n011 := n011 + (J-YCenter)*(K-ZCenter);
        n101 := n101 + (I-XCenter)*(K-ZCenter);
      end;
n200 := n200 / Power(m000,5/3);
n020 := n020 / Power(m000,5/3);
n002 := n002 / Power(m000,5/3);
n110 := n110 / Power(m000,5/3);
n011 := n011 / Power(m000,5/3);
n101 := n101 / Power(m000,5/3);
```

Sample adalah besarnya sampling yang dilakukan terhadap objek (juga berarti besarnya nilai maksimum pada masing-masing dimensi VoxelArray). Power(a,b) adalah fungsi pemangkatan yang memberikan pengembalian nilai berupa a pangkat b. m000 adalah banyaknya sel pada VoxelArray yang bernilai True (solid), dan dapat dihitung dengan cara:

```
m000 := 0;
for i := 1 to Sample do
  for j := 1 to Sample do
    for k := 1 to Sample do
      if VoxelArray[i,j,k] then m000 := m000+1;
```

(XCenter, YCenter, ZCenter) adalah koordinat titik berat benda. Ketiga variabel ini dapat dihitung dengan cara :

```
m000 := 0;
m100 := 0;
m010 := 0;
m001 := 0;
for i := 1 to Sample do
  for j := 1 to Sample do
    for k := 1 to Sample do
      if VoxelArray[i,j,k] then begin
        m000 := m000 + 1;
        m100 := m100 + i;
        m010 := m010 + j;
        m001 := m001 + k;
      end;
XCenter := m100 / m000;
YCenter := m010 / m000;
ZCenter := m001 / m000;
```

3.7.5. Pembandingan Objek-objek.

Kemiripan / similaritas objek-objek diimplementasikan dengan rumus:

```
Similaritas := 1 - 2*ArcCos((FNow1/F1 + FNow2/F2 + FNow3/F3) /
                            Sqrt(3*(Sqr(FNow1/F1)+Sqr(FNow2/F2)+
                                    Sqr(FNow3/F3)
                                    )
                                 )
                            ) / PI;
```

ArcCos adalah invers fungsi Cosinus. Sqrt adalah fungsi akar pangkat 2 dari suatu bilangan. Sqr adalah fungsi pangkat 2 dari suatu bilangan. FNow1, FNow2, dan FNow3 adalah ciri-ciri pertama, kedua, dan ketiga dari file *library*. F1, F2, dan F3 adalah ciri-ciri pertama, kedua, dan ketiga dari file CAD yang sedang menjadi fokus untuk diperbandingkan.

BAB IV

UJI DAN EVALUASI PERANGKAT LUNAK

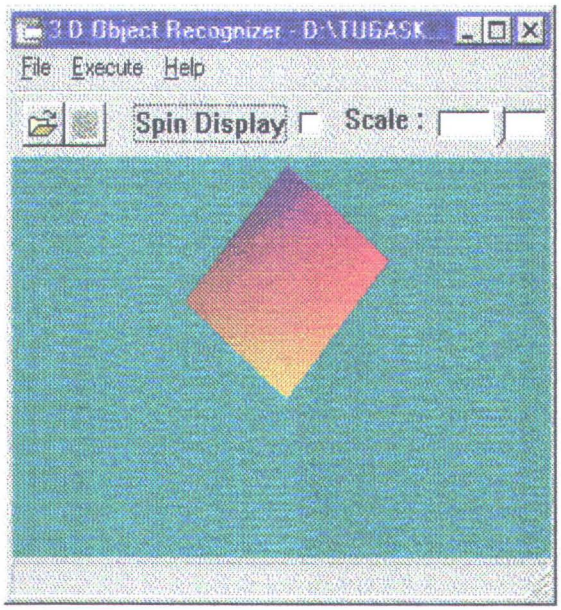
Pembuatan perangkat lunak ini dilakukan dengan menggunakan compiler Borland Delphi 5.0. Pengujian perangkat lunak ini dilakukan dengan menggunakan komputer dengan prosesor Intel Pentium II – 350 MHz, memori sebesar 32 MB, dan VGA card Asus 3D Explorer dengan memori 4 MB.

Uji dan evaluasi dilakukan terhadap sejumlah file DXF yang berisi berbagai variasi dari objek -objek 3 dimensi. Yang disajikan dalam tugas akhir ini hanyalah tiga macam objek berupa kotak (Box1 sampai dengan Box6), bola (Sphere1 dan Sphere2), serta Triceratops (Rhino1 sampai dengan Rhino6). Seluruh hasil pengujian ditampilkan terlebih dahulu, sementara evaluasinya dijelaskan kemudian.

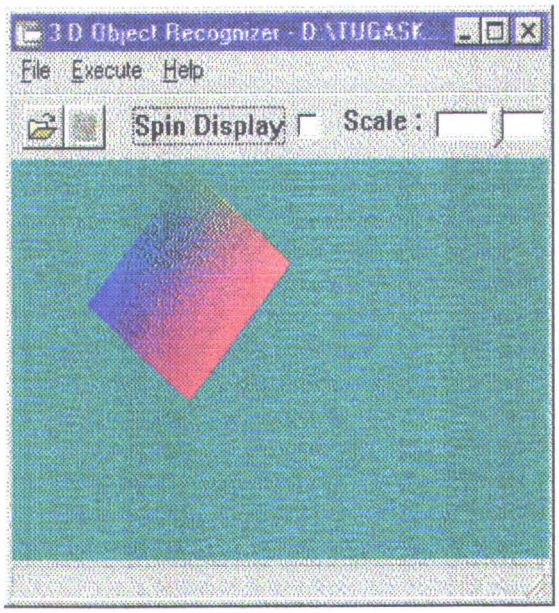
4.1. UJI PERANGKAT LUNAK

Pengujian terhadap perangkat lunak yang telah dibuat dilakukan dengan cara memperbandingkan file-file DXF yang berisi objek – objek 3 dimensi. Pertama kali, untuk menguji metode yang dipakai (pembandingan momen-momen invarian), program diuji dengan file-file DXF yang berisi objek-objek yang sama, namun dilakukan translasi, rotasi, maupun penyekalaan. Setelah itu barulah pembandingan antara benda-benda yang berbeda wujud fisiknya diperbandingkan.

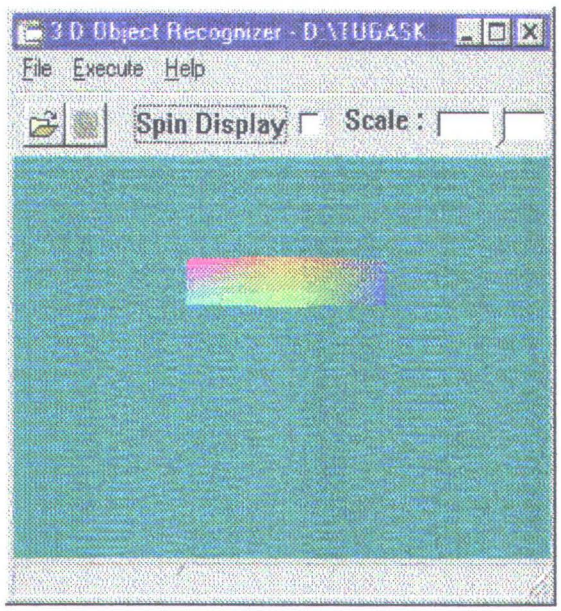
Berikut ini adalah file – file yang masing – masing berisi sebuah objek 3 dimensi berupa kotak.



