PERTEMUAN 1 SEJARAH DAN PERKEMBANGAN GRAFIKA KOMPUTER

Grafika Komputer

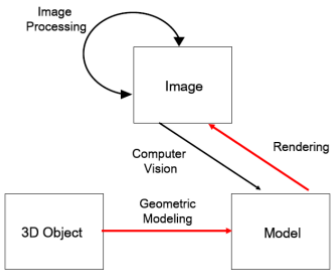


Image Processing

Memperbaiki kualitas gambar, dilihat dari aspek radiometrik dan geometrik:

- **Radiometrik**: Peningkatan kontras, transformasi warna, restorasi citra.
- **Geometrik**: Rotasi, translasi, skala, transformasi geometrik.

Computer Vision

Proses otomasi yang mengintegrasikan sejumlah besar proses untuk persepsi visual:

- Akuisisi citra
- Pengolahan citra
- Klasifikasi, recognition, dan pengambilan keputusan
- Penyusunan deskripsi tentang objek yang terkandung pada suatu gambar atau mengenali objek yang ada pada gambar

Grafika Komputer: Definisi

Proses pembuatan, penyimpanan, dan manipulasi model dan citra:

- Model berasal dari beberapa bidang seperti fisika, matematika, artistik, dan struktur abstrak
- Menciptakan gambar berdasarkan deskripsi objek maupun latar belakang yang terkandung pada gambar tersebut
- Membuat gambar objek sesuai dengan objek tersebut di alam nyata (realisme)

Mengapa Belajar Grafika Komputer?

Grafika komputer merupakan bagian penting dari kurikulum informatika/ilmu komputer:

- **Entertainment**: Animasi komputer-film untuk visualisasi proses, simulator, fenomena alam.
- **User Interfaces**: Memudahkan penggunaan perangkat melalui antarmuka pengguna.
- **Visualisasi Interaktif**: Dalam bisnis dan ilmu pengetahuan.
- **CAD/CAM**: Perancangan, menggambar, visualisasi, dan analisis.
- **Computer Art**: Digunakan dalam commercial art dan fine art.
- **Game**: Pembuatan dan penggunaan game komputer, video player, dan perangkat khusus.

Sejarah Perkembangan Grafika Komputer

- **1950**: The Whirlwind Computer di MIT memiliki Cathode Ray Tube (CRT).
- **1963**: Ivan Sutherland memperkenalkan sketchpad dalam thesisnya.
- **1964**: Munculnya CAD/CAM di General Motor dengan DAC system.
- **1970-an**: Grafika berkembang lambat karena mahalnya perangkat keras.
- **1980-an**: Microchip memungkinkan arsitektur raster, munculnya standar grafika.
- **Akhir 1980-an**: Superworkstation muncul dengan kemampuan grafika tinggi.
- **1990-an**: X Windows mendominasi kelas workstation.
- **1989**: Film Tin Toy (Pixar) memenangkan Academy Award.
- **1995**: Toy Story, film 3D animasi panjang pertama.
- **Akhir 1990-an**: Teknologi visualisasi interaktif berkembang.
- **2000**: Teknologi perangkat keras untuk real-time photorealistic rendering ditemukan.

Graphs and Charts

- Memberikan impresi yang lebih dalam pada data.
- Data dulu digambarkan dengan plotter atau dirender dengan character printer.
- Sekarang, grafik presentasi seperti chart, bar chart, dan pie chart adalah kemudahan tambahan dalam software spreadsheet.

Data Visualizations

- **Scientific Visualization**: Visualisasi model matematis dan struktural.
- **Business Visualization**: Visualisasi model/data dari dunia bisnis.

Education and Training

- Memanfaatkan visualisasi untuk meningkatkan pemahaman konsep.
- Menambah aspek hiburan dalam materi edukatif.
- Membuat simulator untuk melatih keterampilan teknis dengan risiko dan biaya yang lebih rendah.

Computer Art

Karya seni yang diciptakan dengan bantuan komputer.

Computer Animation & Entertainment

- Mensubstitusi bagian film
- Menghasilkan efek visual
- Menghasilkan makhluk jadi-jadian
- Pembuatan film secara total

Video Games

- Game dapat dijalankan pada PC, video player dengan monitor TV, dan perangkat keras khusus.
- Alat input interaktif seperti mouse dan joystick diperlukan.

Augmented Reality

Teknologi yang menggabungkan secara real-time digital konten yang dibuat oleh komputer dengan dunia nyata, memungkinkan pengguna melihat objek maya 2D atau 3D yang diproyeksikan terhadap dunia nyata.

PERTEMUAN 2

Pengantar OpenGL

Mengapa OpenGL?

- **Device Independence**
- **Platform Independence** (SGI IRIX, Linux, Windows)
- **Abstraction** (GL, GLU, GLUT)
- **Open Source**
- **Hardware-independent software interface**
- **Support of client-server protocol**
- **Other APIs** (OpenInventor -> object-oriented toolkit, DirectX (Microsoft), Java3D (Sun))

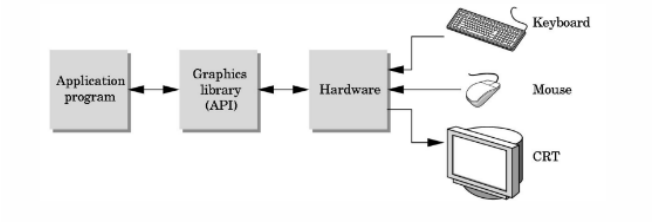
Pengertian OpenGL

- OpenGL merupakan utiliti yang menjembatani (interfacing) antara peralatan grafis (graphic hardware) dengan bahasa pemrograman untuk membangun citra grafis 2D dan 3D.
- OpenGL adalah library standar yang dikembangkan oleh Silicon Graphics, Inc.
- OpenGL Library berisi fungsi-fungsi dan prosedur yang dapat dipanggil oleh bahasa pemrograman untuk mengakses peralatan grafis sehingga memudahkan dalam membuat

aplikasi yang memanfaatkan peralatan grafis (Segel, et al., 2003).

Interface Programmer

Programmer melihat sistem grafis melalui suatu interface software: the Application Programmer Interface (API).



Konten API

- Fungsi-fungsi yang secara spesifik apa yang dibutuhkan untuk suatu bentuk image.
    - Objects
    - Viewer
    - Light Source(s)
    - Materials
- Informasi yang lain
    - Input dari devices seperti mouse dan keyboard
    - Capabilities of system

Fitur dalam OpenGL

- Transformasi 3D (Rotasi, scaling, translasi, perspektif)
- Model Colour (Warna) (Nilai-nilai: R, G, B)
- Lighting (Flat shading, Gouraud shading, Phong shading)
- Rendering (Texture mapping)
- Modeling (non-uniform rational B-spline (NURB), curves, surfaces)
- Others: atmospheric fog, alpha blending, motion blur

Sejarah OpenGL

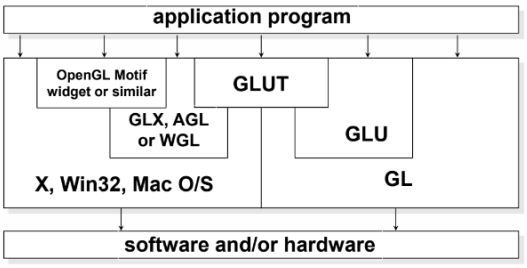
- Silicon Graphics (SGI) merevolusi workstation grafis dengan mengimplementasikan pipeline dalam hardware (1982).
- Untuk mengakses sistem, programmer aplikasi menggunakan library yang disebut GL.
- Dengan GL, relatif sederhana untuk memprogram aplikasi interaktif tiga dimensi.

OpenGL Libraries

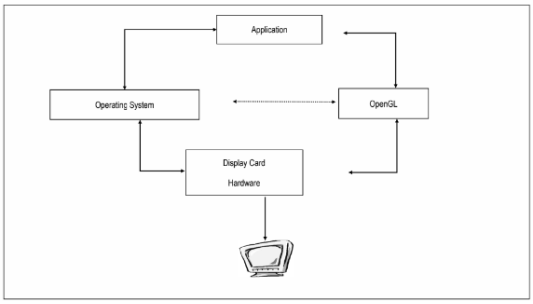
- **GL (Graphics Library):**
    - Primitif (titik, garis, polygon)
    - Shading dan colour
    - Translasi, rotasi, scaling
    - Viewing, Clipping, Texture
    - Hidden surface removal
- **GLU (GL Utilities):** Jenis-jenis fungsi yang berhubungan dengan pengaturan kamera dan deskripsi bentuk tingkat tinggi (Viewing -> perspektif, sphere).
- **GLUT (GL Utility Toolkit):** Toolkit independen window-system dengan banyak fungsi utilitas, sebagian besar berhubungan dengan user interface (key, mouse, handler, window events).

