CmpE240 April 4, 22 Part II - 1

April 4 (Monday).

Topics: 1:30 GE focused on Shadow Computation.

Example: Given

1. x_w-y_w-z_w world coordinate Right Hand System. $\vec{P_i} + \lambda(\vec{P_s} - \vec{P_i})$

$\vec{R}$

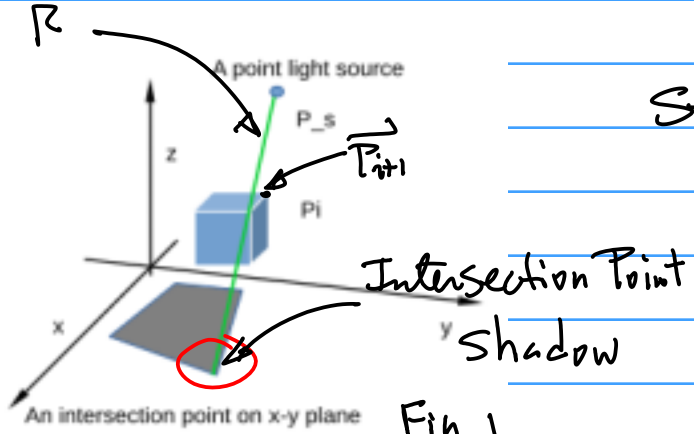


Fig. 1

2. A Given Cube $\{\vec{P_i}(x_i, y_i, z_i) \mid i = 1, 2, \ldots, N\}$

Counter Clockwise

3. A given Point light Source $\vec{P_s}(x_s, y_s, z_s)$

4. Generate Ray Equation / Ray Cast x_w-y_w plane, Produces Shadow if Blocked by the Cube.

References:

2021F-101b-notes-cmpe240-2021-12-1.pdf

Step 1. Generate A Ray Cast / Equation

$$\vec{R} = \vec{P_i} + \lambda(\vec{P_s} - \vec{P_i}) \quad \ldots (1)$$

Starting from $\vec{P_s}$, Passing through $\vec{P_i}$

$$\vec{d}(x, y) = \vec{P_s} - \vec{P_i} \quad \ldots (1b)$$

Intersection Point: $\vec{P_i}$

Between Ray $\vec{R}(x, y)$, and x_w-y_w plane.

Step 2. To find the intersection Point on x_w-y_w plane

Define a plane equation.

a. Define A Normal Vector

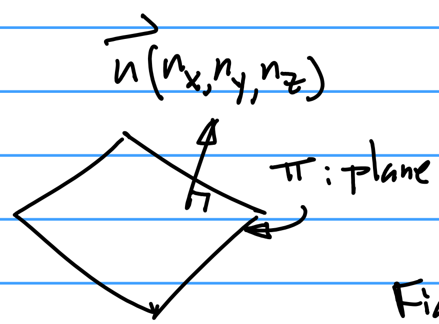


Fig. 2

use the normal vector to define the plane π.

b. Form a vector (on a plane) from $\vec{P_i}$ and $\vec{P_{i+1}}$, $(\vec{P_{i+1}} - \vec{P_i})$ a line

$$\vec{n} \cdot (\vec{P_{i+1}} - \vec{P_i}) = 0 \quad \ldots (2)$$

Take

$\vec{v}(v_x, v_y, v_z)$, $\vec{a}(a_x, a_y, a_z)$

to Replace $\vec{P_i}$, $\vec{P_{i+1}}$

Assume $\vec{a}(a_x, a_y, a_z)$ is a Known Vector; And $\vec{v}(v_x, v_y, v_z)$ is unknown, But an arbitrary Point on the plane π.

Hence, Eqn (2) becomes

$$\vec{n} \cdot (\vec{v} - \vec{a}) = 0 \quad \ldots (2^*)$$

Now, find the intersection point defined By the Ray Eqn (1). In order to that, we will need to find λ.

Since, the intersection point $\vec{P_i}$ is the Common Point By the Ray and the plane π. we have

$$\begin{cases} \vec{R} = \vec{P_i} + \lambda(\vec{P_s} - \vec{P_i}) & \ldots (3a) \\ \vec{n} \cdot (\vec{v} - \vec{a}) = 0 & \ldots (3b) \end{cases}$$

$\vec{n}(n_x, n_y, n_z)$, Normal Vector has to be Known,

$\vec{a}(a_x, a_y, a_z)$ is Known on π.

Starting from the plane Eqn (3b).

$$\vec{n} \cdot (\vec{v} - \vec{a}) = 0$$

where $\vec{v} = \vec{R}$, e.g.

$$\vec{n} \cdot (\vec{v} - \vec{a}) \Big|_{\vec{v} = \vec{R}} = 0 \quad \ldots (4)$$

$$\vec{n} \cdot (\vec{R} - \vec{a}) \Big|_{\vec{R} = \vec{P_i} + \lambda(\vec{P_s} - \vec{P_i})} = 0$$

$$\vec{n} \cdot (\vec{P_i} + \lambda(\vec{P_s} - \vec{P_i}) - \vec{a}) = 0$$

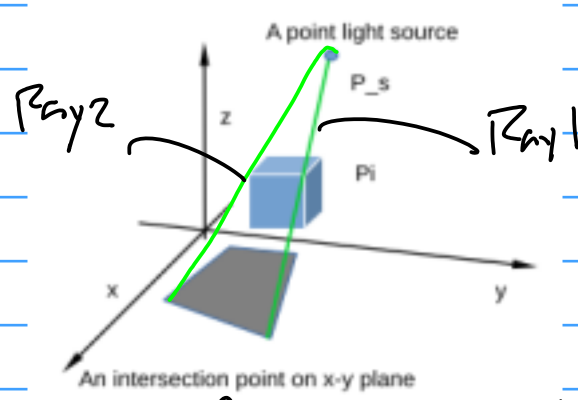
$$\vec{n} \cdot \vec{P_i} + \lambda \vec{n} \cdot (\vec{P_s} - \vec{P_i}) - \vec{n} \cdot \vec{a} = 0$$

$$\lambda \vec{n} \cdot (\vec{P_s} - \vec{P_i}) = \vec{n} \cdot \vec{a} - \vec{n} \cdot \vec{P_i}$$

$$\lambda = \frac{\vec{n} \cdot \vec{a} - \vec{n} \cdot \vec{P_i}}{\vec{n} \cdot (\vec{P_s} - \vec{P_i})}$$

$$= \frac{\vec{n} \cdot (\vec{a} - \vec{P_i})}{\vec{n} \cdot (\vec{P_s} - \vec{P_i})} \quad \ldots (5)$$

Note λ is <u>NOT</u> the intersection Pt. it allows us to use Ray Eqn (1) to find the intersection.



use Eqn (5) to find more than one intersection

April 7th (Wed)

Example: Ref (see pp.1. github LectureNotes, 2021F-101b-~)

Given $\vec{P_s}(-20,110,200)$, A vertex of a given Cube $\vec{P_i}(100,100,110)$. Find the intersection pt to Draw Shadow.