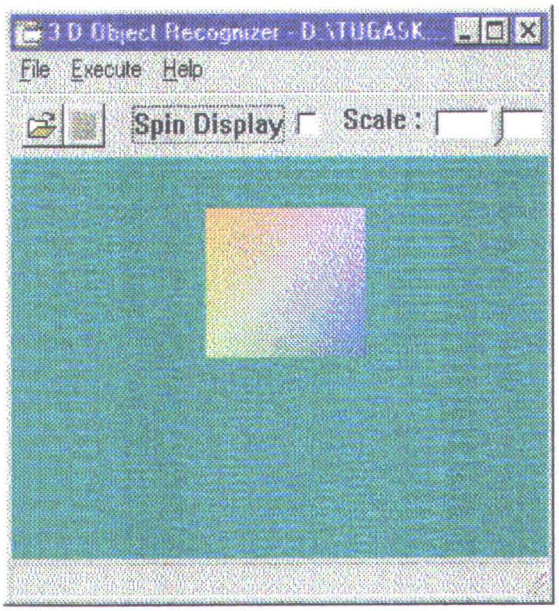
Gambar 4.1. Box1



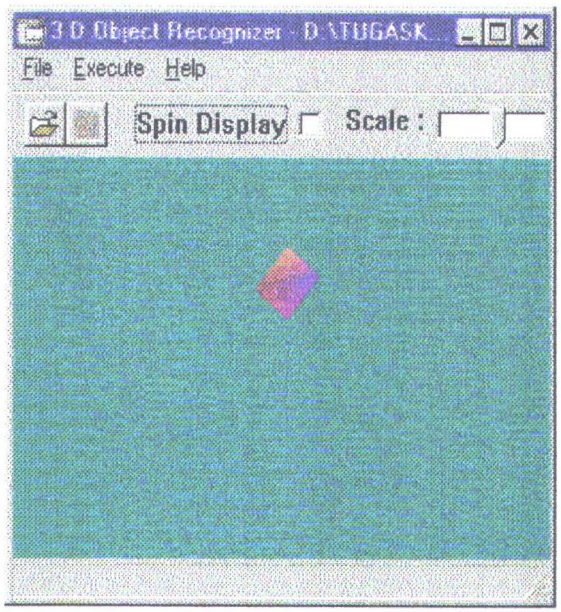
Gambar 4.2. Box2



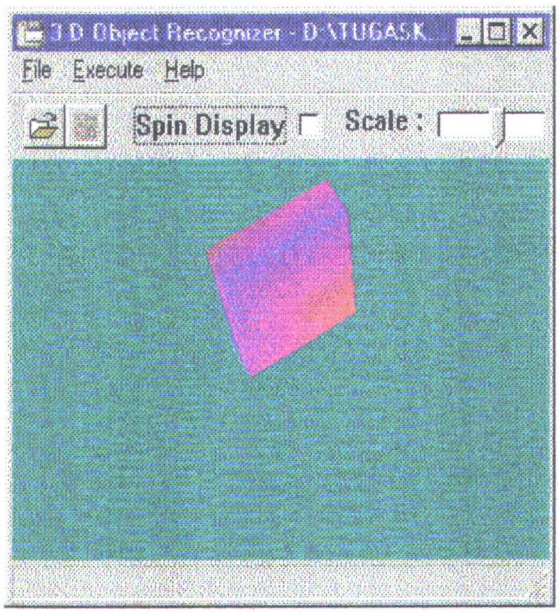
Gambar 4.3. Box3



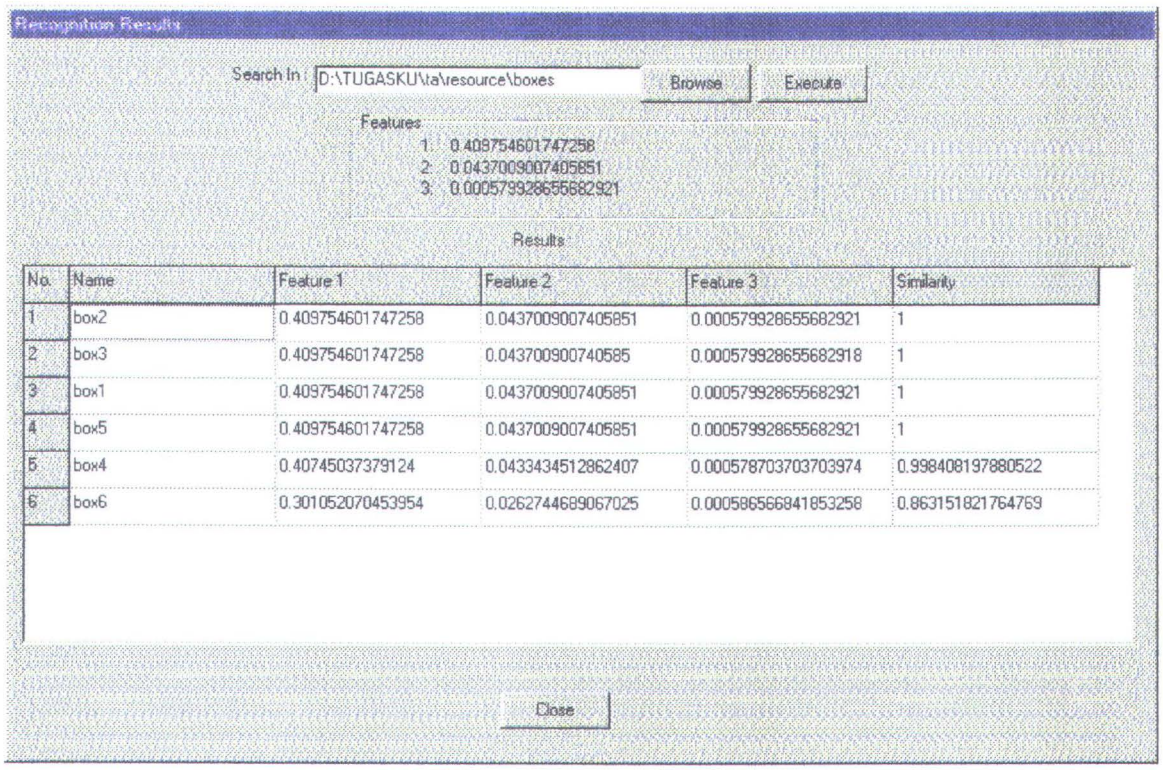
Gambar 4.4. Box4



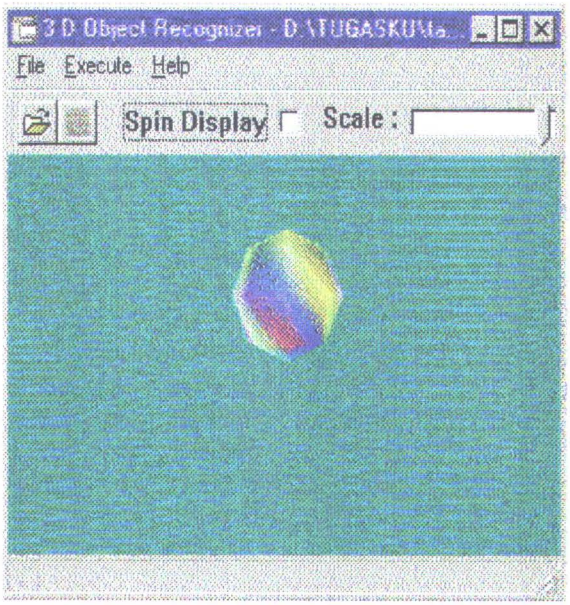
Gambar 4.5. Box5



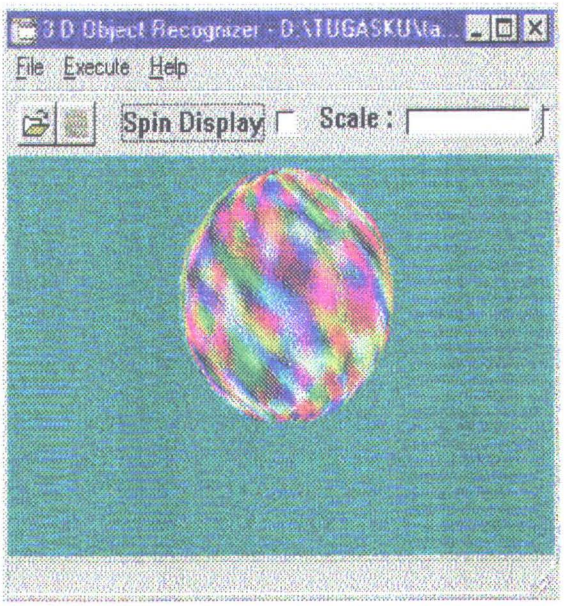
Gambar 4.6. Box6



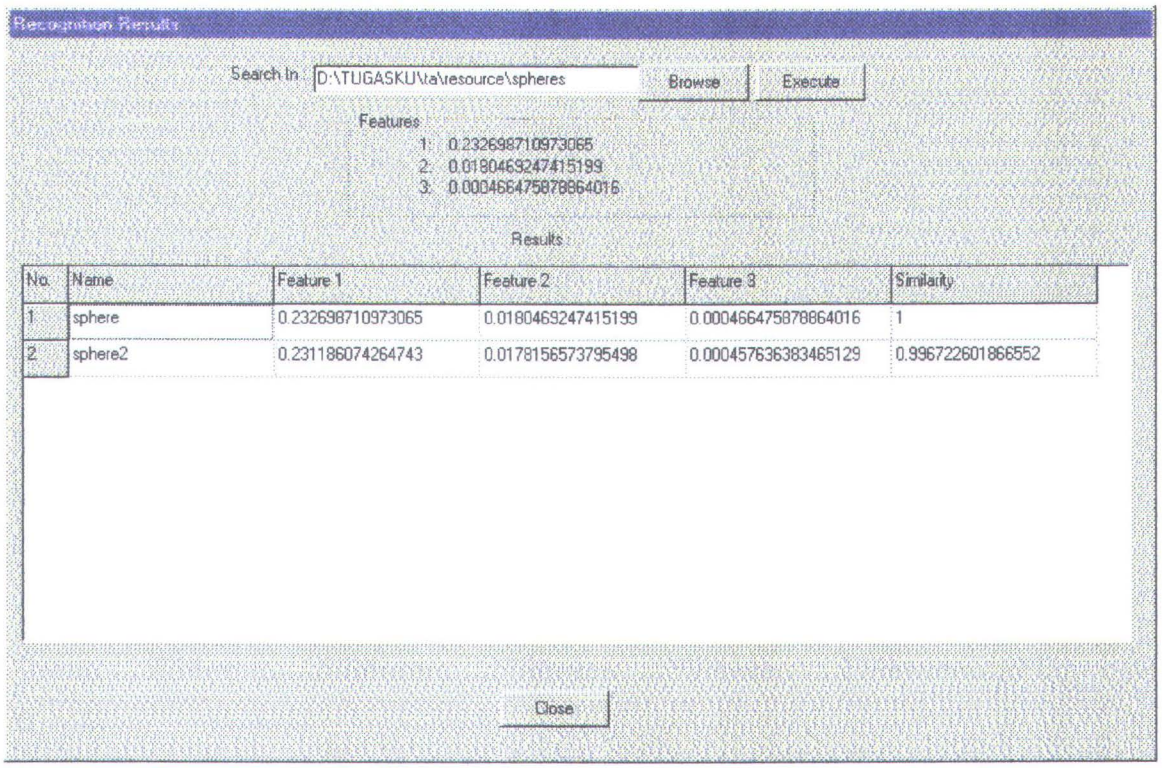
Gambar 4.7. Hasil pembandingan antara Box1 dengan Box1 sampai dengan Box6



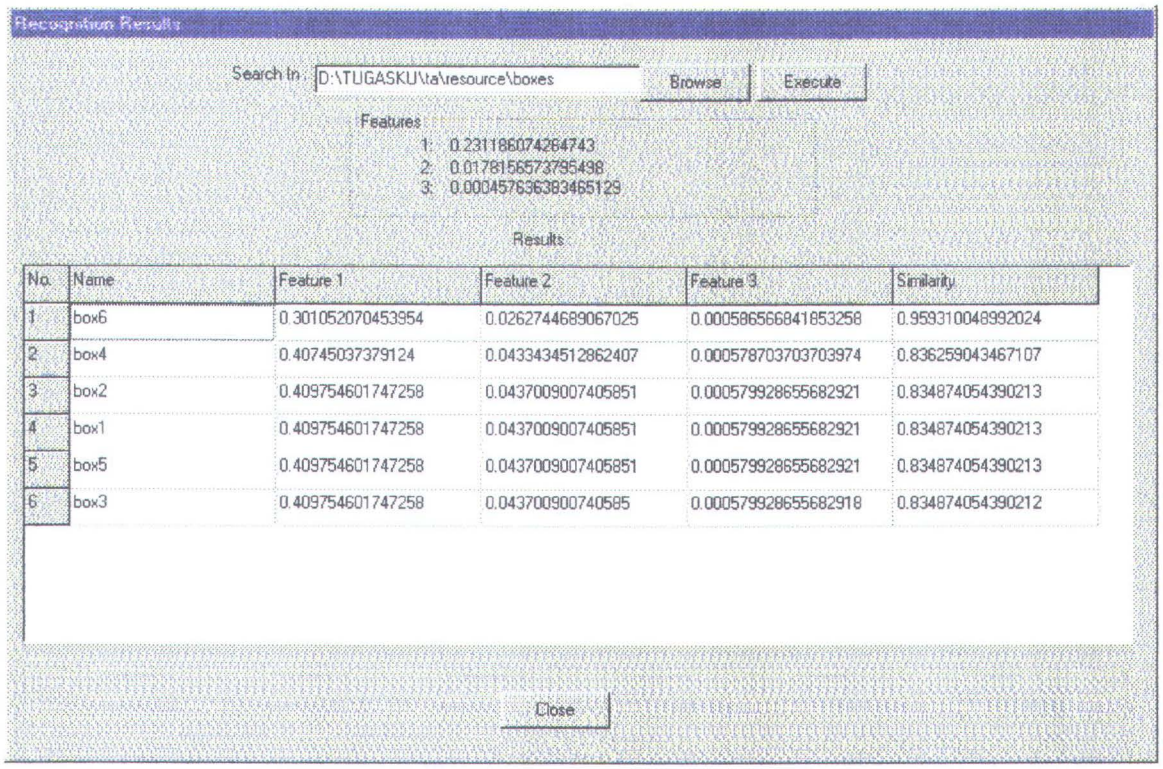
Gambar 4.8. Sphere1



Gambar 4.9. Sphere2

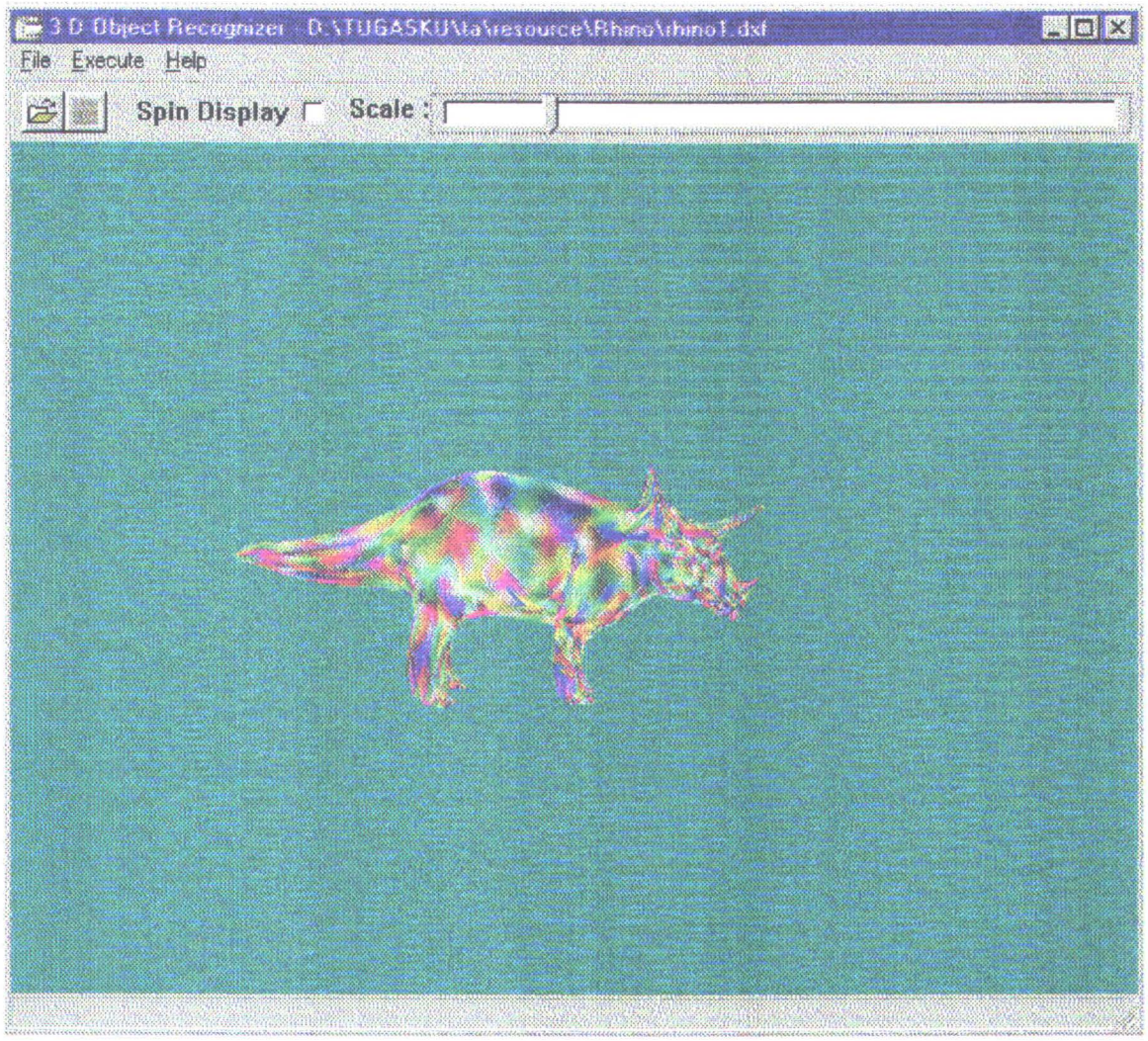


Gambar 4.10. Hasil pembandingan antara Sphere1 dengan Sphere1 dan Sphere2