Organisasi Software

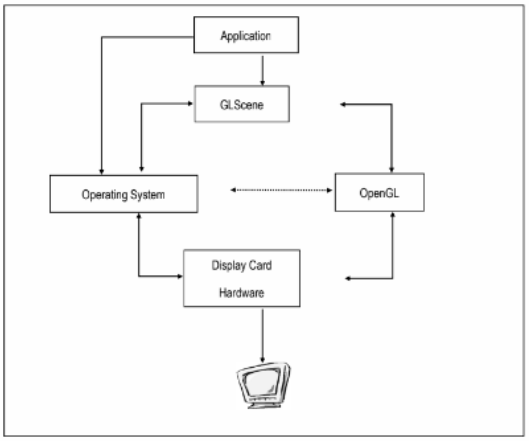
- **Software/hardware**



OpenGL System



GLScene



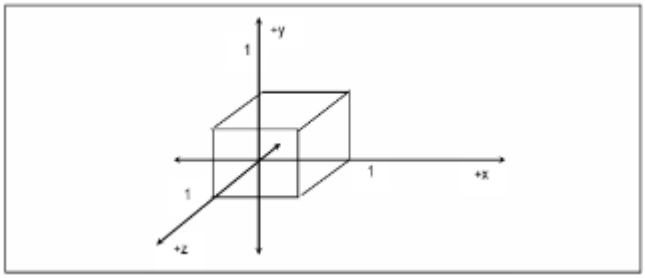
Overview

GLScene merupakan sekumpulan prosedur dan fungsi yang disusun dalam bentuk komponen VCL (visual components library) yang merupakan suatu GLU (Graphics Library Utility) yang menjembatani antara bahasa pemrograman dengan OpenGL Library. GLScene khusus digunakan untuk mengakses fungsi-fungsi dan prosedur grafis 3D yang ada pada OpenGL Library (GL command) untuk memudahkan dalam membuat suatu aplikasi 3D dalam bahasa pemrograman visual Delphi (Grange, 2003).

Grafika Komputer 3D

Teknik grafika komputer 3D adalah cara membuat dan menampilkan objek grafis dalam bentuk 3-dimensi pada peralatan display (Hearn, et al., 1994). Objek grafis yang paling sederhana adalah titik dan garis. Setiap objek grafis mempunyai atribut geometri dan atribut karakteristik lainnya seperti warna, tekstur, tingkat transparansi, dan lain-lain. Atribut geometri merupakan atribut dasar yang berisi informasi tentang keberadaan objek pada suatu sistem koordinat tertentu.

Pemodelan Objek 3D

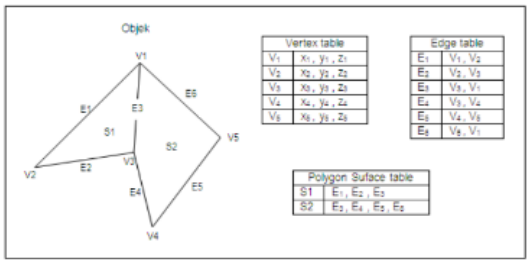


- Secara umum, setiap objek grafis dibentuk dari gabungan beberapa objek grafis sederhana. Misalnya, dua buah titik yang saling dihubungkan melalui suatu algoritma tertentu dapat membentuk suatu pola garis ataupun kurva tertentu.

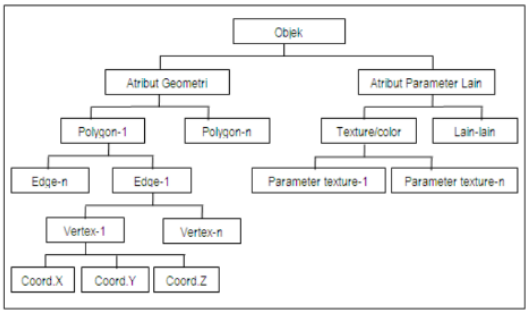
- Objek grafis 3D juga dibentuk dari objek-objek grafis sederhana seperti titik dan garis. Objek grafis 3D dirancang berdasarkan sistem koordinat 3-dimensi.

Representasi Objek 3D

- Representasi objek adalah cara menjelaskan atau menggambarkan karakteristik suatu objek. Setiap objek grafis mempunyai karakteristik tertentu berupa atribut geometri dan berbagai atribut parameter karakteristik lainnya yang digunakan untuk menjelaskan tentang keberadaan serta ciri yang dimiliki objek tersebut. Setiap objek grafis akan ditampilkan berdasarkan informasi deskripsinya.
- Ada berbagai skema representasi objek 3D. Objek yang berbeda akan mempunyai skema representasi yang berbeda pula, misalnya representasi dengan polygon dan quadric surface sangat baik untuk merepresentasikan objek-objek sederhana seperti polyhedronsdan ellipsoids. Representasi spline surfaces baik digunakan untuk objek-objek berbentuk kurva seperti sayap pesawat terbang dan lain-lain.
- Skema representasi untuk objek-objek solid umumnya dibagi dalam dua kategori.B-reps (boundary representations) menjelaskan objek 3D sebagai kumpulan dari suatu permukaan (surface) yang membatasi antara bagian objek sebelah dalamdengan lingkungannya.
- Sedangkan Space-partitioning representations digunakan untuk menjelaskan properti bagian sebelah dalam (bagian-bagian dari objek itu sendiri). Representasi untuk skema boundary yang umum digunakan adalah polygon surfaces representation. Polygon adalah objek grafis yang merupakan gabungan dari beberapa garis sehingga membentuk suatu pola atau shape tertutup. Representasi polygon surfaces menjelaskan Objek 3D sebagai kumpulan polygon yang menutupi bagian di permukaan objek.
- Dalam representasi polygon surfaces, setiap objek mempunyai karakteristik tertentu yang dinyatakan dalam satu atau lebih polygon. Setiap polygon mempunyai informasi karakteristik tertentu yang dinyatakan dalam sekumpulan garis (edge). Setiap garis mempunyai karakteristik tertentu yang dinyatakan dalam sekumpulan titik (vertex). Dan setiap titik dinyatakan dalam bentuk sekumpulan nilai koordinat (x,y,z). Semua informasi ini disimpan dalam masing-masing daftar tertentu seperti yang terlihat pada gambar 2.3 berikut ini :

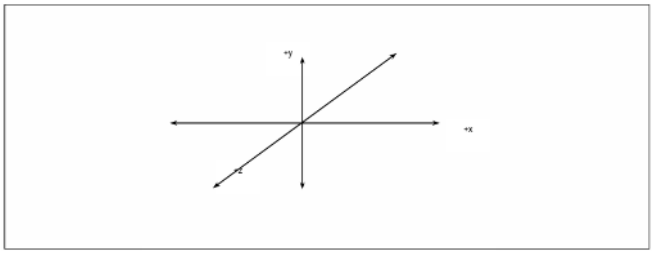


- Secara hirarki, struktur penyimpanan informasi deskripsi objek dalam polygon surfaces representation dalam gambar 2.4:

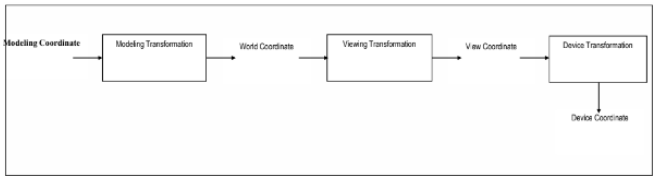


Sistem Koordinat 3D

Pada dasarnya, sistem koordinat 3-dimensi tidak berbeda dengan sistem koordinat 2-dimensi, yaitu sumbu-x secara horisontal, sumbu-y secara vertikal, dan menambahkan sumbu-z untuk menyatakan kedalaman.



Dalam grafika komputer 3D, untuk dapat menampilkan objek dalam tampilan 3-dimensi pada peralatan display, perlu dilakukan beberapa proses transformasi geometri dari sistem koordinat. Proses transformasi ini dilakukan dalam beberapa tahap seperti yang terlihat pada gambar.2.6 berikut :



Keterangan :

- Koordinat model (modeling coordinates) adalah sistem koordinat yang digunakan dalam pembentukan model.
- Koordinat lingkungan (world coordinates) adalah sistem koordinat yang digunakan pada lingkungan tempat objek diletakkan.
- Koordinat proyeksi atau penampakan (viewing coordinate) adalah sistem koordinat yang digunakan dalam memproyeksikan kenampakan objek berdasarkan posisi dan sudut pandang (orientation).
- Koordinat display (device coordinate) adalah sistem koordinat yang digunakan pada peralatan display.