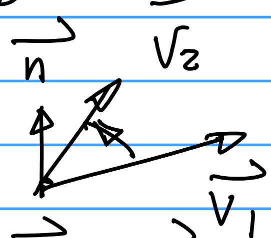
Sol First, Ray Equation, Eqn(1):

$$\vec{R} = \vec{P_i} + \lambda(\vec{P_s} - \vec{P_i}) \quad \ldots (1)$$

then, plane Eqn for X_w-Y_w plane

$$\vec{n}\cdot(\vec{v} - \vec{a}) = 0$$

Find $\vec{n}$,



$$\vec{V_1} \times \vec{V_2} = \vec{n} \quad \ldots (2)$$

Let $\vec{V_1} = x_w\vec{i}$, $\vec{V_2} = y_w\vec{j}$

$$\vec{n} = x_w\vec{i} \times y_w\vec{j} = (0,0,1)$$

Then, 2nd find λ for the Ray Equation, from Eqn.

$$\lambda = \frac{\vec{n}\cdot(\vec{a} - \vec{P_i})}{\vec{n}\cdot(\vec{P_s} - \vec{P_i})} \quad \ldots (5)$$

where $\vec{a} = (0,0,0)$, Hence,

$$\lambda = \frac{-\vec{n}\cdot\vec{P_i}}{\vec{n}\cdot(\vec{P_s} - \vec{P_i})}$$

Now, hand Calculation, also for Coding

$$\lambda = \frac{-\vec{n}\cdot\vec{P_i}}{\vec{n}\cdot(\vec{P_s} - \vec{P_i})} = -\frac{n_x x_i + n_y y_i + n_z z_i}{n_x(x_s - x_i) + n_y(y_s - y_i) + n_z(z_s - z_i)}$$

From the given condition $\vec{n}(0,0,1)$, Therefore

$$\lambda = -\frac{\cancel{n_z} z_i}{\cancel{n_z}(z_s - z_i)} = -\frac{z_i}{z_s - z_i} = -\frac{110}{200-110} = -\frac{110}{90} = -11/9$$

Now, back to the Ray Equation

$$\vec{R} = \vec{P_i} + \lambda(\vec{P_s} - \vec{P_i})$$

$$\vec{R} = (100,100,110) - \frac{11}{9}(-20-100, 110-100, 200-110)$$

$$= (100,100,110) - \frac{11}{9}(-120, 10, 90)$$

$$= \left(\frac{1100\times 120}{9}, -\left(100-\frac{110}{9}\right), 110-110\right)$$

$$= \left(\frac{1100\times 120}{9}, \frac{110}{9}-100, 0\right) = (246.7, 87.8, 0)$$

please finish this calculation.

Now, Coding Part. Same Code on github.

2018F-116-11diffuse20181114.cpp

Note:

a. Define Normal vector $\vec{n}$ for X_w-Y_w plane

```
world.X[47] = 0; world.Y[47] = 0; world.Z[47] = 1; // normal vector for x-y plane
```

b. Note the "typedef struct" for Defining 3D points.

```
typedef struct {
    float X[UpperBD], Y[UpperBD], Z[UpperBD];
} pworld;
```

```
pworld  world;
pviewer viewer;
```

Now, λ Calculation. Eqn (5)

$$\lambda = \frac{\vec{n}\cdot(\vec{a}-\vec{P_i})}{\vec{n}\cdot(\vec{P_s}-\vec{P_i})} \quad \cdots (5)$$

$$= -\frac{n_x x_i + n_y y_i + n_z z_i}{n_x(x_s-x_i)+n_y(y_s-y_i)+n_z(z_s-z_i)}$$

```
//----------lambda for intersection pt on xw-yw plane----------*
float temp = (world.X[47]*(world.X[46]-world.X[45])
             +(world.Y[47]*(world.Y[46]-world.Y[45]))
             +(world.Z[47]*(world.Z[46]-world.Z[45]));
float lambda = temp / ((world.X[47]*(world.X[45]-world.X[7]))
                      +(world.Y[47]*(world.Y[45]-world.Y[7]))
                      +(world.Z[47]*(world.Z[45]-world.Z[7])));
float lambda_2 = temp / ((world.X[47]*(world.X[45]-world.X[6]))
                        +(world.Y[47]*(world.Y[45]-world.Y[6]))
                        +(world.Z[47]*(world.Z[45]-world.Z[6])));
```

Note, Substitute λ to Ray Equation to find the intersection point

```
//-----------ray equation to find intersection pts------------*
world.X[48] = world.X[45] + lambda*(world.X[45] - world.X[7]); // Intersection pt p7
world.Y[48] = world.Y[45] + lambda*(world.Y[45] - world.Y[7]); // Intersection pt p7
world.Z[48] = 0.0;

world.X[49] = world.X[45] + lambda_2*(world.X[45] - world.X[6]);  //intersection pt p6
world.Y[49] = world.Y[45] + lambda_2*(world.Y[45] - world.Y[6]);  //intersection pt p6
world.Z[49] = 0.0;
```