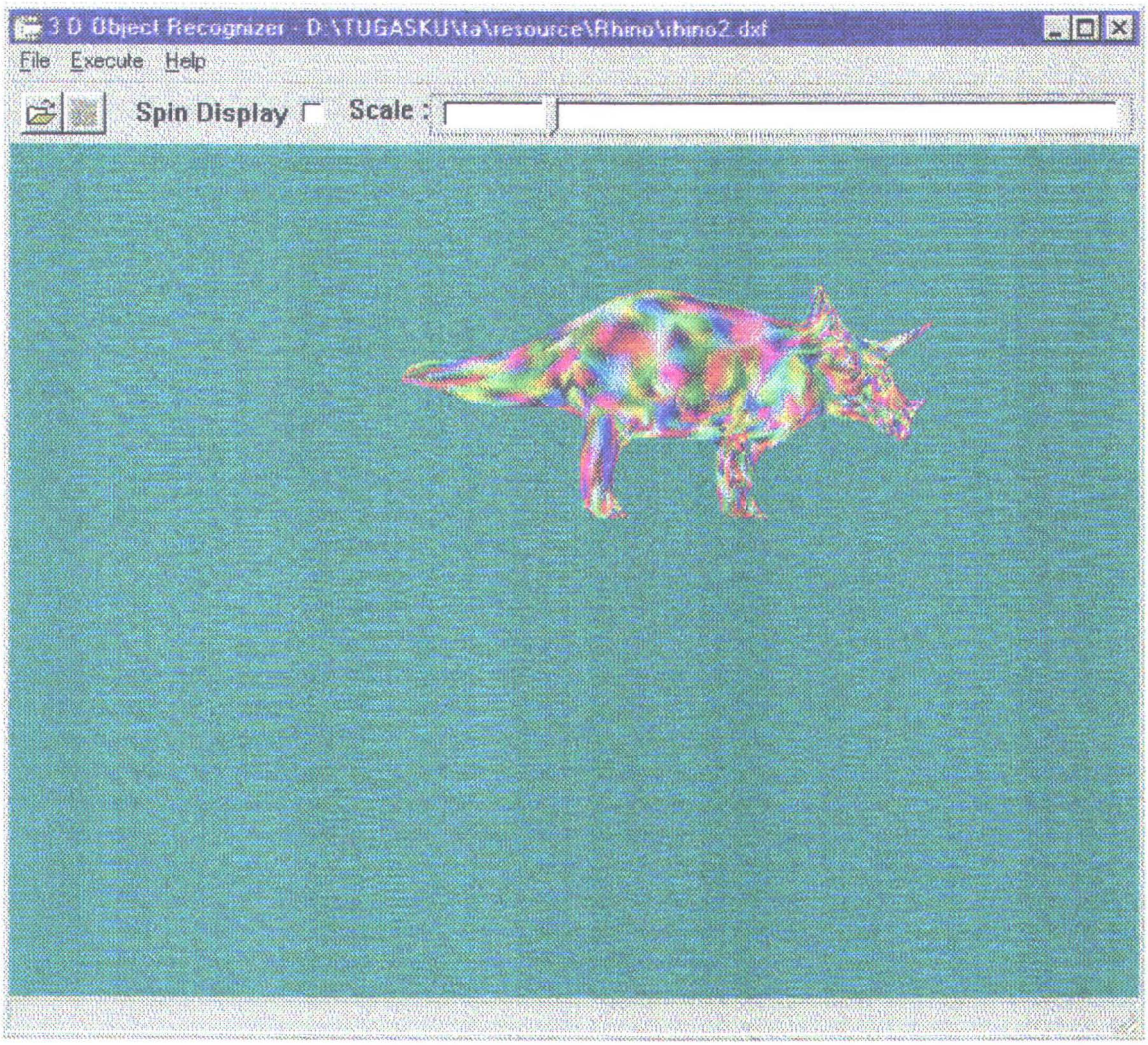


Gambar 4.11. Hasil pembandingan antara Sphere2 dengan Box1 sampai Box6

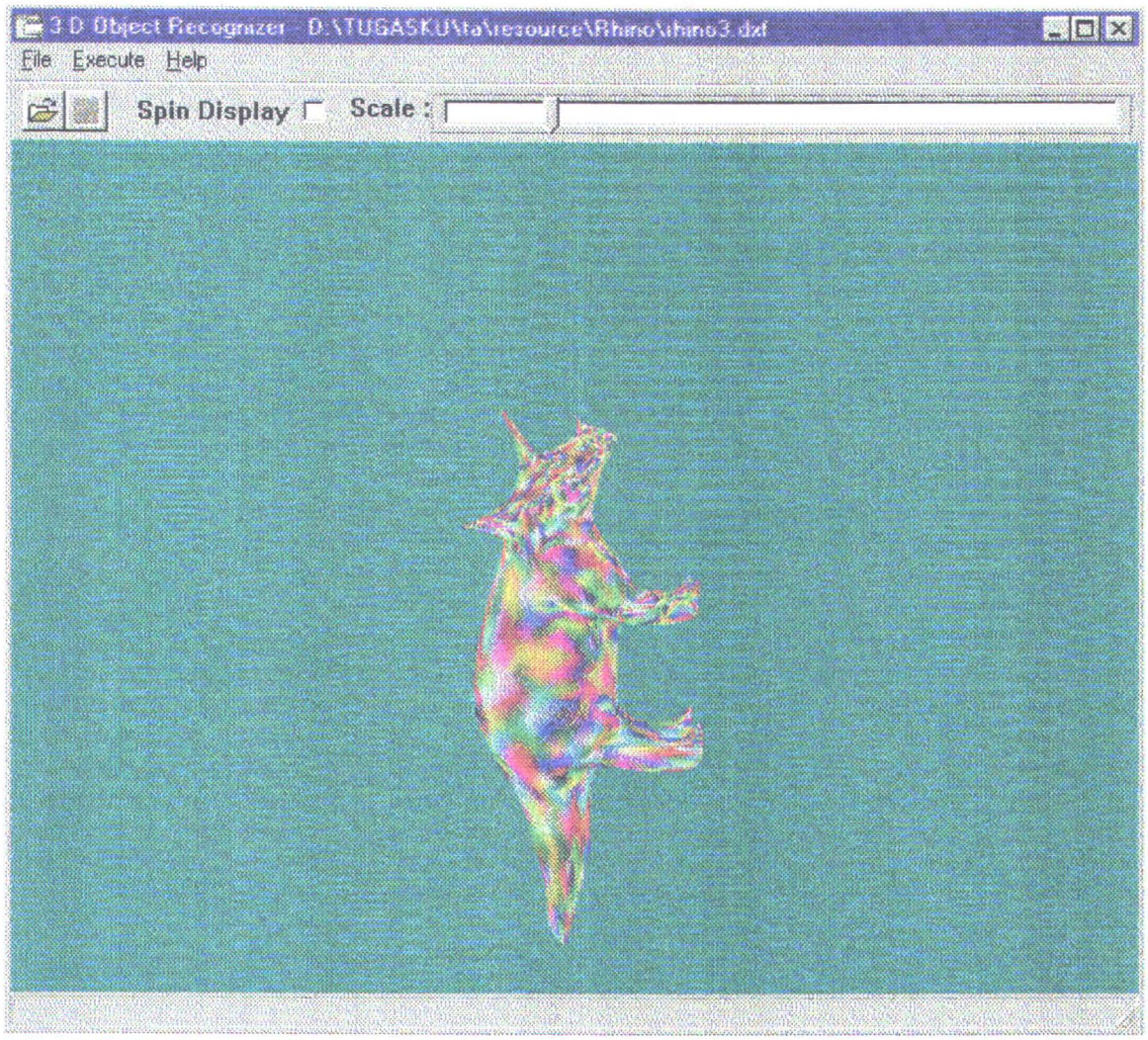
Untuk pengujian program terhadap benda yang memiliki bentuk kompleks, file yang berisi bangun berbentuk Triceratops dipakai. File yang dipakai sebagai dasar adalah Rhino1. File-file Rhino2 sampai dengan Rhino6 adalah variasi dari file Rhino1. Berikut ini adalah file-file tersebut:



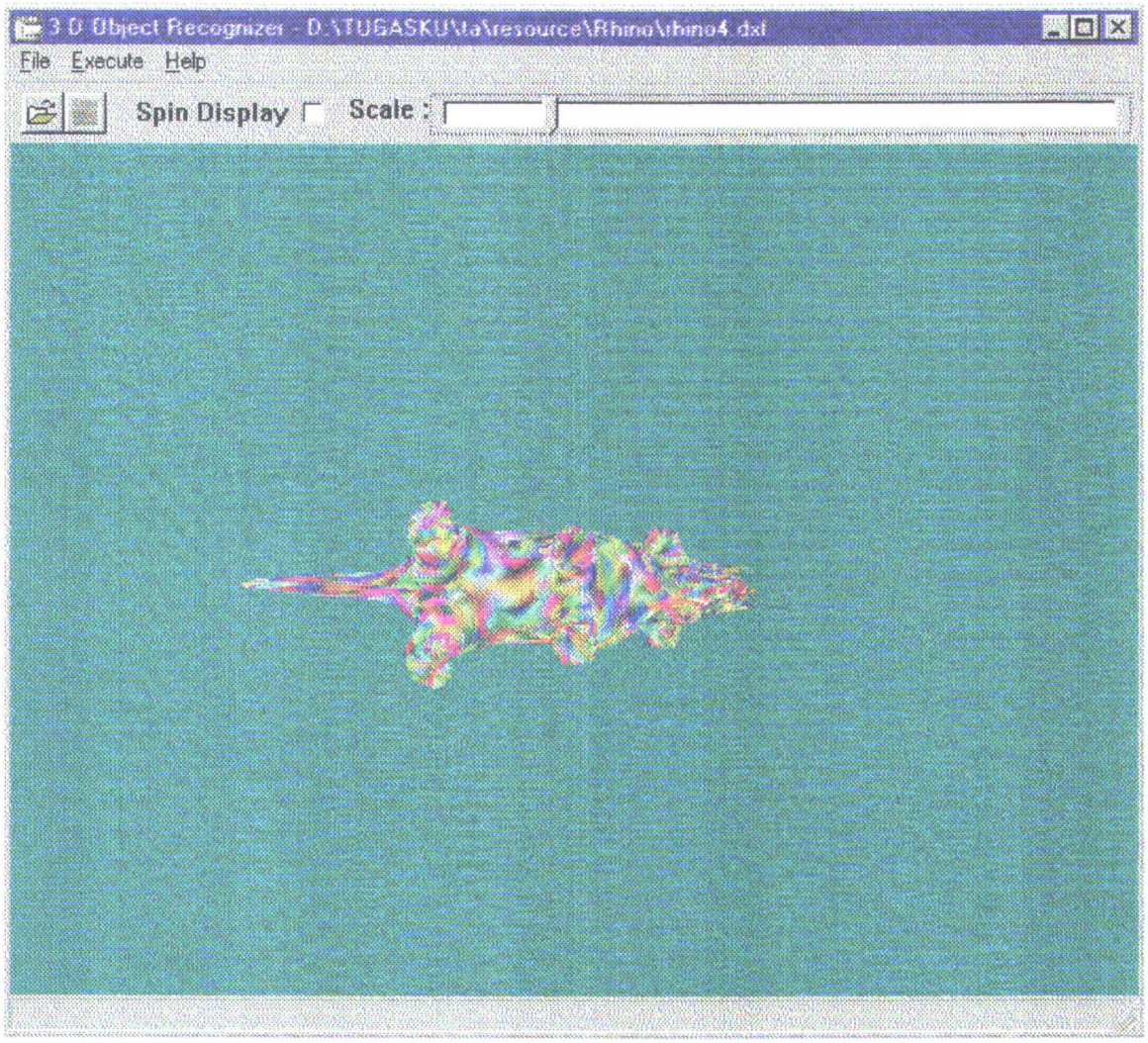
Gambar 4.12. Rhino1



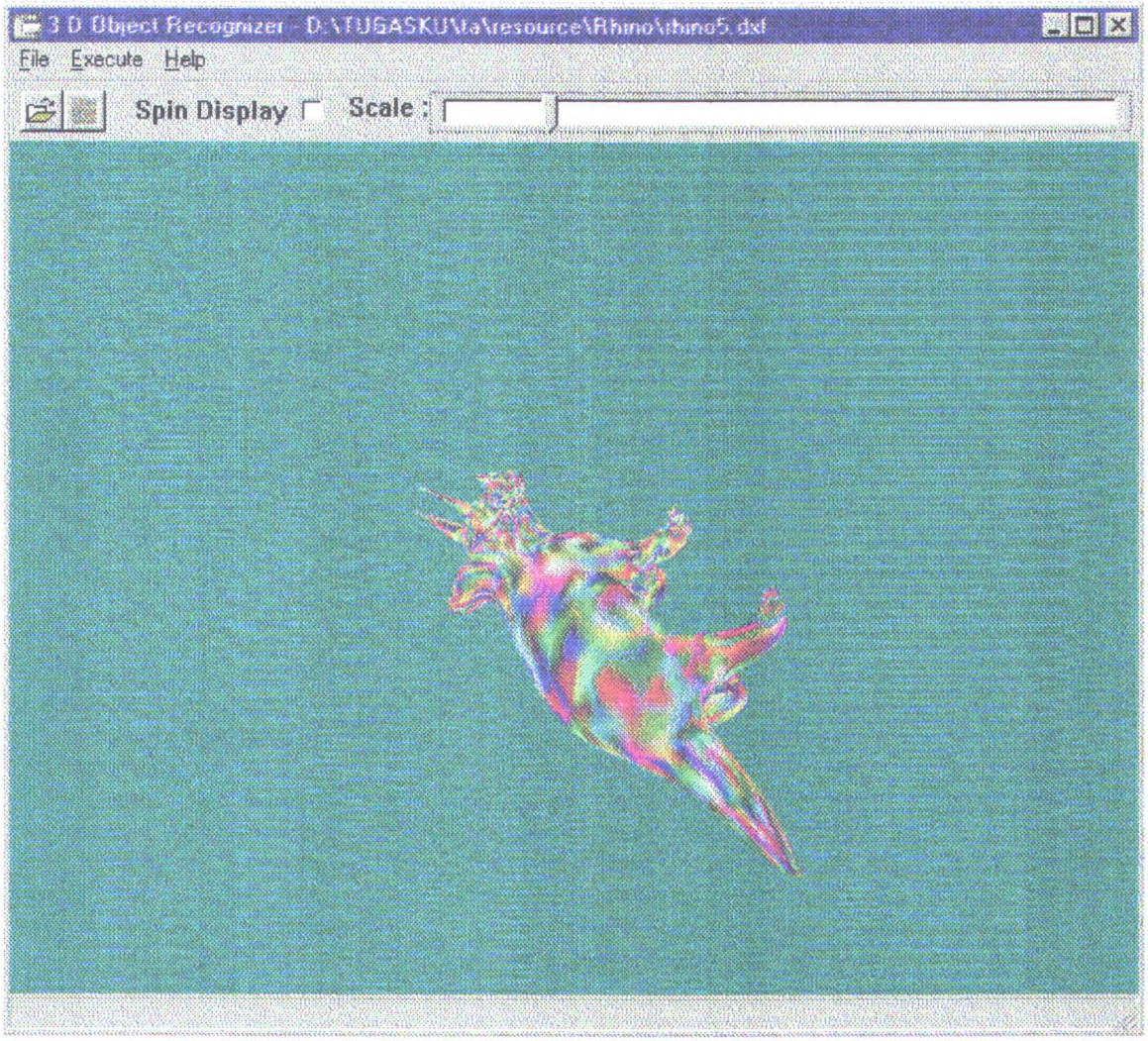
Gambar 4.13. Rhino2



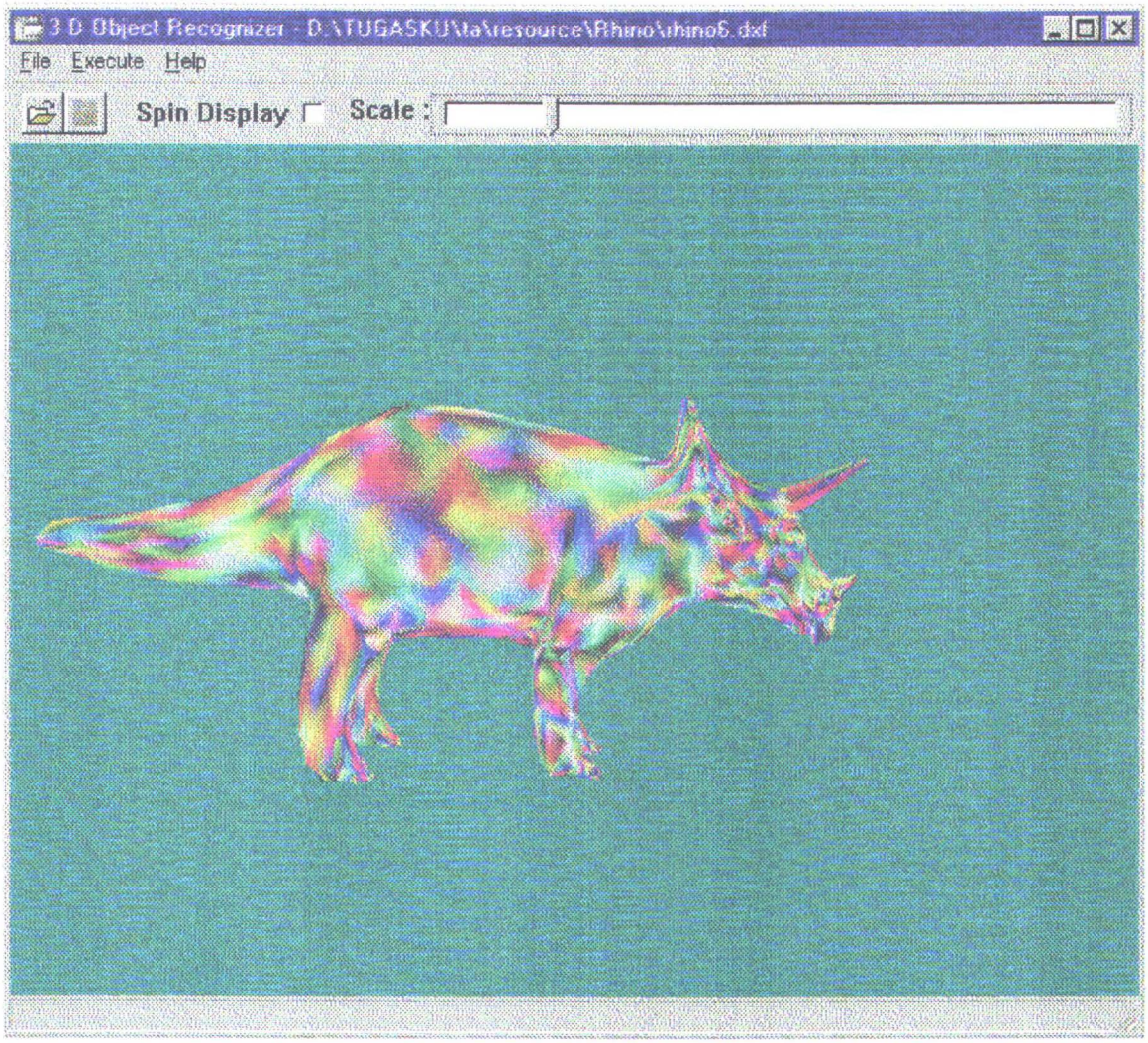
Gambar 4.14. Rhino3



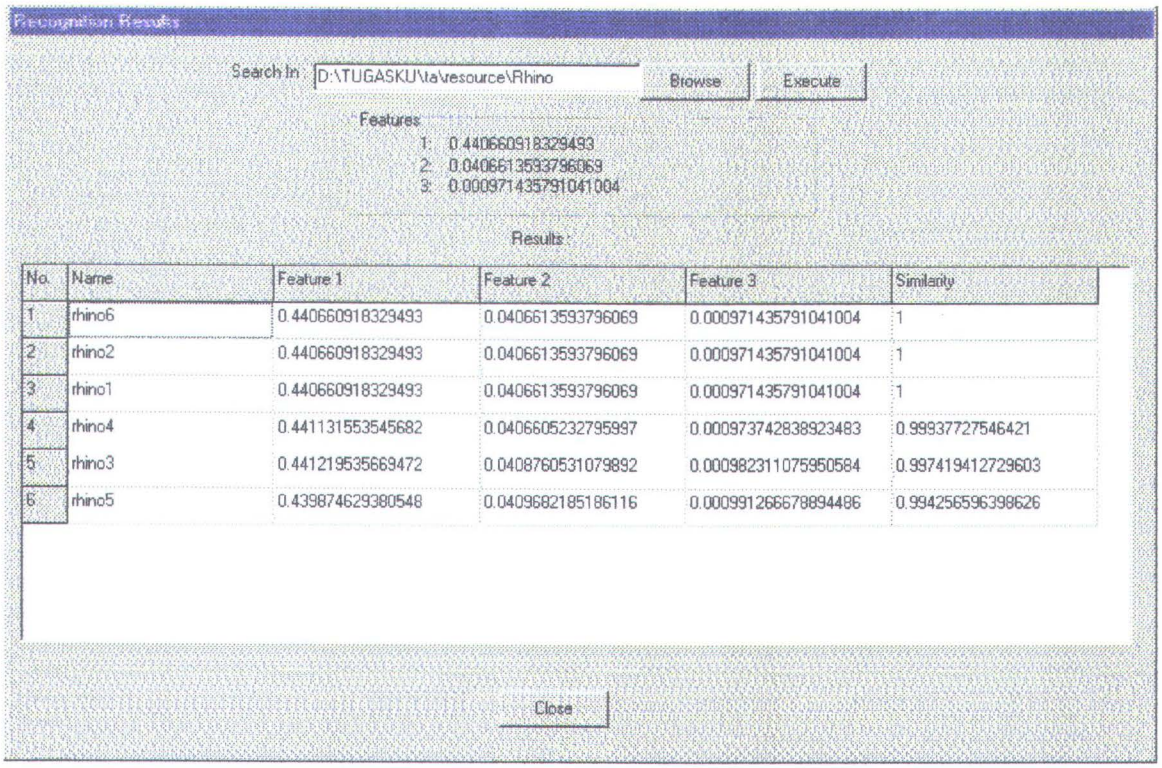
Gambar 4.15. Rhino4