OpenGL Function Format

- **Format**


```
glVertex3fv(p)
```

p adalah pointer to an array

Contoh: simple.c

```
#include <GL/glut.h>
void mydisplay(){
  glClear(GL_COLOR_BUFFER_BIT);
  glBegin(GL_POLYGON);
        glVertex2f(-0.5, -0.5);
        glVertex2f(-0.5, 0.5);
        glVertex2f(0.5, 0.5);
        glVertex2f(0.5, -0.5);
  glEnd();
  glFlush();
}
int main(int argc, char** argv){
  glutCreateWindow("simple");
  glutDisplayFunc(mydisplay);
  glutMainLoop();
}
```

Event Loop

- Program mendefinisikan display callback function dinamakan **mydisplay**.
    - Setiap program GLUT harus punya display callback.

- Display callback dijalankan kapan saja OpenGL memutuskan display harus di-refresh, contoh: ketika window dibuka.
- Fungsi main berakhir saat program memasuki suatu event loop.

Tutorial OpenGL

OpenGL adalah suatu API yang terdiri dari beberapa library dengan berbagai tingkat abstraksi: GL, GLU, dan GLUT.

- GL
  - Lowest level: vertex, matrix manipulation
  - glVertex3f(point.x, point.y, point.z)
- GLU
  - Helper functions for shapes, transformations
  - gluPerspective(fovy, aspect, near, far)
- GLUT
  - Highest level: Window and interface management
  - glutSwapBuffers()

OpenGL Implementations

OpenGL adalah API yang tersedia dalam platform spesifik implementasi seperti Windows, Linux, UNIX, dll.
OpenGL IS an API (think of as collection of .h files):

- #include <GL/gl.h>
- #include <GL/glu.h>
- #include <GL/glut.h>

Windows: opengl32.lib glu32.lib glut32.lib
Linux: -l GL -l GLU –l GLUT

OpenGL API

Sebagai programmer, Anda perlu melakukan hal-hal berikut:

- Tentukan lokasi/parameter kamera.
- Tentukan geometri (dan penampilan).
- Tentukan pencahayaan (opsional).

OpenGL akan menghitung hasil gambar 2D.

Konvensi OpenGL (Conventions)

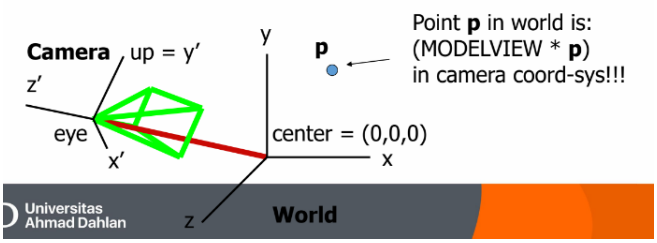
- Banyak fungsi memiliki beberapa bentuk:
  - **glVertex2f**, **glVertex3i**, **glVertex4dv**, dll.
- Angka menunjukkan jumlah argumen.
- Huruf menunjukkan tipe:
  - **f: float**
  - **d: double**
  - **ub**: unsigned byte, dll.
- **v** (jika ada) menunjukkan argumen pointer tunggal:
  - **glVertex3f(point.x, point.y, point.z)**
  - **glVertex3fv(point)**

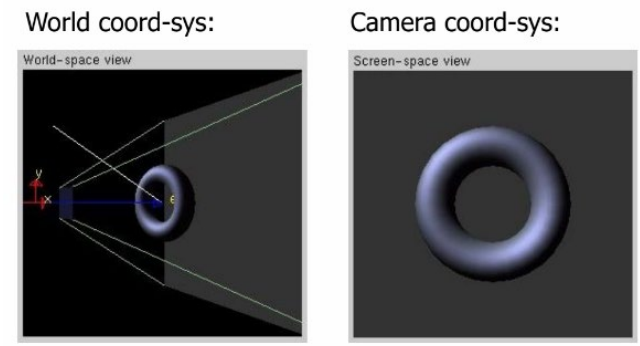
OpenGL: Camera

- Dua hal yang harus ditentukan:
  - Lokasi fisik kamera di scene (MODELVIEW matrix dalam OpenGL):
    - Dimana kamera?
    - Arah mana yang kamera tunjukkan?
    - Orientasi kamera?
  - Sifat proyeksi kamera (PROJECTION matrix dalam OpenGL):
    - Kedalaman bidang?
    - Field of view di arah x dan y?
- Sistem Koordinat direpresentasikan sebagai matriks dalam OpenGL.
- Kamera diimplikasikan sebagai suatu sistem koordinat dipusatkan pada bidang gambar.
- Oleh karena itu, menentukan matriks ini berarti menentukan sifat fisik kamera.

OpenGL: MODELVIEW

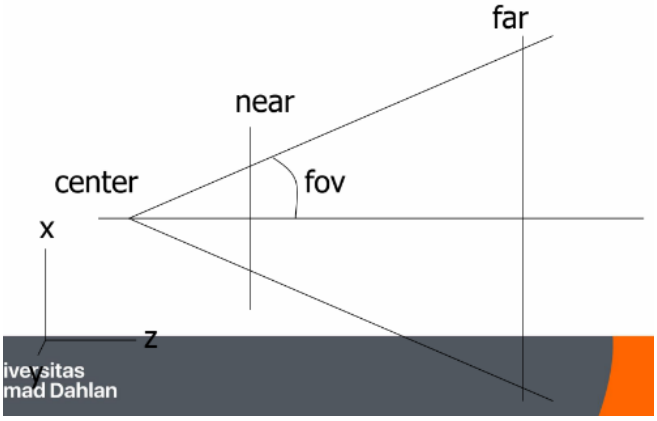
```
glMatrixMode(GL_MODELVIEW); // Specify matrix mode
glLoadIdentity(); // Clear modelview matrix
//Utility function provided by the GLU API (included with OpenGL)
gluLookAt(eyex,eyey,eyez,centerx,centery,centerz,upx,upy,upz);
```



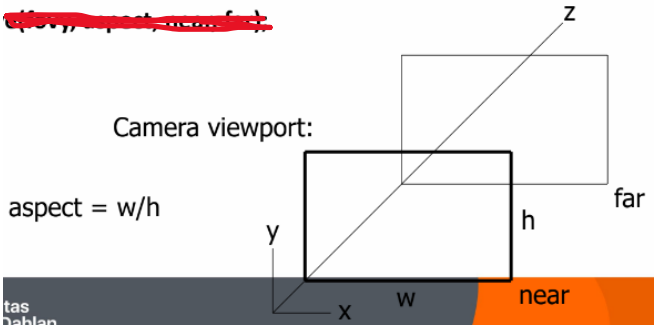


OpenGL: PROJECTION

Intrinsic (optical) properties of camera:



glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluPerspective(fovy, aspect, near, far);



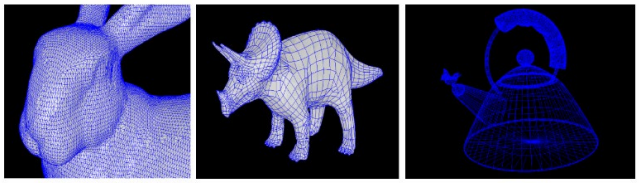
OpenGL: Setting Camera

Assume window is widthxheight:

```
void SetCamera()
{
  glViewport(0, 0, width, height);
  /* Set camera position */
  glMatrixMode(GL_MODELVIEW);
  glLoadIdentity();
  gluLookAt(m_vEye[0], m_vEye[1], m_vEye[2],
        m_vRef[0], m_vRef[1], m_vRef[2],
        m_vUp[0],  m_vUp[1],  m_vUp[2]);
  /* Set projection frustrum */
  glMatrixMode(GL_PROJECTION);
  glLoadIdentity();
  gluPerspective(m_fYFOV, width / height, m_fNear, m_fFar);
}
```

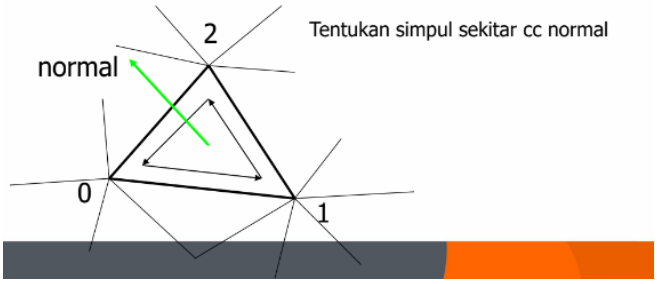
OpenGL: Geometry

Tentukan geometri menggunakan primitif: segitiga, segi empat, garis, dll.