$\vec{R} = \vec{P_i} + \lambda(\vec{P_s} - \vec{P_i}) \rightarrow$

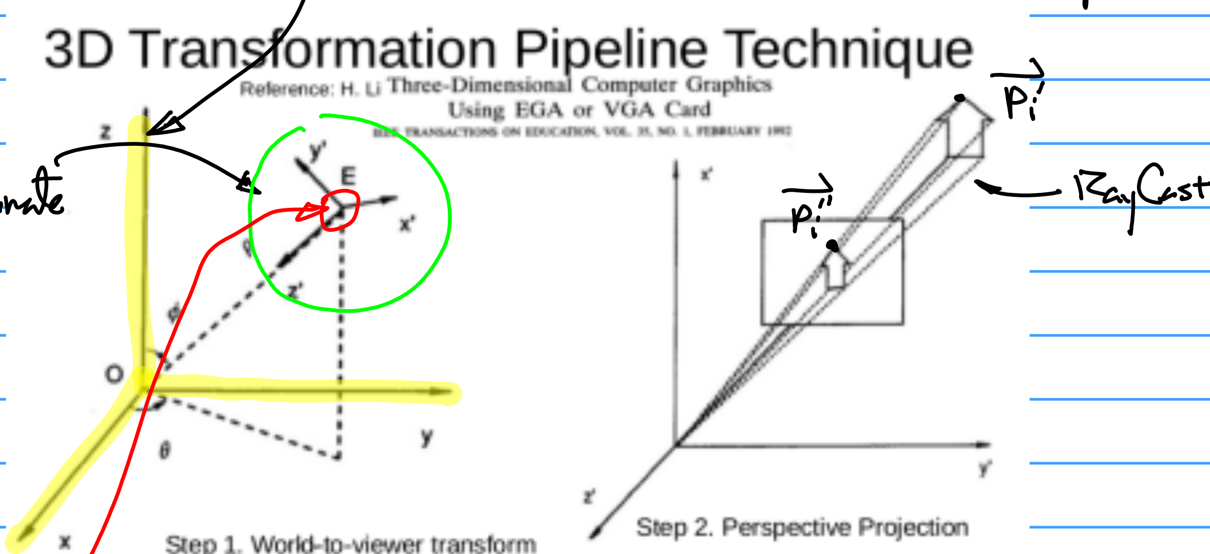
$$x = x_i + \lambda(x_s - x_i)$$
$$y = y_i + \lambda(y_s - y_i)$$
$$z = z_i + \lambda(z_s - z_i)$$

Assignment in-Class show & Tell. Implement Intersection Computation on LPC1769, "Show+Tell" Demo in Class. On April 11 (Monday),

To Be Able to Display 3D Graphics on 2D Display Devices. Let's Define Transformation Pipeline. 1. Define World-Coordinate System; Right Hand System $x_w-y_w-z_w$

2. Viewer Coordinate System $x_v-y_v-z_v$ Left-Hand System

3. Virtual Camera is located $\vec{E}$ (e_x, e_y, e_z)



$$T = \begin{bmatrix} -\sin\theta & \cos\theta & 0 & 0 \\ -\cos\phi\cos\theta & -\cos\phi\sin\theta & \sin\phi & 0 \\ -\sin\phi\cos\theta & -\sin\phi\cos\theta & -\cos\phi & \rho \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$x_p = x_e\left(\frac{D}{z_e}\right)$$
$$y_p = y_e\left(\frac{D}{z_e}\right)$$

Harry Li, Ph.D.

Example: Display Shadows on 2D Display Device.

Step 1. World To Viewer Transform.

$$T=\begin{bmatrix} -\sin\theta & \cos\theta & 0 & 0 \\ -\cos\phi\cos\theta & -\cos\phi\sin\theta & \sin\phi & 0 \\ -\sin\phi\cos\theta & -\sin\phi\cos\theta & -\cos\phi & \rho \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \cdots (1)$$

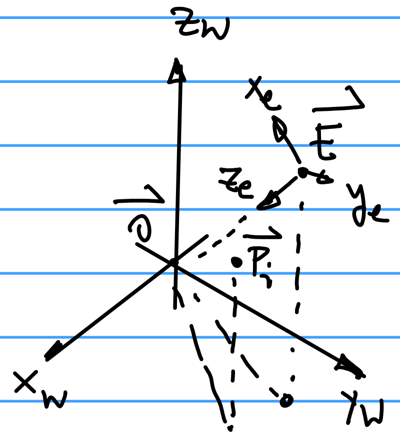


Fig. 1.

Everything is defined in the World Coordinate System x_w-y_w-z_w including a Virtual Camera.

$\vec{E}(x_e, y_e, z_e)$, x_e-y_e-z_e "Viewer" Coordinate System.

Given $\vec{P_i}(x_i, y_i, z_i)$ in x_w-y_w-z_w World Coordinate, Represent this point in x_e-y_e-z_e Coordinate System.

$$\vec{P_i'}: \begin{bmatrix} x_i' \\ y_i' \\ z_i' \\ 1 \end{bmatrix} = \begin{bmatrix} -\sin\theta & \cos\theta & 0 & 0 \\ -\cos\phi\cos\theta & -\cos\phi\sin\theta & \sin\phi & 0 \\ -\sin\phi\cos\theta & -\sin\phi\cos\theta & -\cos\phi & \rho \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix} \quad \cdots (1\text{-}b)$$

$\vec{P_i}$

Assume $\vec{E}(x_e, y_e, z_e) = (200, 200, 200)$

Physical meaning of Transformation Matrix T.

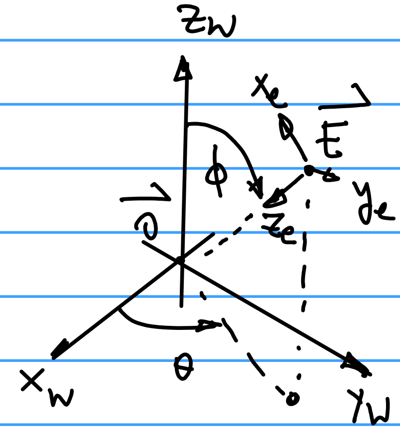


Fig. 2

θ: Angle from the dash line on x_w-y_w plane w.r.t positive x_w-axis

ϕ: Angle Between z_w & z_e.

ρ (rho): $\rho = \sqrt{x_e^2 + y_e^2 + z_e} \quad \cdots (2)$

distance from $\vec{E}$ to the origin $\vec{O}$ of x_w-y_w-z_w.

Suppose $\vec{E}(200, 200, 200)$ is given, Find $\cos\theta$, $\sin\theta$, $\cos\phi$, $\sin\phi$ for T-matrix.