Gambar 4.16. Rhino5



Gambar 4.17. Rhino6



Gambar 4.18. Hasil pembandingan antara Rhino1 dengan Rhino1 sampai Rhino6

4.2. EVALUASI

Hasil pengujian program menunjukkan bahwa terdapat kesalahan (*error*) yang cukup signifikan terhadap pengenalan antara Box1 terhadap Box6. Hal ini disebabkan karena kurang sempurnanya modul voxelisasi yang dipergunakan dalam pembuatan tugas akhir ini.

Hal yang serupa tidak terlihat signifikan untuk bangun-bangun yang kompleks. Ini terbukti dari pengujian bangun berbentuk Triceratops yang juga telah ditampilkan sebelumnya. Hal ini disebabkan karena pada bangun Triceratops tersebut ukuran poligon-poligon yang menjadi sisi-sisi bangun tersebut relatif sangat kecil terhadap keseluruhan bangun, sehingga kesalahan karena voxelisasi poligon dapat diabaikan.

Kesalahan yang disebabkan karena voxelisasi poligon dapat diabaikan seluruhnya bila tidak terdapat poligon yang memiliki *range* koordinat melebihi lebar voxel. Yang dimaksud range koordinat di sini adalah selisih nilai X maksimum dengan nilai X minimum, selisih nilai Y maksimum dengan nilai Y minimum, maupun selisih nilai Z maksimum dengan nilai Z minimum (dalam konteks ini adalah untuk masing-masing poligon). Lebar voxel dihitung dari maksimum *range* koordinat benda secara keseluruhan dibagi dengan besarnya sampling yang dilakukan secara konsisten atas benda-benda yang diperbandingkan (nilai *default*-nya adalah 50).

Secara keseluruhan, hasil perangkat lunak menunjukkan bahwa metode pembandingan invarian RST yang dipakai berhasil mengenali objek-objek 3 dimensi berbasis vektor.

BAB V

PENUTUP

Uraian yang terdapat di dalam bab-bab sebelumnya beserta pengujian perangkat lunak dipakai untuk membuat beberapa kesimpulan dan saran yang dapat meningkatkan kinerja perangkat lunak di waktu yang akan datang.

5.1. KESIMPULAN

Beberapa kesimpulan yang dapat diambil dari tugas akhir ini sebagai berikut:

- Pembandingan invarian – invarian RST (Rotation, Scaling, Translation atau rotasi, penyekalaan, translasi) dapat dipergunakan untuk mengenali objek-objek 3 dimensi.