OpenGL: Triangle Mesh

Merepresentasikan permukaan objek melalui sekumpulan orientasi segitiga:



OpenGL: glBegin()...glEnd()

Semua geometri harus dibungkus antara **glBegin()** dan **glEnd()**.

```
glBegin(GL_TRIANGLES);
for (int i=0; i<ntris; i++)
{
   glColor3f(tri[i].r0,tri[i].g0,tri[i].b0);        // Color of vertex
   glNormal3f(tri[i].nx0,tri[i].ny0,tri[i].nz0);   // Normal of vertex
   glVertex3f(tri[i].x0,tri[i].y0,tri[i].z0);       // Position of vertex
   ...
   glColor3f(tri[i].r2,tri[i].g2,tri[i].b2);
   glNormal3f(tri[i].nx2,tri[i].ny2,tri[i].nz2);
   glVertex3f(tri[i].x2,tri[i].y2,tri[i].z2);
}
glEnd();  // Sends all the vertices/normals to the OpenGL library
```

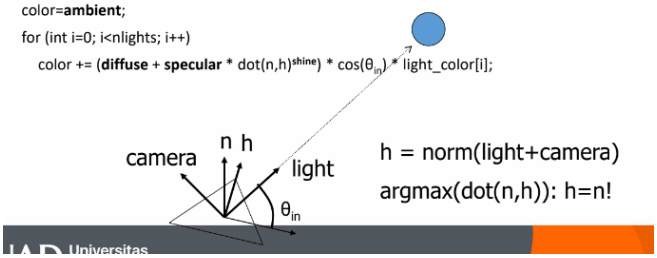
OpenGL mendukung beberapa primitif:

- glBegin(GL_LINES);
- glBegin(GL_QUADS);
- glBegin(GL_POLYGON);

Gunakan GL_LINES untuk menentukan garis dalam ruang 3D (seperti sinar/ray). Semua vertek dapat punya suatu normal.

OpenGL: Lighting

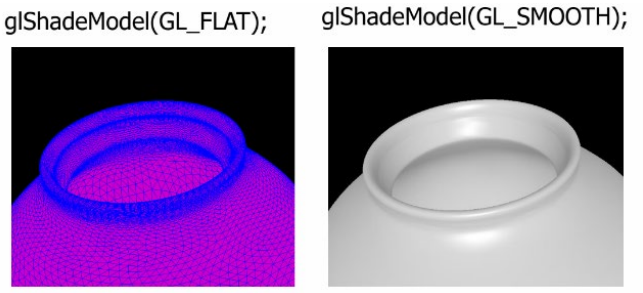
OpenGL menghitung warna pada setiap vertex dengan suatu penghitungan shading




```
GLfloat mat_diffuse[4] = {0.75, 0.75, 0.75, 1.0};
GLfloat mat_specular[4] = {0.55, 0.55, 0.55, 1.0};
GLfloat mat_shininess[1] = {80};

glMaterialfv(GL_FRONT_AND_BACK,GL_AMBIENT_AND_DIFFUSE,mat_
diffuse);
glMaterialfv(GL_FRONT,GL_SPECULAR,mat_specular);
glMaterialfv(GL_FRONT,GL_SHININESS,mat_shininess);

GLfloat light0_ambient[4] = { 0.0, 0.0, 0.0, 1.0};
GLfloat light0_color[4] = { 0.4, 0.4, 0.4, 1.0 };
glLightfv(GL_LIGHT0, GL_AMBIENT, light0_ambient);
glLightfv(GL_LIGHT0, GL_DIFFUSE, light0_color);
glLightfv(GL_LIGHT0, GL_SPECULAR, light0_color);

glEnable(GL_LIGHT0);
glEnable(GL_LIGHTING);
```

- Ada dua "mode shading" dalam OpenGL:
  - **Shading Datar (Flat Shading)**: Seluruh wajah (face) memiliki warna yang dihitung pada titik vertex ke-0.
  - **Shading Gouraud (Smooth Shading)**: Warna dihitung pada setiap vertex dan diinterpolasi di seluruh polygon.

OpenGL: Flat Shading



OpenGL: Lighting

- Lighting merupakan suatu opsi
  - glEnable(GL_LIGHTING);
  - glDisable(GL_LIGHTING);
- Saat disabled, warna dari tiap vertex secara langsung glColor3f(r,g,b).
- Tidak dibutuhkan lighting saat debugging menelusuri sinar (ray tracer).

GLUT

- GLUT adalah API yang sangat sederhana yang mendukung pembuatan aplikasi GUI+OpenGL.
- Panggil fungsi-fungsi ini dalam fungsi **main**:

```
/* Inisialisasi mesin GLUT */
glutInit(&argc, argv);
glutInitWindowPosition(100, 100);
glutInitWindowSize(window_width, window_height);
glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
main_window = glutCreateWindow("Ray Viewer");

/* Daftarkan callback */
glutDisplayFunc(MainRedraw); // Fungsi redraw utama

/* Jalankan antarmuka interaktif */
glutMainLoop();
```

GLUT: MainRedraw

```
void MainRedraw()
{
  glClearColor(0, 0, 0, 1);
  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
```

```
  SetCamera();
  DrawGeometry();
  glutSwapBuffers(); // Perbarui layar
}
```

PERTEMUAN 3: TRANSFORMASI GEOMETRI

scan konversi : objek → primitif → pixel

Transformasi 2 objek : objek →objek

Dalam 2D, primitif yang mungkin adalah titik, garis, poligon

Untuk mengetahui tentang transformasi dibutuhkan dasar aljabar vektor

Trigonometry Table

Angles (in Degrees)	0°	30°	45°	60°	90°	180°	270°
Angles (in Radians)	0	π/6	π/4	π/3	π/2	π	3π/2
Sin	0	1/2	1/√2	√3/2	1	0	-1
Cos	1	√3/2	1/√2	1/2	0	-1	0
Tan	0	1/√3	1	√3	Not Defined	0	Not Defined
Cot	Not Defined	√3	1	1/√3	0	Not Defined	0
Cosec	Not Defined	2	√2	2/√3	1	Not Defined	-1
Sec	1	2/√3	√2	2	Not Defined	-1	Not Defined

Translasi

menggeser objek geometri tanpa mengubah ukuran atau orientasinya. Ini dilakukan dengan menambahkan nilai konstan pada koordinat x dan y setiap titik pada objek.

P'= (x + tx, y + ty)

Scaling

Mengubah ukuran objek geometri dengan mengalikan koordinat x dan y setiap titik pada objek dengan faktor skala tertentu.

$$P' = (S_x \times x, S_y \times y)$$

Pencerminan

Menciptakan bayangan cermin dari objek geometri terhadap garis tertentu yang disebut sumbu pencerminan. Setiap titik pada objek dicerminkan pada posisi yang sama di sisi berlawanan dari sumbu.

1. Sumbu X:

P' = (x, -y)

2. Sumbu Y:

P' = (-x, y)

3. Garis y = x:

P' = (y, x)

4. Garis y = -x:

P' = (-y, -x)

5. Garis y = h

P'= (2h-x,y)

5. Garis x = k

P'= (x,2k-y)

Rotasi

- Rotasi adalah proses memutar suatu objek atau titik di sekitar suatu sumbu putar.
- Rotasi dua dimensi pada suatu objek akan memindahkan objek tersebut menurut garis melingkar.
- Perlu sudut rotasi dan pivot point (xp,yp) atau titik rotasi dimana objek tersebut dirotasi
- Nilai positip dari sudut rotasi menentukan arah rotasi berlawanan dengan jarum jam, demikian sebaliknya nilai negative akan memutar objek searah dengan jarum jam
- r menyatakan jarak konstan dari titik pusat.Sedangkan sudut adalah sudut posisi suat titik dengan sumbu horizontal, sedangkan sudut adalah sudut rotasi.

Rotasi 2 Dimensi

- Rotasi terhadap titik (0, 0)

x' = xcos(θ) - ysin(θ)

y' = xsin(θ) + ycos(θ)

- Rotasi terhadap titik sembarang (a,b)

x' = (x - a)cos(θ) - (y - b)sin(θ) + a

y' = (x - a)sin(θ) + (y - b)cos(θ) + b

Rotasi 3 Dimensi

- Rotasi terhadap sumbu X:

x' = x

y' = ycos(θ) - zsin(θ)

z' = ysin(θ) + zcos(θ)

- Rotasi terhadap sumbu Y:

x' = xcos(θ) + zsin(θ)

y' = y

z' = -xsin(θ) + zcos(θ)

- Rotasi terhadap sumbu Z:

x' = xcos(θ) - ysin(θ)

y' = xsin(θ) + ycos(θ)

z' = z

PERTEMUAN 4: ALGORITMA GARIS

Garis harus tampak bagus dengan meminimalkan jagged effect atau meminimalkan aliasing

Mempunyai ketebalan garis yang seragam

Membutuhkan waktu yang singkat

Persamaan garis lurus:

$$y = mx + b$$

Bila ada dua titik yang akan dihubungkan $(x_1, y_1) dan\ (x_2, y_2)$

Dan titik potong b dapat dihitung dengan:

$$b = y_1 - mx_1$$

Algoritma DDA (Digital Differential Analyzer)

Algoritma scan-konversi garis dengan melakukan sampling pada garis di rentang Δx atau Δy

Dioptimalkan untuk mempercepat kecepatan menggambar garis

Algoritma DDA lebih cepat dari solusi sebelumnya akrena tidak ada operasi perkalian

Akumulasi error pada pembulatan dapat membuat piksel jauh dari garis.