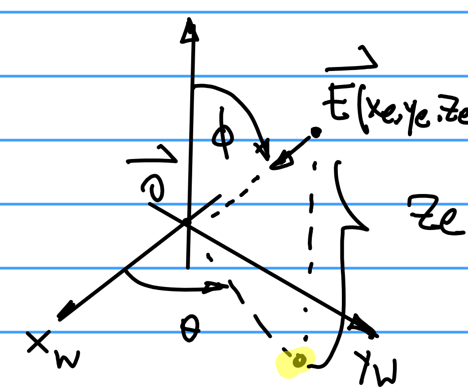
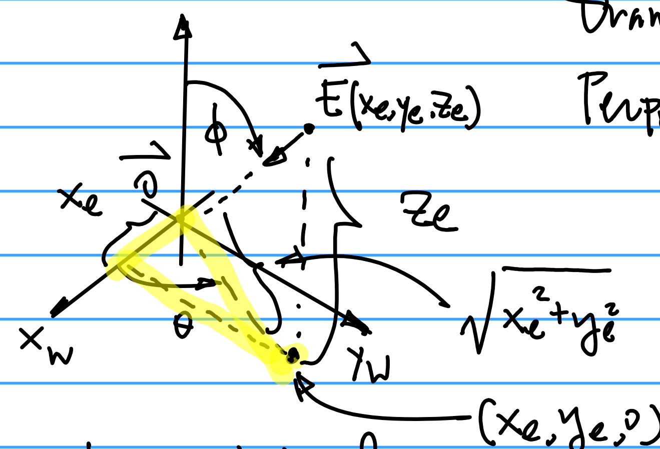


Fig. 3

Draw A Line Passing through $\vec{E}$ Perpendicular to X_w-Y_w $\Rightarrow$ Form an intersection point.

Draw A Line Passing through the intersection point on X_w-Y_w plane on the plane and Perpendicular to X_w-axis.

Fig.4

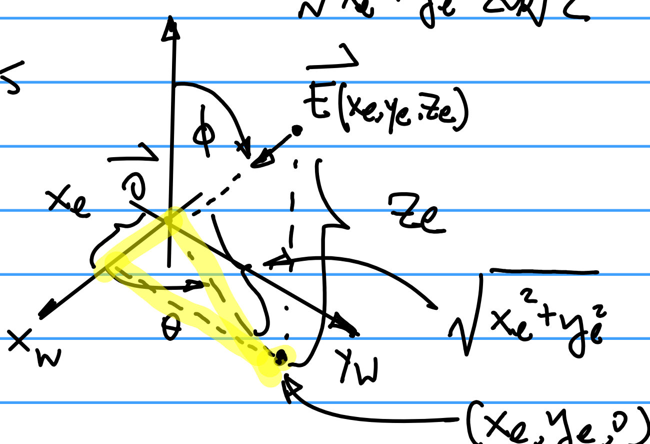


We can form a triangle on X_w-Y_w plane, as in Fig.4, hence

$$\cos\theta = \frac{x_e}{\sqrt{x_e^2+y_e^2}} = \frac{200}{200\sqrt{2}} = \frac{\sqrt{2}}{2}$$

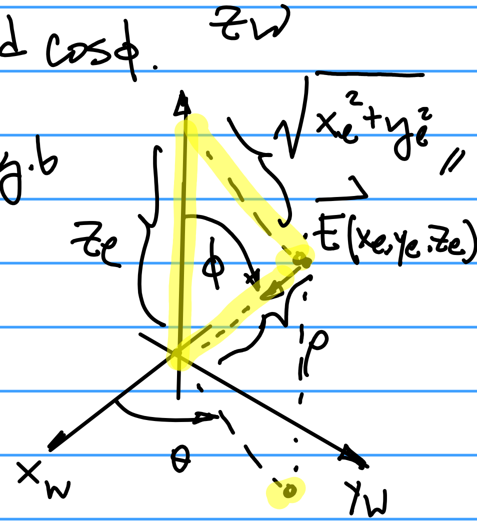
similarly, $\sin\theta = \dfrac{y_e}{\sqrt{x_e^2+y_e^2}} = \dfrac{200}{200\sqrt{2}} = \dfrac{\sqrt{2}}{2}$

Fig.5



Find $\cos\phi$.

Fig.6



Draw A Line Passing $\vec{E}(x_e, y_e, z_e)$ Perpendicular to Z_w-axis

$$\cos\phi = \frac{z_e}{\rho} = \frac{z_e}{\sqrt{x_e^2+y_e^2+z_e}} = \frac{200}{200\sqrt{3}} = \sqrt{3}/3$$

$$\sin\phi = \frac{\sqrt{x_e^2+y_e^2}}{\sqrt{x_e^2+y_e^2+z_e}} = \frac{200\sqrt{2}}{200\sqrt{3}} = \frac{\sqrt{2}\cdot\sqrt{3}}{3} = \frac{\sqrt{6}}{3}$$

Homework: Due April 18th (Monday)

1. Draw A world Coordinate System X_w-Y_w-Z_w axis, with X_w Red, Y_w Green, Z_w Blue

2. Draw A cube, size Length = 100, floats 10 unit Above $X_w - Y_w$ plane. in other word $\vec{P_i}(100,100,110)$;

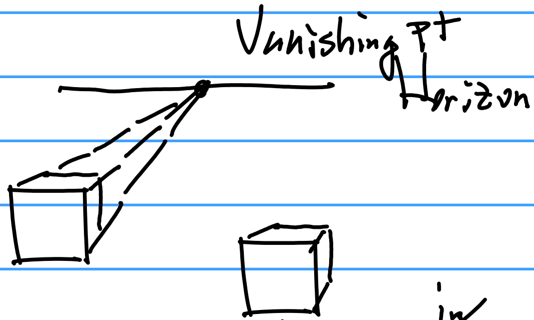
3. Draw a point light Source $\vec{P_s}(-20,110,200)$, And Ray Cast to Connect $\vec{P_s}$ to $\vec{P_i}$; use Green Color.

4. Compute the Shadow point $\vec{P_i'}$, Draw the intersection Point to link $\vec{P_s} - \vec{P_i} - \vec{P_i'}$

Note: You may want to Adjust the $\vec{P_s}$ Position, So this $\vec{P_i'}$ (Intersection Point) Can be Visible on Your Display

Step 2. Perspective Projection

$$\boxed{\begin{aligned} x_p &= x_e\left(\frac{D}{z_e}\right) \\ y_p &= y_e\left(\frac{D}{z_e}\right) \end{aligned}} \quad \ldots (3)$$



Eqn(3) project a point $\vec{P_i}(x_i, y_i, z_i)$ in $X_e - Y_e - Z_e$ Onto a 2D Display Device, like LCD.