- Dengan mempergunakan pembandingan invarian-invarian RST tersebut, proses pengenalan dapat dilakukan secara cukup akurat (dengan nilai ketepatan antara 98% sampai dengan 100%, dengan asumsi besarnya setiap poligon tidak melebihi lebar voxel) dan efisien (linier terhadap banyaknya benda yang disimpan dengan *library*).

5.2. SARAN

Bagi para pembaca yang tertarik untuk mengembangkan lebih lanjut tugas akhir ini, ataupun siapa saja yang berminat untuk menerapkan tugas akhir ini dalam aplikasi yang nyata, penulis menyarankan:

- Untuk menyempurnakan atau memakai metode lain dalam modul voxelisasi, sehingga validitas diskritisasi poligon-poligon dapat ditingkatkan.
- Mencoba kombinasi antara metode yang telah penulis implementasikan dengan metode-metode lain yang mungkin lebih mutakhir.

DAFTAR PUSTAKA

- Ayres, Frank Jr. *Matriks (Versi SI / Metrik)*. Cetakan kedua. Penerbit Erlangga. Jakarta. 1989.
- Casio. *FX-501P/FX-502P Program Library*. Casio Computer Co, Ltd. Tokyo.
- Casio. *Personal Computer FX-850P/FX-880P Owner's Manual*. Casio Computer Co, Ltd. Tokyo.
- Cohen, Kurt D. *Feature Extraction and Pattern Analysis of Three Dimensional Objects*, Paper at the Thayer School of Engineering, Darmouth College.
- Conte, Samuel D. dan Carl de Boor. *Dasar-dasar Analisis Numerik*. Edisi Ketiga. Penerbit Erlangga. Jakarta. 1993.
- Gerber, Harvey. *Elementary Linear Algebra*. Brooks/Cole Publishing Company. Pacific Grove, California. 1990.
- Goldberg, Jack L. *Matrix Theory with Applications*. McGraw-Hill, Inc. New York. 1991.
- Gonzalez-Ochoa, Carlos, Scott McCammon, dan Jörg Peters. *Computing Moments of Objects Enclosed by Piecewise Polynomial Surfaces* (dari ACM Transactions on Graphics Volume 17 Number 3 July 1998).
- Grossman, Stanley I. *Elementary Linear Algebra*. Fourth Edition. Saunders College Publishing. Fort Worth. 1991.
- Harrington, Steven. *Computer Graphics : A Programming Approach*. Second Edition. McGraw-Hill Book Company. Singapore. 1987.

- Horn, Franz E. *Elementary Matrix Algebra.* Third Printing. The Macmillan Company. New York. 1960.
- Horn, Roger A. dan Charles R. Johnson. *Matrix Analysis.* Cambridge University Press. Cambridge. 1985.
- Kreyszig, Erwin. *Advanced Engineering Mathematics.* Sixth Edition. John Wiley & Sons. New York. 1988.
- Mittal. P.K. *Introduction to Matrices (For Engineering and Physics Students).* Har-Anand Publications. New Delhi. 1994.
- Pipes, Lewis A. *Matrix Methods for Engineering.* Prentice-Hall, Inc. Englewood Cliffs, New Jersey. 1963.
- Schalkoff, Robert J. *Pattern Recognition : Statistical, Structural, and Neural Approaches.* John Wiley & Sons, Inc. Singapore. 1992.
- Stroud. K. A. *Matematika untuk Teknik.* Edisi Ketiga. Penerbit Erlangga. Jakarta. 1989
- Taylor, Walter F. *The Geometry of Computer Graphics.* Wadsworth & Brooks / Cole Advanced Books & Software. Pacific Grove, California. 1992.

LAMPIRAN

1. MASALAH NILAI EIGEN

Nilai-nilai Eigen dicari dengan cara mencari nilai variabel λ pada persamaan sebagai berikut :

$$\begin{vmatrix} \mu_{200}-\lambda & \mu_{110} & \mu_{101} \\ \mu_{110} & \mu_{020}-\lambda & \mu_{011} \\ \mu_{101} & \mu_{011} & \mu_{002}-\lambda \end{vmatrix} = 0$$

atau secara polinomial berupa

$$-\lambda^3+\lambda^2(\mu_{200}+\mu_{020}+\mu_{002})-\lambda(\mu_{002}\mu_{020}+\mu_{002}\mu_{200}+\mu_{020}\mu_{200}-\mu_{011}^2-\mu_{101}^2-\mu_{110}^2)$$

$$+(\mu_{002}\mu_{020}\mu_{200}+2\mu_{011}\mu_{101}\mu_{110}-\mu_{101}^2\mu_{020}-\mu_{011}^2\mu_{200}-\mu_{110}^2\mu_{002})=0$$

Tentu saja pencarian nilai λ dalam hal ini memerlukan modul tambahan yang dapat menurunkan akar-akar persamaan polinomial kubik tersebut.

2. AKAR-AKAR POLINOMIAL KUBIK[2]

Secara umum, persamaan polinomial kubik dapat ditulis sebagai berikut:

$$ax^3+bx^2+cx-d=0.$$

Persamaan tersebut dapat ditransformasikan sebagai berikut

$$y^3+3py+q=0$$

dengan menempatkan

---

[2] CASIO. *FX-501P/FX-502P Program Library*. Casio Computer Co., Ltd. Tokyo. Hal 13.

$$x = y - \frac{b}{3a},$$

$$p = \frac{c}{3a} - \frac{b^2}{9a^2},$$

$$q = \frac{2b^3}{27a^3} - \frac{bc}{3a^2} + \frac{d}{a}.$$

Kemudian dapat dicari nilai-nilai berikut ini:

$$A = \sqrt[3]{\frac{q + \sqrt{C}}{2}},$$

$$B = \sqrt[3]{\frac{q - \sqrt{C}}{2}},$$

$$C = q^2 + 4p^3$$

Selanjutnya, terdapat empat kondisi untuk menentukan akar-akar y. Kondisi - kondisi ini ditentukan dengan menggunakan variabel C dan p.

1. Jika $C > 0$, terdapat satu akar real (α), dan dua akar-akar imajiner (β,γ) :

$$\alpha = -(A + B),$$

$$\beta = \frac{A + B}{2} + \frac{\sqrt{3}}{2}(A - B)i,$$

$$\gamma = \frac{A + B}{2} - \frac{\sqrt{3}}{2}(A - B)i$$

2. Jika $C = 0$, p = 0, terdapat satu akar real:

$$\alpha = -(A + B)$$

3. Jika $C = 0$, p $\neq$0, terdapat dua akar real:

CASIO. *Personal Computer Scientific Library 116 FX-850P/FX880P*. Casio Computer Co., Ltd. Tokyo. Hal 188.

$$\alpha = -(A+B),$$

$$\beta = \frac{A+B}{2}$$

4. Jika $C < 0$, terdapat tiga akar real:

$$\alpha = -2\sqrt{-p}\cos\theta,$$

$$\beta = -2\sqrt{-p}\cos(\theta + 120°),$$

$$\gamma = -2\sqrt{-p}\cos(\theta + 240°)$$

di mana

$$\theta = \frac{1}{3}\cos^{-1}\frac{q}{2\sqrt{-p^3}}$$

Satu hal lagi yang perlu diperhatikan dalam implementasi akhir rumus - rumus tersebut adalah implementasi akar pangkat 3. *Library* standar yang diberikan oleh Delphi tidak memungkinkan untuk menghitung akar pangkat 3 apabila bilangan yang dicari akar pangkat 3-nya tersebut adalah bilangan negatif. Untuk itu, khusus dalam menghitung akar pangkat 3 ini fungsi POWER diubah menjadi :

```
function MyPower(Base, Exponent: Extended): Extended;
begin
  if Exponent = 0.0 then
    Result := 1.0                { n**0 = 1 }
  else if (Base = 0.0) and (Exponent > 0.0) then
    Result := 0.0                { 0**n = 0, n > 0 }
  else if (Frac(Exponent)=0.0) and (Abs(Exponent) <= MaxInt) then
    Result := IntPower(Base, Integer(Trunc(Exponent)))
  else if Base <= 0 then
    Result := -(Exp(Exponent * Ln(-Base)))
  else
    Result := Exp(Exponent * Ln(Base))
end;
```

3. VEKTOR-VEKTOR EIGEN

Nilai-nilai eigen dari matriks momen orde kedua itu telah diketemukan setelah akar-akar dari persamaan polinomial kubik itu dikalkulasi. Langkah selanjutnya adalah menetapkan vektor – vektor eigen.