Operasi pembulatan dan operasi pada floating point memakan waktu

Contoh menggambar garis:

$$(2,2),(3, 2\tfrac{3}{5}),(4, 3\tfrac{1}{5}), (5, 3\tfrac{4}{5}), (6, 4\tfrac{2}{5}),(7 ,5)$$

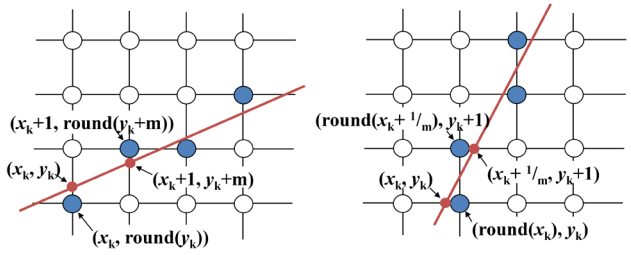
Koordinat x naik satu demi Saturday koordinat y naik berdsarkan slope dari garis

Ketika slope berada pada nilai $-1 \leq m \leq 1$ dan mulai dari titik awal

Dengan menaikkan koordinaat y dengan 1 maka x dapat dihitung

$$x_{k+1} = x_k + \frac{1}{m}$$

Nilai hasil perhitungan harus dibulatkan agar cocok dengan nilai piksel



Algoritma Bresenham

Merupakan algoritma yang sekarang digunakan di computer grafis modern

Lebih baik dari algoritma DDA, karena

Incremental algorithm : nilai sekarang menggunakan nilai sebelumnya

Digunakan untuk tipe data integer : menghindari operasi floating point

Algoritma bresenham menggunakan fungsi keputusan untuk menentukan letak koordinat selanjutnya

Diberikan 2 titik (x_1, y_1) dan (x_2, y_2)

Dicari titik penghubung di antara kedua dengan sumsi:

$$x_1 < x_2$$

$$0 \leq m \leq 1$$

Tentukan titik awal garis (x_1, y_1) dan akhir garis (x_2, y_2)

Inisialisasi awal, hitung:

selisih lebar $= \Delta x = x_2 - x_1$

$$2\Delta y = 2(y_2 - y_1)$$

Inisial parameter keputusan $= p_0 = 2\Delta y - \Delta x$

Setiap x_k di sepanjang garis, mulai dari $k = 0$, cek kondisi berikut

Jika $p_k < 0$ maka titik selanjutnya untuk digambar di: $(x_k + 1, y_k)$
$P_{k+1} = p_k + 2\Delta y$
Selain it umaka titik selanjutnya utnuk digambar di: $(x_k + 1, y_k + 1)$
$P_{k+1} = P_k + 2\Delta y - 2\Delta x$
Ulangi langkah di atas sebanyak Δx
Bila $m > 1$
Inisial parameter keputusan $= p_0 = 2\Delta x - \Delta y$
Setiap y_k di sepanjang garis, mulai dari $k = 0$, cek kondisi berikut
Jika $p_k < 0$ maka titik selanjutnya untuk digambar di: $(x_k, y_k + 1)$
$P_{k+1} = p_k + 2\Delta y$
Selain it umaka titik selanjutnya utnuk digambar di: $(x_k + 1, y_k + 1)$
$P_{k+1} = P_k + 2\Delta y - 2\Delta x$
Ulangi langkah di atas sebanyak Δy
Contoh:
Bila diberikan 2 titik yang akan dihubungkan garis A(20,10) dan B(30,18)
Hitung:
$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{\Delta y}{\Delta x} = \frac{8}{10} = 0.8$
$\Delta x = 10$
$\Delta y = 8$
$P_0 = 2\Delta y - \Delta x = 6$
$2\Delta y = 16$
$2\Delta y - 2\Delta x = -4$

k	p_k	(x_{k+1}, y_{k+1})	k	p_k	(x_{k+1}, y_{k+1})
0	6	(21,11)	5	6	(26,15)
1	2	(22,12)	6	2	(27,16)
2	-2	(23,12)	7	-2	(28,16)
3	14	(24,13)	8	14	(29,17)
4	10	(25,14)	9	10	(30,18)

PERTEMUAN 5-6: INTERPOLASI (TWEENING)

menyisipkan di antara dua bagian yang berbeda atau memperkirakan nilai dari suatu fungsi antara dua nilai yang telah diketahui. Pada komputer grafik, interpolasi digunakan untuk menggabungkan beberapa efek yang ingin dilakukan pada suatu obyek.
Interpolasi digunakan untuk memberikan nilai sela di antara dua titik
Ada beragam teknik interpolasi, salah satu di antaranya adalah interpolasi linear.
Nilai sela yang diberikan tergantung dari fungsi interpolasi
Interpolation & Approximation

Interpolation: garis akan melalui semua titik (tidak stabil)

ApproximationGaris tidak selalu melalui semua titik (Lebih stabil)
Konsep Kurva
Kurva: Rentetan titik 1D yang berkelanjutan pada bidang 2D atau 3D
Kurva: Pemetaan sebuah interval pada bidang
Atribut Kurva : warna , ketebalan , pola , bentuk
Representasi kurva
1) Eksplisit
- Dalam bidang x,y

Bila x variable bebasmaka $y=f(x)$atau kebalikannya $x=g(y)$
- Pada bidang 3D

Bila x variable bebas maka $y=f(x)$ dan$z=g(x)$
- Pada permukaan 2D

Dibutuhkan 2 variable bebas $z=f(x,y)$
Contoh:
Garis lurus : $y=mx+b$
Lingkaran: $y = \sqrt{r^2 - x^2}$dan $y = -r^2 - x^2$untuk $0 \leq |x| \leq r$
2) Implisit
- Dalam bidang x,y,representasiimplisitnya

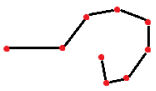
$f(x,y)$=0
- Pada bidang 3D, deskripsipermukaanya

$f(x,y,z)$=0
Contoh:
Garislurus: $ax+by+c$=0
Lingkaran: $x^2 + y^2 - r^2 = 0$
Bola : $x^2 + y^2 + z^2 - r^2 = 0$
3) Parametrik
Setiap variable titikpadakurvadinyatakandengan variable bebas (
- Dalam bidang 3D

$x=x(u)$
y=$y(u)$
$z=z(u)$
- Pada permukaan membutuhkan 2 parameter

$x=x(u,v)$
y=$y(u,v)$
$z=z(u,v)$

Kurva Dapat dimodelkan dengan :
1) Polylines



• Rentetan titik yang terkoneksi oleh garis lurus
• Tidak terlalu halus bentuk kurvanya
• Semua kurva akan dikonversi ke
2) bentuk polyline

• Kurva
• Menggunakan fungsi
• Bentuknya halus / smooth
• Cukup sulit dalam pemodelannya

Kurva Polinomial
Kurva polynomial dengan derajat n didefinisikan
$$y = \sum_{k=0}^{n} a_k x^k = a_0 + a_1 x + \cdots + a_{n-1} x^{n-1} + a_n x^n$$
Derajat 2 = kuadrat ,
Derajat 3 = kubik ,
Derajat 4 = quadric, dst

Mendesain obyek diperlukan titik titik yang mewakili bentuk obyek
Kurva akan dibentuk dari titik titik tersebut (curve fiting)
Misalnya dengan polynomial kubik yang bentuk parametriknya
$$x = a_{x0} + a_{x1}u + a_{x2}u^2 + a_{x3}u^3$$
$$y = a_{y0} + a_{y1}u + a_{y2}u^2 + a_{y3}u^3$$
Dimana parameter u=0...1
Kurva kontinu yang dibentuk dari potongan kurva polynomial
Disebut kurva spline

Macam macam interpolasi:
Interpolasi Nearest Neighbor
Menggunakan pengulangan data terdekat untuk interpolasi
Bila terdapat dua titik yang akan diinterpolasi yaitu
(x_0, y_0) dan (x_1, y_1)
Maka titik sela (x_i, y_i) dari dua titik tersebut adalah pengulangan data yang terdekat dengan titik sela tersebut
$\Delta x < \Delta x$ .menggunakan y_0
$\Delta x > \Delta x$ .menggunakan y_n
Contoh:
Hitung
Nilai interpolasi titik (1,1) sampai (11,5) denganjumlah titiksela n= 5
Jawab:
$n = 5$, jarak kenaikan x disetiap titik sela $= \frac{\Delta x}{5} = \frac{10}{5} = 2$