April 13 (Wed)

Topics: 1° Perspective Projection

2° Diffuse Reflection

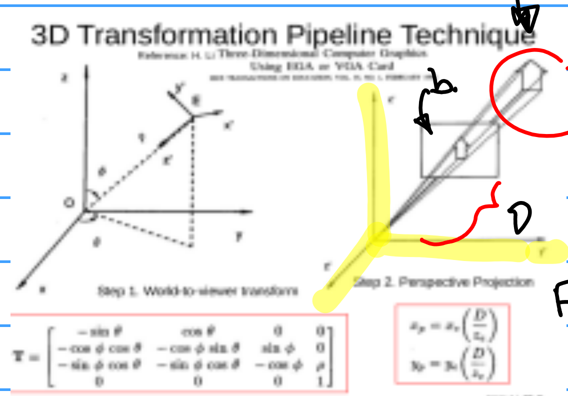


Fig.1

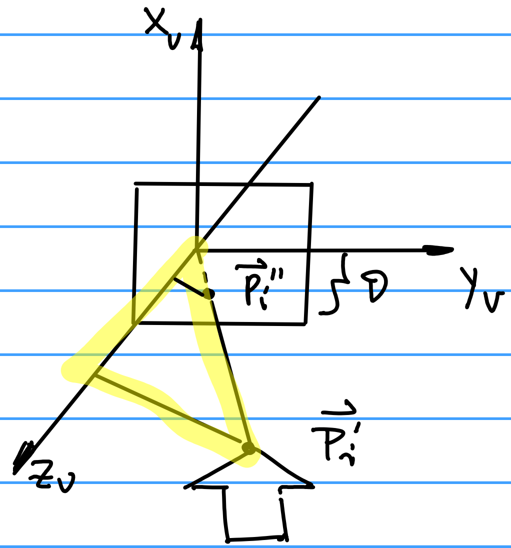
Note: $X_v - Y_v - Z_v$ is Left Hand System.

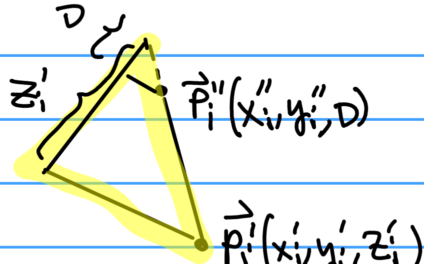
a. It is a 3D object

b. projection Plane, distance ~~to~~ the Viewer Coordinate System $(0,0,0) \rightarrow (x_e, y_e, z_e)$ in $X_w Y_w Z_w$. D is the distance.

c. Origin of $X_v - Y_v - Z_v$. $\rightarrow$ Camera Location, Camera is modeled as "pin-Hole" model.

Projection is formulated By using Similar Triangles in Fig.1





$$x_p = \frac{D}{z_e} x_e \quad \text{from Eqn (8).}$$

Or,

$$x_i'' = \frac{D}{z_i'} x_i'$$

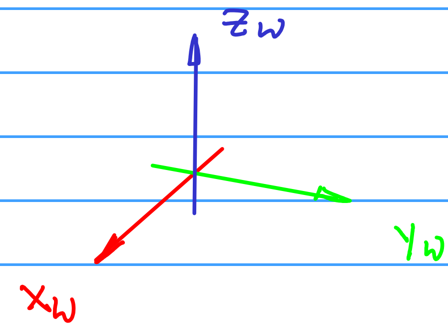
Similarly,

$$y_p = \frac{D}{z_e} y_e \quad \text{or}$$

$$y_i'' = \frac{D}{z_e} y_i'$$

Homework, Due A week from Today April 20th

1. Draw a World Coordinate System, X_w: Red, Y_w: Green, Z_w: Blue, Design the size (X_w, Y_w, Z_w, 50 units)



2. Design By Defining Dimension of a Cube.
   (Example: length = 100)
   $\vec{P}_i(x_i, y_i, z_i) = (100, 100, 110)$
   Elevate the cube By 10 units.
3. Draw the Cube on the LCD
4. Submission:
   a. Screen Capture of your XPRESSO Screen, which Shows your Name (Folder Name) And your Program (Partial) — 0.5 pt
   b. Take a photo of your display. With Entire Prototype System of your own — 0.5 pt.
5. Submission to CANVAS.

Note: You will need transform from a virtual coordinate System to physical coordinate

Consider Diffuse Reflection.

2018S-23-lec7-DiffuseReflection-v6-2018-4-25.pdf

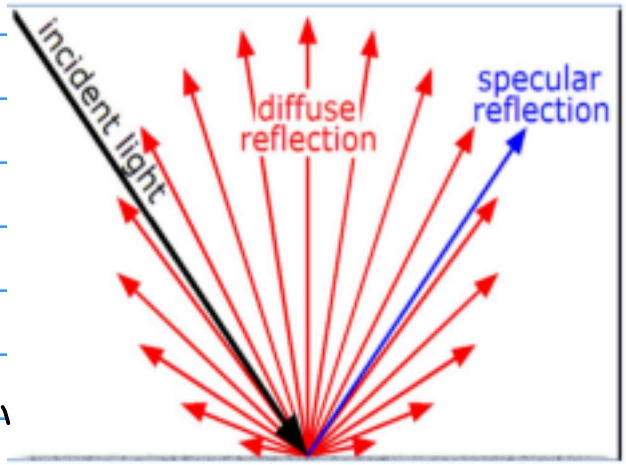


Fig. 1

https://en.wikipedia.org/wiki/Diffuse_reflectio

Definition: A Reflection from an object surface uniformly in all directions.

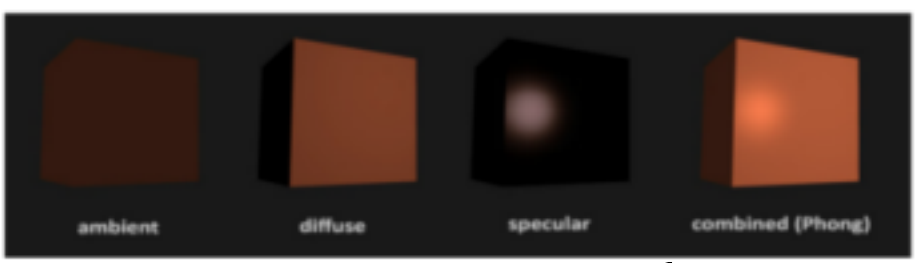


Fig2.