Pertama kali, yang harus dilakukan adalah mengisikan nilai eigen yang didapat dari prosedur sebelumnya sehingga menghasilkan matriks – matriks

$$M_n = \begin{pmatrix} \mu_{xx} - \lambda_n & \mu_{xy} & \mu_{xz} \\ \mu_{xy} & \mu_{yy} - \lambda_n & \mu_{yz} \\ \mu_{xz} & \mu_{yz} & \mu_{zz} - \lambda_n \end{pmatrix}$$

di mana n mungkin bernilai 1, 1 dan 2, atau 1 sampai 3 tergantung pada banyaknya nilai-nilai eigen yang dikalkulasi. Matriks – matriks inilah yang akan dipakai untuk mencari vektor-vektor eigen.

Sistem persamaan simultan yang dipakai untuk menemukan vektor-vektor eigen adalah:

$$\begin{pmatrix} \mu_{xx} - \lambda_n & \mu_{xy} & \mu_{xz} \\ \mu_{xy} & \mu_{yy} - \lambda_n & \mu_{yz} \\ \mu_{xz} & \mu_{yz} & \mu_{zz} - \lambda_n \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Dengan memisalkan $a = \mu_{xx} - \lambda_n$, $b = \mu_{xy}$, $c = \mu_{xz}$, $e = \mu_{yy} - \lambda_n$, $f = \mu_{yz}$, $g = \mu_{zz} - \lambda_n$, persamaan di atas dapat ditulis ulang dalam bentuk persamaan simultan sebagai berikut:

$$ax + by + cz = 0$$

$$bx + ey + fz = 0$$

$$cx + fy + gz = 0$$

Sistem ini mempunyai solusi non-trivial (solusi trivial adalah x = 0, y = 0, z = 0) apabila rank matriks ini berjumlah kurang dari 3, atau dengan kata lain determinan matriks bernilai nol.

Karena determinan matriks bernilai nol, maka tidak mungkin mendapatkan hasil yang diinginkan dengan melakukan eliminasi Gauss-Jordan seperti yang biasa dilakukan dalam persamaan simultan. Sebagai gantinya, yang perlu dilakukan adalah eliminasi Gauss tanpa substitusi kembali.

4. ELIMINASI GAUSS DAN KALKULASI VEKTOR-VEKTOR EIGEN

Algoritma yang akan dikemukakan di sini bersifat tidak lazim karena eliminasi Gauss biasanya berurusan dengan persamaan simultan yang memiliki tepat sebuah sistem solusi. Konvensi yang dipakai untuk mempermudah penulisan algoritma berikut ini dalam bentuk *pseudo-code* adalah:

- **Col[n]** berarti matriks kolom ke-n dari matriks M.
- **Row[n]** berarti matriks baris ke-n dari matriks M.
- **0** berarti matriks nol yang sesuai dengan dimensi matriks yang bersesuaian.
- Elemen – elemen matriks M adalah $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$

Algoritmanya adalah sebagai berikut:

1. Jika **Col[1] = 0**, lanjutkan ke nomor 5; selain itu, lanjutkan ke nomor 2.

2.- Lakukan pertukaran baris (bila diperlukan) agar elemen *a* berisi angka bukan nol.

- Reduksi baris lainnya (baris kedua dan baris ketiga) dengan baris pertama.
- Jika e dan h bernilai nol, lanjutkan ke nomor 4; selain itu lanjutkan ke nomor 3.

3.- Lakukan pertukaran baris kedua dengan baris ketiga (bila diperlukan) agar elemen e berisi angka bukan nol.

- Reduksi baris ketiga.
- Jika $i = 0$, maka vektor eigen yang dihasilkan adalah $\begin{pmatrix} bf - ce \\ -af \\ ae \end{pmatrix}$; selain itu, tidak terdapat solusi non-trivial untuk sistem ini. Algoritma dihentikan.

4.- Jika f dan i bernilai nol, maka vektor-vektor eigen yang dihasilkan adalah $\begin{pmatrix} -c \\ 0 \\ a \end{pmatrix}$ dan $\begin{pmatrix} -b \\ a \\ 0 \end{pmatrix}$; selain itu, vektor eigen yang dihasilkan hanya $\begin{pmatrix} -b \\ a \\ 0 \end{pmatrix}$. Algoritma dihentikan.

5. Jika **Col[2] = 0**, lanjutkan ke nomor 7; selain itu, lanjutkan ke nomor 6.

6.- Lakukan pertukaran baris (bila diperlukan) agar elemen b berisi angka bukan nol.

- Reduksi kedua baris lainnya (baris kedua dan baris ketiga).
- Jika f dan i bernilai nol, maka vektor-vektor eigen yang dihasilkan adalah $\begin{pmatrix} 0 \\ -c \\ b \end{pmatrix}$ dan $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$; selain itu, vektor eigen yang dihasilkan hanyalah $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$. Algoritma dihentikan.

7. Jika **Col[3] = 0**, maka vektor-vektor eigen yang dihasilkan adalah $\begin{pmatrix}1\\0\\0\end{pmatrix}$, $\begin{pmatrix}0\\1\\0\end{pmatrix}$, dan $\begin{pmatrix}0\\0\\1\end{pmatrix}$; selain itu, vektor-vektor eigen yang dihasilkan adalah $\begin{pmatrix}1\\0\\0\end{pmatrix}$ dan $\begin{pmatrix}0\\1\\0\end{pmatrix}$. Algoritma dihentikan.

Setelah algoritma dihentikan, maka vektor-vektor eigen telah dikalkulasi seluruhnya. Agar tidak kehilangan jejak, maka nilai-nilai eigen perlu dicatat ulang sehingga jelas nilai eigen mana yang menghasilkan vektor eigen tertentu.

5. PROSES ORTONORMALISASI GRAM-SCHMIDT

Sesudah ketiga vektor eigen dikalkulasi pada proses sebelumnya, yang perlu dilakukan adalah proses ortonormalisasi.

Konvensi yang dipakai dalam proses ortonormalisasi yang akan dijabarkan adalah:

- u_n adalah vektor eigen ke-n yang belum diortonormalisasi.
- v_n adalah vektor eigen ke-n yang telah diortonormalisasi.
- v'_n adalah vektor eigen perantara ke-n yang masih harus dibagi dengan panjang vektornya.
- | | adalah notasi yang dipakai untuk menandakan panjang vektor tertentu. Contohnya: $|\mathbf{u_n}|$ berarti $\sqrt{u_{1n}{}^2 + u_{2n}{}^2 + u_{3n}{}^2}$

Proses ortonormalisasi dilakukan dengan langkah-langkah sebagai berikut:

1. $v_1 = \frac{u_1}{|u_1|}$. Dengan proses normalisasi ini, $|\mathbf{v_1}| = 1$, atau $v_1 . v_1 = 1$.

2. $v'_2 = u_2 - (u_2 . v_1)\, v_1$.

3. $v_2 = \frac{v'_2}{|v'_2|}$.

4. $v'_3 = u_3 - (u_3 . v_1)\, v_1 - (u_3 . v_2)\, v_2$.

5. $v_3 = \frac{v'_3}{|v'_3|}$. [3]

Dengan berakhirnya proses ortonormalisasi Gram-Schmidt, matriks ortonormal yang dapat mendiagonalisasi matriks simetrik momen orde kedua telah diperoleh dan siap dipergunakan.

6. CONTOH KASUS

Untuk matriks $\begin{pmatrix} 5 & 1 & 6 \\ 1 & 3 & 4 \\ 6 & 4 & 2 \end{pmatrix}$, informasi yang didapat dari matriks tersebut adalah:

- Persamaan karakteristiknya: $-\lambda^3 + 10\lambda^2 + 22\lambda - 112 = 0$
- Nilai-nilai eigen untuk matriks tersebut (akar-akar persamaan karakteristik di atas) adalah $\lambda_1 = -3{,}76195102528386; \lambda_2 = 11{,}0733541019519; \lambda_3 = 2{,}68859692333192$.
- Matriks ortonormal untuk diagonalisasi matriks tersebut:

$$\begin{pmatrix} -0.489844911096367 & 0.678427375439653 & 0.547529231481806 \\ -0.389069051316402 & 0.391911406367246 & -0.833685026174156 \\ 0.780177695387882 & 0.621403046189276 & -0.0719792873397666 \end{pmatrix}$$

[3] Gerber. *Op. Cit.* Hal 237.