Sehingga kenaikan nilai $x = 1,3,5,7,9,11$
$untuk\ x_0 = 1, maka\ y_0 = 1 \rightarrow\ koordinat\ titik\ (1,1)$
$untuk\ x_1 = 3,$
Dicari dulu mana jarak yang paling dekat ke x_1 dari kedua ujung titik x_0 dan x_5
$\Delta x_{1,0} = |x_1 - x_0| = 3 - 1 = 2$
$\Delta x_{5,1} = |x_5 - x_1| = 11 - 3 = 8$
Karena jarak terdekat ke x_0 adalah x_1 maka $y_1 = 1 \Rightarrow$ koordinat titik (3,1)
Untuk nilai x yang lain terapkan aturan yang sama
Untuk $x_2 = 5,$
Karena x_2 lebih dekat ke x_0 maka $y_2 = 1 \Rightarrow$koordinat titik koordinat titik(5,1)
Untuk$x_3 = 7,$
Karena x_3 lebih dekat ke x_5 maka $y_3 = 5 \Rightarrow$koordinat titik(7,5)
Untuk $x_4 = 9,$
Karena x_4 lebih dekat ke x_5 maka $y_4 = 5 \Rightarrow$koordinat titik(9,5)
$x_5 = 11$maka $y_5 = 5 \Rightarrow$koordinat titik(11,5)

Interpolasi Linear

Menggunakan fungsi line untuk melakukan interpolasi
Bila terdapat dua titik yang akan diinterpolasi yaitu:
(x_0, y_0) dan (x_1, y_1)
Maka titik sela (x_i, y_i) dari dua titik tersebut:

$$\frac{y_i - y_0}{x_i - x_0} = \frac{y_1 - y_0}{x_1 - x_0}$$

$$y_i = y_0 + (x_i - x_0)\frac{y_1 - y_0}{x_1 - x_0}$$

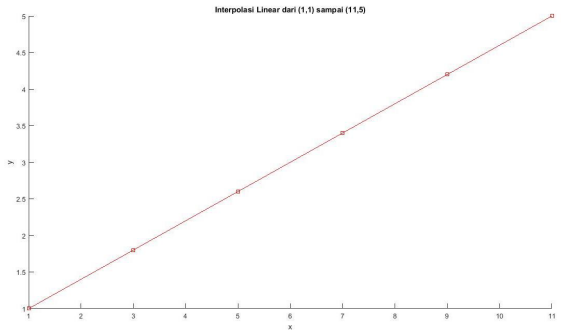
Bila jarak (x_0, y_0) sampai (x_1, y_1) dimisalkan 1 (dinormalisasi)
Diketahui jarak awal (x_0, y_0) sampai titik sela (x_1, y_1) adalah μ maka:

$$y_i = y_0 \times (1 - \mu) + y_1 \times \mu$$

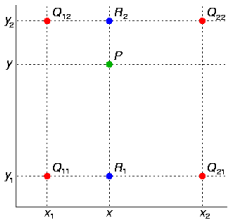
Dimana $\mu = \frac{x_i - x_0}{x_1 - x_0}$
Contoh:
Hitung nilai interpolasi titik nilai interpolasi titik(1,1) sampai(11,5) dengan jumlah titik $n = 5$
Jawab:
Bila $n = 5$ maka jarak setiap titik sela $u = \frac{1}{5}$ sehingga kenaikan $u = 0, \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}, 1$
Bila $n = 5$ maka jarak setiap titik sela adalah

$$d = \frac{x_1 - x_0}{5} = \frac{11 - 1}{5} = \frac{10}{5} = 2$$

sehingga kenaikan $x = 1,3,5,7,9,11$
Menggunakan rumus interpolasi linear
$y_i = y_0 \times (1 - u) + y_1 \times u$
maka titik sela dapat dihitung
Untuk $x_0 = 1$, maka $y_0 = 1 \times (1 - 0) + 5 \times 0 = 1 \Rightarrow$koordinat titik(1,1)
Untuk $x_1 = 3$, maka $y_1 = 1 \times \left(1 - \frac{1}{5}\right) + 5 \times \frac{1}{5} = 1\frac{4}{5} \Rightarrow$koordinat titik(3,1.8)
Untuk $x_2 = 5$, maka $y_2 = 1 \times \left(1 - \frac{2}{5}\right) + 5 \times \frac{2}{5} = 2\frac{3}{5} \Rightarrow$koordinat titik(5,2.6)
Untuk $x_3 = 7$, maka $y_3 = 1 \times \left(1 - \frac{3}{5}\right) + 5 \times \frac{3}{5} = 3\frac{2}{5} \Rightarrow$koordinat titik(7,3.4)
Untuk $x_4 = 9$, maka $y_3 = 1 \times \left(1 - \frac{4}{5}\right) + 5 \times \frac{4}{5} = 4\frac{1}{5} \Rightarrow$koordinat titik(9,4.2)
Untuk $x_5 = 11$, maka $y_3 = 1 \times (1 - 1) + 5 \times 1 = 5 \Rightarrow$koordinat titik(11,5)



Interpolasi Bilinear



Merupakan interpolasi linear pada data 2D
Dilakukan dengan cara interpolasi linear ke arah X kemudian interpolasi linear ke arah Y
Terdapat dua titik merah kemudian diinterpolasi linear terhadap X menghasilkan dua titik biru
Dua titik biru diinterpolasi linear terhadap Y menghasilkan titik hijau sebagai hasil interpolasi bilinear

Interpolasi Cosine

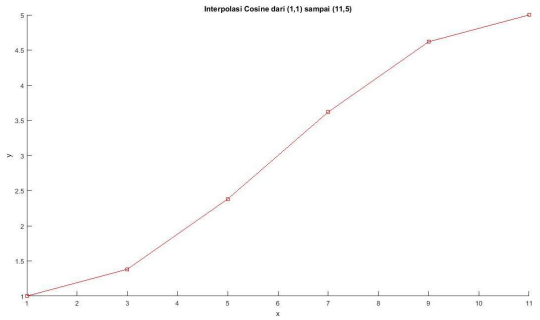
Menggunakan fungsi cosine untuk melakukan interpolasi
Bila terdapat dua titik yang akan diinterpolasi yaitu:

$$(x_0, y_0) dan (x_1, y_1)$$

Bila jarak tersebut dinormalisasi menjadi 1
Dan jarak titik sela (x_i, y_i) dengan titik awal adalah μ maka:

$$y_i = y_0 \times \left(1 - \frac{1 - \cos(\mu\pi)}{2}\right) + y_1 \times \left(\frac{1 - \cos(\mu\pi)}{2}\right)$$

Dimana $\mu = \frac{x_i - x_0}{x_1 - x_0}$
Contoh:
Hitung
Nilai interpolasi titik(1,1) sampai (11,5) dengan jumlah titik n= 5
Jawab:
Bila $n = 5$ maka jarak setiap titik sela $u = \frac{1}{5}$ sehingga kenaikan $\mu = 0, \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}, 1$
Bila $n = 5$ maka jarak setiap titik sela adalah $d = \frac{x_1 - x_0}{n} = \frac{11-1}{5} = \frac{10}{5} = 2$
sehingga kenaikan $x = 1,3,5,7,9,11$
Menggunakan rumus interpolasi cosine $y_i = y_0 \times \left(1 - \frac{1-\cos(\mu\pi)}{2}\right) + y_1 \times \left(\frac{1-\cos(\mu\pi)}{2}\right)$
Maka titik sela dapat dihitung
Untuk $x_0 = 1$, maka $y0$=1⇒koordinattitik(1,1)
Untuk $x_1 = 3$, maka $y_1 = 1{,}382 \Rightarrow$koordinat titik(3,1.382)
Untuk $x_2 = 5$, maka $y_2 = 2{,}382 \Rightarrow$koordinat titik(5,2.382)
Untuk $x_3 = 7$, maka $y_3 = 3{,}618 \Rightarrow$koordinat titik(7,3.618)
Untuk $x_4 = 9$, maka $y_4 = 4{,}618 \Rightarrow$koordinat titik(9,4.618)
Untuk $x_5 = 11$, maka $y_5 = 5 \Rightarrow$koordinat titik(11,5)