1. Define Reflectivity, A property of An object surface.

$\vec{K_r} = (k_r, k_g, k_b) = (r, g, b)$ ...(1)

$0 \leq r, g, b \leq 1$

Note: for a Black object,

$r=0, g=0, b=0$

for a green leaf.

$r \cong 0;\ g \neq 0, 0 < g;\ b \cong 0$

2. Color Space. R, G, B

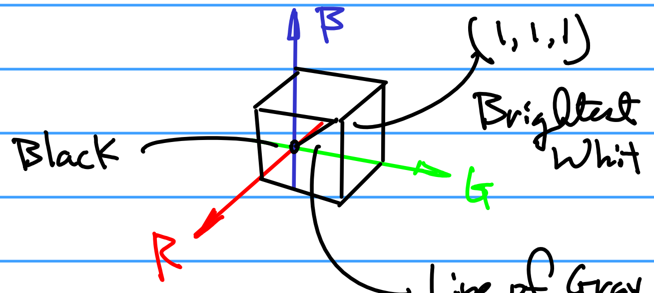
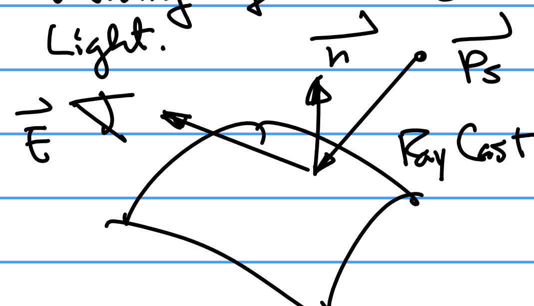


Fig3

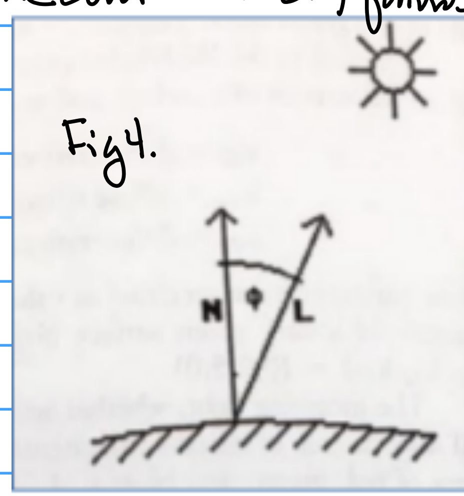
3. Viewing Angle v.s. Incident Light.



a. Perceived color is independent of viewing Angle.

b. Normal vector $\vec{N}$ and incident Light $\vec{L}$ ($\vec{R}$ Ray Cast) form An Angle ϕ, the color Intensity follows

Fig4.



$\cos\phi$ function.

Color Intensity.

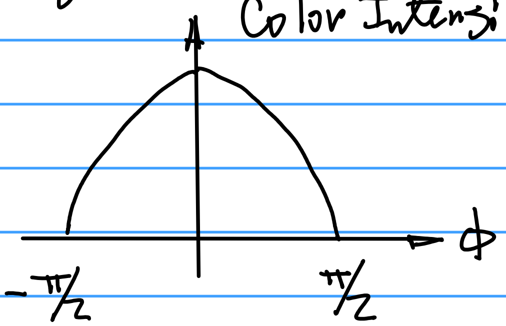


Fig 5.

$\phi = 0$, $\cos\phi = 1$, Strongest Reflection.
Highest Intensity.

$\phi = \pi/2$, $\cos\phi = 0$, No Reflection,
hence, no color.

4. Distance.

April 18 (Monday)

Topics: Continue on Diffuse Reflection.

Light Intensity. Satisfies the relationship Below.

$$\text{Intensity} \sim \frac{1}{\|\vec{r}\|^2} \quad \ldots (1)$$

$\vec{r}$ Ray Equation.

$$\vec{r} = \vec{P_i} + \lambda(\vec{P_s} - \vec{P_i})$$

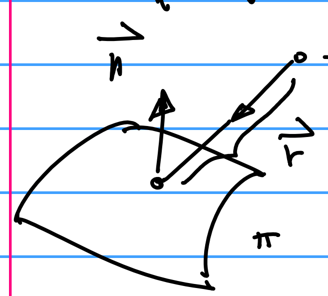


Fig.1

Example: Given $\vec{P_s}(x_s, y_s, z_s)$ And A Cube.

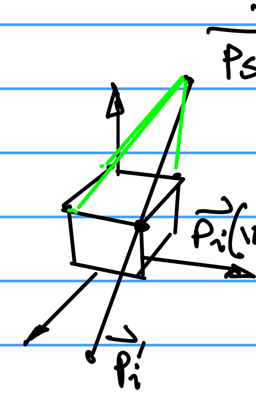


Fig.2

$\vec{P_i}$ shadow Point.

Find Diffuse Reflection at $\vec{P_i}(x_i, y_i, z_i)$

Sol: Formulation.

$$\vec{I}_{diff}(x, y, z) = \left(I_{diff,r}(x,y), I_{diff,g}(x,y), I_{diff,B}(x,y)\right)$$

$$\vec{K_d} = (K_r, K_g, K_b) \quad \ldots (2)$$

$$0 \leq K_r, K_g, K_b \leq 1$$

Note: For Simplicity, we will focus on one type of Reflectivity for Now, "r" red

$$\boxed{I_r = K_{d_r} \frac{\vec{n} \cdot \vec{r}}{\|\vec{n}\|\|\vec{r}\|} \frac{1}{\|\vec{r}\|^2}} \quad \ldots (3)$$

Reflectivity for "r"

$\cos\phi$. $\vec{n} \cdot \vec{r} = \|\vec{n}\|\|\vec{r}\| \cos\phi$

$$\|\vec{r}\|_2 = \sqrt{(x_s-x_i)^2+(y_s-y_i)^2+(z_s-z_i)^2} \quad \ldots (4)$$

$$\|\vec{r}\|_2^2 = (x_s-x_i)^2+(y_s-y_i)^2+(z_s-z_i)^2$$

Suppose $\vec{P_s}(-20,110,200)$, $\vec{P_i}(100,100,110)$

Assume: $K_{dr}=0.8$

Find Normal vector for the Cube Surface.

$\vec{n}$ Defined By Vector Cross Product $\vec{n}=\vec{A}\times\vec{B}$

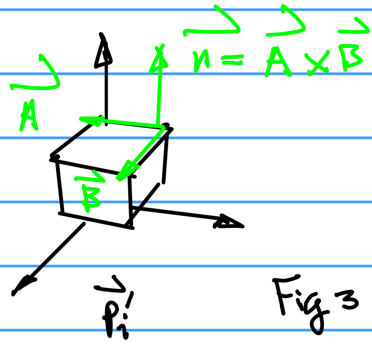


Fig 3

But the Surface of the Cube is in Parallel with X_w-Y_w plane.

$\therefore \vec{n}(0,0,1)$

Now, find

$$\cos\phi = \frac{\vec{n}\cdot\vec{r}}{\|\vec{n}\|\|\vec{r}\|}$$

Directional vector of $\vec{r}$

$$= \frac{(n_x,n_y,n_z)\cdot(x_i-x_s,\, y_i-y_s,\, z_i-z_s)}{\sqrt{n_x^2+n_y^2+n_z^2}\sqrt{(x_i-x_s)^2+(y_i-y_s)^2+(z_i-z_s)^2}}$$

Substitute the given Condition, So

$$\cos\phi = \frac{z_i-z_s}{\sqrt{(100+20)^2+10^2+90^2}} = \frac{110-200}{\sqrt{\Delta}}$$

Therefore $\cos\phi = \left|\frac{110-200}{\sqrt{\Delta}}\right|$

$$= \left\|-\frac{90}{\sqrt{\Delta}}\right\| = \frac{90}{\sqrt{\Delta}} \qquad \frac{1}{\sqrt{\Delta}} \ll 1.$$

And the distance from $\vec{P_s}$ to $\vec{P_i}$

$$\|\vec{r}\|^2 = (x_s-x_i)^2+(y_s-y_i)^2+(z_s-z_i)^2$$

$$= 120^2+10^2+90^2$$

Therefore, we have

$$I_{diff} = 0.8\times\frac{90}{\sqrt{\Delta}}\times\frac{1}{120^2+10^2+90^2}$$

Note: Very Small! Need Post Processing for Better Visualization.

$$= 0.8\times\frac{90}{\sqrt{120^2+10^2+90^2}}\times\frac{1}{120^2+10^2+90^2} = 7.12\times10^{-5}$$

Since the result is very Small, we need to Perform Post Processing.

Note: Directional vector $\vec{P_i}-\vec{P_s}$ gives Negative Value, $\vec{P_s}-\vec{P_i}$ Positive

$\to$ take Absolute Value.

Objective: To Scale up the result in Eqn(3), to match the entire dynamic range of the display device.

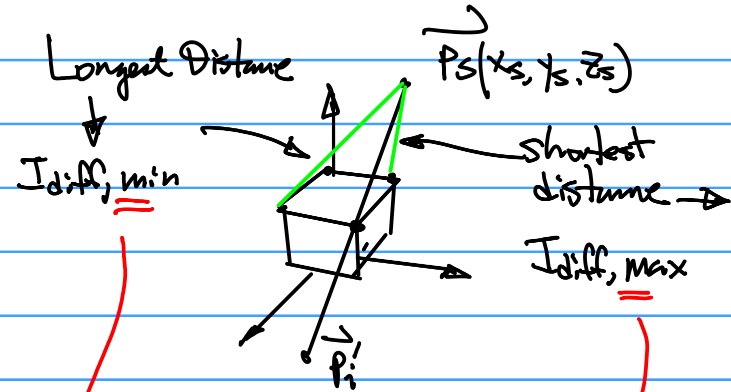
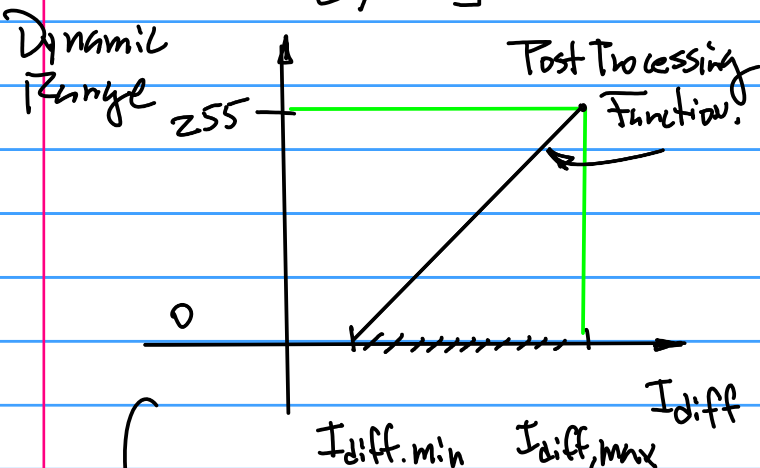


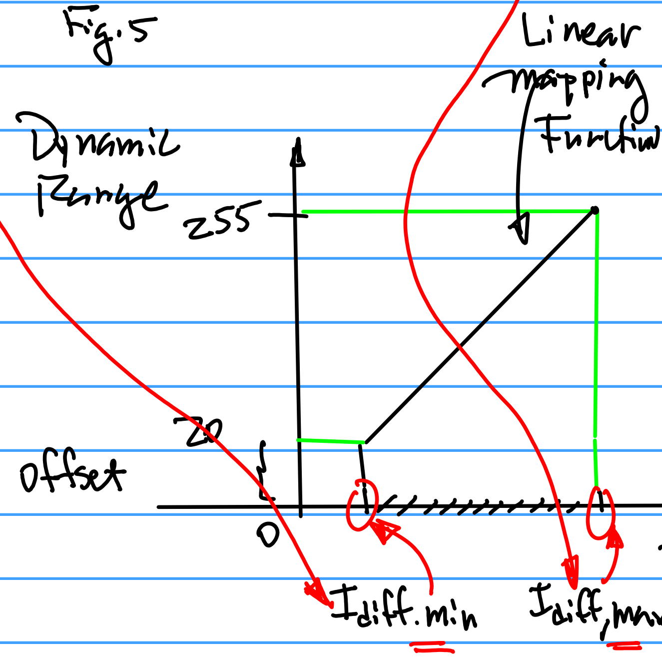
Fig. 5

1. Dynamic Range of the Display Device → 8 bits for Each Primitive color.

$2^8 = 256$. Dynamic Range $[0, 255]$



By Eqn(3) Fig(4-a)