Interpolasi Cubic

Menggunakan fungsi pangkat tiga / kubik untuk melakukan interpolasi
Memerlukan 2 titik tambahan di ujung 2 titik utama untuk interpolasi
Bila terdapat 4 titik yang akan diinterpolasi yaitu

$$(x_0, y_0), (x_1, y_1), (x_2, y_2), (x_3, y_3)$$

Bila jarak tersebut dinormalisasi menjadi 1
Dan jarak titik awal $(x0, y0)$ sampai titik sela (x_i, y_i) adalah u
Dari dua titik tersebut maka persamaannya

$$y_i = au^3 + bu^2 + cu + d$$

Dimana:
$a = y_3 - y_2 - y_0 + y_1$
$b = 2y_0 - 2y_1 - y_3 + y_2$
$c = y_2 - y_0$
$d = y_1$
Contoh:
Hitung
Nilai interpolasi titik (-5,5),(5,5),(1,1), (11,5), dan (15,0) dengan jumlah titik n= 5
Jawab:
Bila $n = 5$ maka jarak setiap titik sela $u = \frac{1}{5}$ sehingga kenaikan $\mu = 0, \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}, 1$
Bila $n = 5$ maka jarak setiap titik sela adalah $d = \frac{x_1 - x_0}{n} = \frac{11-1}{5} = \frac{10}{5} = 2$ sehingga kenaikan $x = 1,3,5,7,9,11$

Menggunakan rumus interpolasi cubic

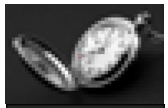
$$y_i = au^3 + bu^2 + cu + d$$

Maka titik sela dapat dihitung:
Untuk $x_0 = 1$, maka $y_0 = 1 \Rightarrow$koordinattitik(1,1)
Untuk $x_1 = 3$, maka $y_1 = 1.448 \Rightarrow$koordinat titik(3,1.448)
Untuk $x_2 = 5$, maka $y_2 = 2{,}504 \Rightarrow$koordinat titik(5,2.504)
Untuk $x_3 = 7$, maka $y_3 = 3{,}736 \Rightarrow$koordinat titik(7,3.736)
Untuk $x_4 = 9$, maka $y_4 = 4{,}712 \Rightarrow$koordinat titik(9,4.712)
Untuk $x_5 = 11$, maka $y_5 = 5 \Rightarrow$koordinat titik(11,5)

Interpolasi Bicubic dll.

Penerapan Interpolasi

Nearest Neighbor



Linear

Bilinear

Bicubic

Pembahasan UK

Pengantar Grafika Komputer

1. a. Jelaskan secara singkat dan jelas tentang sejarah perkembangan grafika komputer sampai saat ini .

Jawab:

Tahun 1950: The Whirlwind Computer di MIT memiliki Cathode Ray Tube (CRT) untuk menampilkan keluaran grafis.
Tahun 1963: Ivan Sutherland dalam tesisnya memperkenalkan konsep grafika komputer interaktif Sketchpad yang menjadi dasar konsep standar grafika saat ini.
Tahun 1964: Munculnya sistem CAD/CAM di General Motor.
Tahun 1970-an: Perkembangan perangkat keras grafika masih lambat karena mahal.
Awal 1980-an: Teknologi microchip memungkinkan arsitektur peraga raster/bitmap. Muncul usaha standarisasi seperti Core, CGM, CGI.
Pertengahan 1980-an: Muncul kelas komputer grafik workstation dengan kemampuan grafis tinggi seperti produk HP, Apollo, DEC, Xerox.
Akhir 1980-an: Kebutuhan superkomputer untuk komputasi grafis intensif. Muncul GUI seperti Macintosh, Windows. X Windows untuk lingkungan grafis terdistribusi.
Tahun 1990-an: Perkembangan teknologi visualisasi interaktif, artistic rendering, image based rendering.
Tahun 2000-an: Teknologi perangkat keras untuk real-time photorealistic rendering.

b. Sebutkan contoh penggunaan komputer grafik dalam bidang entertainment

Film dan Animasi:
Menciptakan karakter dan lingkungan 3D.
Menghasilkan efek visual yang realistis (ledakan, air, api, dll.).
Menambahkan detail dan tekstur pada objek.
Mengedit dan menggabungkan video.
Video Games:
Merancang dunia game, karakter, dan objek.
Mensimulasikan fisika dan gerakan.
Merender grafis real-time untuk gameplay yang interaktif.
Virtual Reality (VR):
Menciptakan lingkungan virtual yang imersif.
Merender grafis real-time untuk responsif terhadap gerakan pengguna.
Augmented Reality (AR):
Menambahkan elemen digital ke dunia nyata.
Digunakan dalam game, aplikasi hiburan, dan filter media sosial

OpenGL

2. Apa yang anda ketahui tentang OpenGL, dan sebutkan perintah yang ada dalam OpenGL untuk menampilkan segiempat di layar.

OpenGL (Open Graphics Library) adalah sebuah spesifikasi standar lintas platform yang mendefinisikan API untuk rendering grafis 2D dan 3D. API ini digunakan untuk berinteraksi dengan GPU (Graphics Processing Unit) untuk mencapai rendering hardware-accelerated.
Keuntungan menggunakan OpenGL:

- Independensi Perangkat: Berjalan di berbagai macam perangkat keras grafis.
- Independensi Platform: Berjalan di berbagai sistem operasi seperti Windows, Linux, dan macOS.
- Abstraksi: Menyediakan berbagai level abstraksi melalui library GL, GLU, dan GLUT.
- Open Source: Memiliki kode sumber terbuka yang dapat diakses dan dimodifikasi.
- Dukungan Client-Server: Mendukung protokol client-server untuk rendering jarak jauh.

```
#include <GL/glut.h>

void display() {
  glClear(GL_COLOR_BUFFER_BIT);

  glBegin(GL_QUADS);
    glVertex2f(-0.5f, -0.5f); // Vertex kiri bawah
    glVertex2f( 0.5f, -0.5f); // Vertex kanan bawah
    glVertex2f( 0.5f,  0.5f); // Vertex kanan atas
    glVertex2f(-0.5f,  0.5f); // Vertex kiri atas
  glEnd();

  glFlush();
}
```

```
int main(int argc, char** argv) {
  glutInit(&argc, argv);
  glutCreateWindow("Segiempat");
  glutDisplayFunc(display);
  glutMainLoop();
  return 0;
}
```

Keterangan:
glClear(GL_COLOR_BUFFER_BIT);: Membersihkan layar.
glBegin(GL_QUADS);: Menandai awal dari definisi primitif GL_QUADS.
glVertex2f(...): Mendefinisikan setiap vertex (sudut) dari segiempat.
glEnd();: Menandai akhir dari definisi primitif.
glFlush();: Memastikan semua perintah OpenGL dijalankan.

Algoritma Pembentukan Garis

3. Diketahui : Dua titik pembentukan garis yaitu A(1,2) dan titik akhir B(5,5)
Buatlah tabel perhitungan dan gambarkan grafiknya untuk titik-titik yang dihasilkan dengan menggunakan algoritma pembangkitan garis Bressenham
Jawab:
Calculate dx and dy:
dx = abs(x1 - x0)
dx = abs(5 - 1)
dx = 4
dy = abs(y1 - y0)
dy = abs(5 - 2)
dy = 3

Since dx > dy, calculate initial pk as 2*dy - dx
pk = 2*3 - 4
pk = 2

Step k=0: Given pk (2) >= 0
Calculate new pk as pk + 2*dy - 2*dx and new x as x + sx, new y as y + sy
new pk = 2 + 2*3 - 2*4
new pk = 0
new x = 1 + 1, new y = 2 + 1
new x = 2, new y = 3

Step k=1: Given pk (0) >= 0
Calculate new pk as pk + 2*dy - 2*dx and new x as x + sx, new y as y + sy
new pk = 0 + 2*3 - 2*4
new pk = -2
new x = 2 + 1, new y = 3 + 1
new x = 3, new y = 4

Step k=2: Given pk (-2) < 0
Calculate new pk as pk + 2*dy and new x as x + sx
new pk = -2 + 2*3
new pk = 4
new x = 3 + 1
new x = 4

Step k=3: Given pk (4) >= 0
Calculate new pk as pk + 2*dy - 2*dx and new x as x + sx, new y as y + sy
new pk = 4 + 2*3 - 2*4
new pk = 2
new x = 4 + 1, new y = 4 + 1
new x = 5, new y = 5

Table of Calculated Points:

k	p_k	(x_k+1, y_k+1)
0	0	(2, 3)
1	-2	(3, 4)
2	4	(4, 4)
3	2	(5, 5)

UTS (2023)

1. Algoritma Raster Dasar untuk Grafik 2D

Diketahui: Dua titik pembentukan garis yaitu A(1,2) dan titik akhir B(6,4). Buatlah tabel perhitungan dan gambarkan grafiknya untuk titik-titik yang dihasilkan dengan menggunakan algoritma pembangkitan garis DDA

$dx = xe - x0 = 6 - 1 = 5$
$dy = ye - y0 = 4 - 2 = 2$
$abs(dx) = 5$
$abs(dy) = 2$
$abs(dx) > abs(dy) == 5 > 2\ (true)$
$step = 5$

k	x	y	(x,y) bulat
	1	2	(1, 2)
0	2	2.4	(2, 2)
1	3	2.8	(3, 3)
2	4	3.2	(4, 3)
3	5	3.6	(5, 4)
4	6	4	(6, 4)

2. Transformasi dan Viewing 3D

a. Jelaskan dan beri contoh translasi, rotasi dan scalling pada transformasi 2D.

- **Definisi:** Menggeser objek ke posisi baru tanpa mengubah u kuran atau orientasinya.
- **Rumus:**

x' = x + tx
y' = y + ty
Dimana:
(x, y) adalah koordinat titik awal.
(x', y') adalah koordinat titik setelah translasi.
tx adalah pergeseran pada sumbu x.
ty adalah pergeseran pada sumbu y.

- **Contoh:**

Misalkan kita ingin menggeser segitiga dengan titik-titik (1, 1), (3, 1), dan (2, 3) sejauh 2 unit ke kanan dan 1 unit ke atas. Maka tx = 2 dan ty =
1. Koordinat baru segitiga setelah translasi adalah:
(1+2, 1+1) = (3, 2)
(3+2, 1+1) = (5, 2)
(2+2, 3+1) = (4, 4)

Rotasi

- **Definisi:** Memutar objek di sekitar titik tertentu (titik rotasi) d engan sudut tertentu.
- **Rumus:**

x' = x*cos(θ) - y*sin(θ)
y' = x*sin(θ) + y*cos(θ)
Dimana:
(x, y) adalah koordinat titik awal.
(x', y') adalah koordinat titik setelah rotasi.
θ adalah sudut rotasi (dalam radian).

- **Contoh:**

Misalkan kita ingin merotasi persegi dengan titik-titik (1, 1), (3, 1), (3, 3), dan (1, 3) sebesar 90 derajat searah jarum jam di sekitar titik asal (0, 0). Maka θ =- π/2. Koordinat baru persegi setelah rotasi adalah:
(1*cos(-π/2)* - *1*sin(-π/2), 1*sin(-π/2)* + *1*cos(-π/2)) = (1, -1)
(3*cos(-π/2)* - *1*sin(-π/2), 3*sin(-π/2)* + *1*cos(-π/2)) = (1, -3)
(3*cos(-π/2)* - *3*sin(-π/2), 3*sin(-π/2)* + *3*cos(-π/2)) = (3, -3)
(1*cos(-π/2)* - *3*sin(-π/2), 1*sin(-π/2)* + *3*cos(-π/2)) = (3, -1)

Scaling

- **Definisi:** Mengubah ukuran objek, membuatnya lebih besar atau lebih kecil.
- **Rumus:**

x' = x * Sx

y' = y * Sy
Dimana:
(x, y) adalah koordinat titik awal.
(x', y') adalah koordinat titik setelah scaling.
Sx adalah faktor skala pada sumbu x.
Sy adalah faktor skala pada sumbu y.

- **Contoh:**
  Misalkan kita ingin memperbesar lingkaran dengan pusat di (2, 2) dan jari-jari 1 sebesar 2 kali lipat. Maka Sx = 2 dan Sy = 2. Jari-jari baru lingkaran adalah 1 * 2 = 2.

b. Tentukan matriks hasil transformasi gabungan suatu objek (M) dengan urutannya sebagai berikut: Translasikan titik origin dengan $(P_x, P_y, P_z) = (10,20,30)$, lalu dirotasikan sekitar sumbu x dengan sudut $30°$ dan kemudian ditranslasikan objek tersebut ke translasi sebelumnya. Adapun matriksnya adalah sebagai berikut:

Translation

$$T(-P) = \begin{bmatrix} 1 & 0 & 0 & -P_x \\ 0 & 1 & 0 & -P_y \\ 0 & 0 & 1 & -P_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation Around x

$$R(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Translation

$$T(P) = \begin{bmatrix} 1 & 0 & 0 & P_x \\ 0 & 1 & 0 & P_y \\ 0 & 0 & 1 & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M = T(P)R(\theta)T(-P)$$

Jawab:

Misalkan titik origin $(0,0,0)$

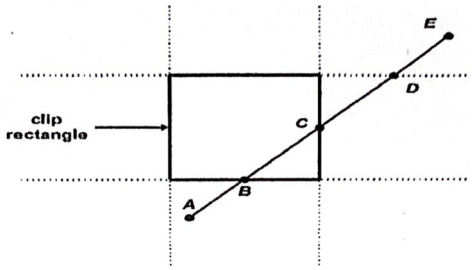
- Transformasi $(P_x, P_y, P_z) = (10,20,30)$
  $P' = T(P) = (0 + 10,0 + 20,0 + 30)$
  $P' = (10,20,30)$
- Rotasi terhadap sumbu x dengan sudut $30°$
  $P'' = R(\theta)$
  $x'' = x' = 10$
  $y'' = y'\cos(\theta) - zsin(\theta)$
  $y'' = 20\cos(30) - 30\sin(30)$
  $y'' = 20 \times \frac{\sqrt{3}}{2} - 30 \times \frac{1}{2}$
  $y'' = 10\sqrt{3} - 15$
  $z'' = y'sin(\theta) + z'cos(\theta)$
  $z'' = 20\sin(30) + 30\cos(30)$
  $z'' = 20 \times \frac{1}{2} + 30 \times \frac{\sqrt{3}}{2}$
  $z'' = 10 + 15\sqrt{3}$
  $P'' = (10,10\sqrt{3} - 15,10 + 15\sqrt{3})$
- Transformasi kembali $(P_x, P_y, P_z) = (10,20,30)$
  $P''' = T(-P)$
  $P''' = (10 - 10,10\sqrt{3} - 15 - 20,10 + 15\sqrt{3} - 30)$
  $P''' = (0,10\sqrt{3} - 35,15\sqrt{3} - 20)$

3. Algoritma Clipping

a. Apa yang dinamakan algoritma Clipping?

Algoritma clipping adalah prosedur yang digunakan untuk menentukan bagian-bagian gambar atau objek yang berada di dalam atau di luar area tertentu yang disebut jendela clipping (clipping window). Dengan kata lain, algoritma ini "memotong" bagian gambar yang tidak diinginkan, sehingga hanya bagian yang berada di dalam jendela clipping yang ditampilkan. Jendela clipping biasanya berbentuk persegi panjang, tetapi bisa juga berupa bentuk lain seperti poligon atau kurva.

b. Tentukan hasil clipping dari gambar berikut dengan menggunakan algoritma Cohen-Sutherland



Algoritma Cohen-Sutherland menggunakan kode 4-bit untuk mengklasifikasikan posisi titik terhadap jendela clipping. Beri kut langkah-langkahnya:

1) **Tentukan kode 4-bit untuk setiap titik:**

Bit 1: Kiri: bernilai 1 jika x < x_min
Bit 2: Kanan: bernilai 1 jika x > x_max
Bit 3: Bawah: bernilai 1 jika y < y_min
Bit 4: Atas: bernilai 1 jika y > y_max

2) **Analisis kode 4-bit untuk setiap garis:**

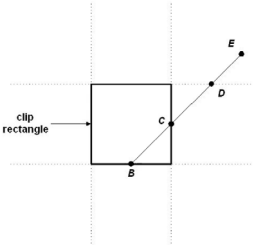
- **Jika kedua titik memiliki kode 0000:** Garis berada sepenuhnya di dalam jendela clipping, tidak perlu clipping.
- **Jika hasil AND bitwise dari kode kedua titik tidak nol:** Garis ber ada sepenuhnya di luar jendela clipping, garis dibuang.
- **Jika tidak ada kondisi di atas terpenuhi:** Garis memotong jendel a clipping, perlu clipping.

3) **Lakukan clipping pada garis yang memotong jendela clipping:**

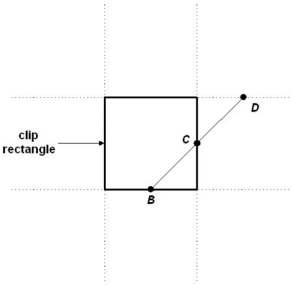
- Tentukan titik potong garis dengan batas jendela clipping.
- Ganti titik ujung garis di luar jendela clipping dengan titik potong t ersebut.

Hasil Clipping:

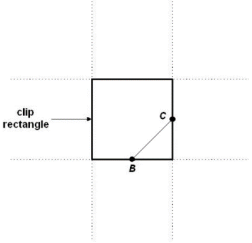
Berdasarkan algoritma Cohen-Sutherland dan gambar yang diberikan:



- Garis AB: Dihapus, karena kedua titik ujung berada di luar clipping window.



- Garis DE: Dihapus, karena kedua titik ujung berada di luar clipping window.
- Garis BC: Tidak dihapus, karena kedua titik ujung berada di dalam clipping window.



- Garis CD: Dihapus, karena kedua titik ujung berada di luar clipping window.