Post processing function

Add An offset to this function

$$\frac{x-x_2}{y-y_2} = \frac{x_1-x_2}{y_1-y_2} \quad \ldots (5)$$

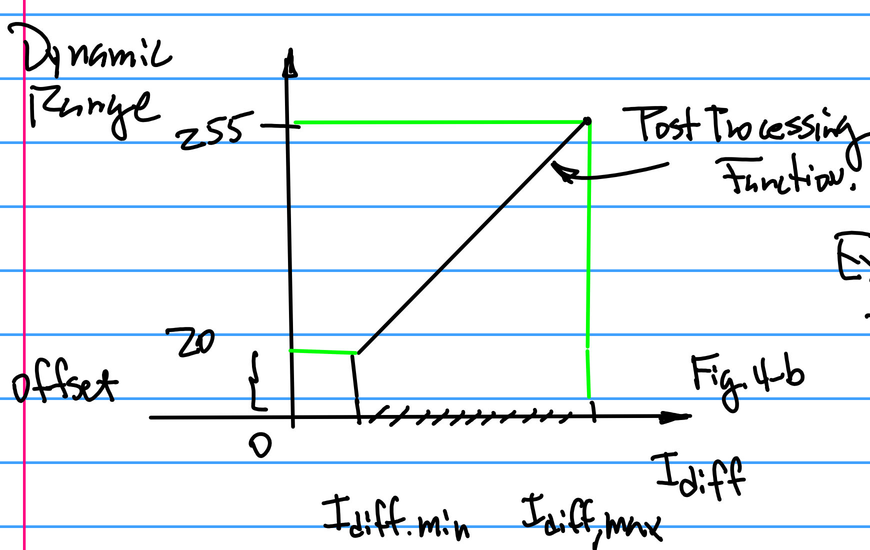


Fig. 4-b

April 20 (Wed)

Example: Post Processing Technique From Fig 5, And Eqn (5).

Dynamic Range

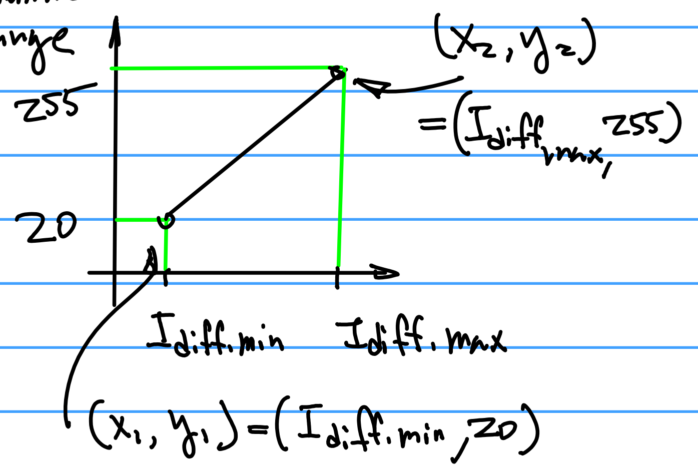


Fig. 1

From Eqn (5),

$$\frac{x - x_2}{y - y_2} = \frac{x_1 - x_2}{y_1 - y_2}$$

$$\frac{y_1 - y_2}{x_1 - x_2} = \frac{y - y_2}{x - x_2}$$

$$y = y_2 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_2)$$

$$\boxed{y = \frac{y_2 - y_1}{x_2 - x_1} x - \frac{y_2 - y_1}{x_2 - x_1} x_2 + y_2} \quad \cdots (1)$$

$$a y = b x + c, \quad y = \frac{b}{a} x + \frac{c}{a} \quad \cdots (2)$$

$$\frac{b}{a} = \frac{y_2 - y_1}{x_2 - x_1} \quad \cdots (2\text{-}b)$$

$$\frac{c}{a} = -\frac{y_2 - y_1}{x_2 - x_1} x_2 + y_2 \quad \cdots (2\text{-}c)$$

Substitute $(I_{diff,min}, 20)$, $(I_{diff,max}, 255)$

$$I_{dyn} = \frac{255 - 20}{I_{diff,max} - I_{diff,min}} x - \frac{255 - 20}{I_{diff,max} - I_{diff,min}} I_{diff,max} + 255$$

Project on Diffuse Reflection
Due May 8th (Sunday)
Compute Diffuse Reflection.
You can work the Diffuse Reflection on 4 corners of the Cube.
Make sure post processing is implemented good.

General Guide for Diffuse Reflection

Point(s) → Line(s) → Surface/plane
Vertex ✓ | Boundaries | planes

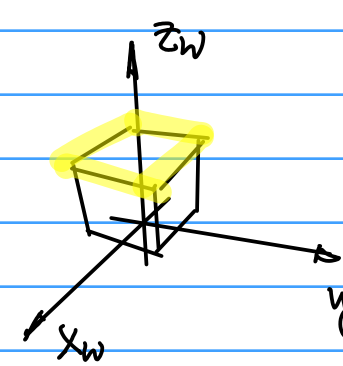


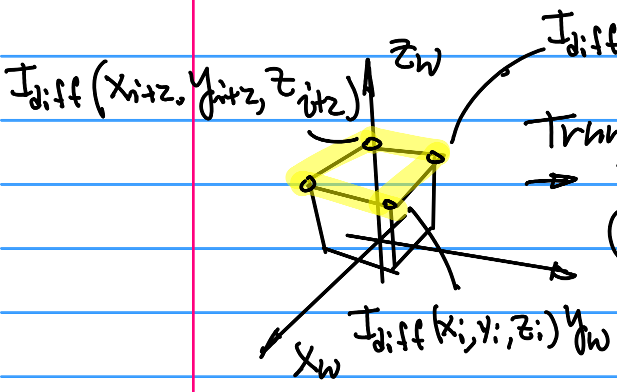
Fig 2

Two Phased Approach

- Phase I: Compute Diffuse Reflection After Transformation Pipeline
- Phase II: Map the Diffuse Reflection Result in Phase I to Actual Hardware Display Device

Example: Phase I Computation

In X_w-Y_w-Z_w



→ Transformation Pipeline (Virtual Display To Physical Display)

Display Device

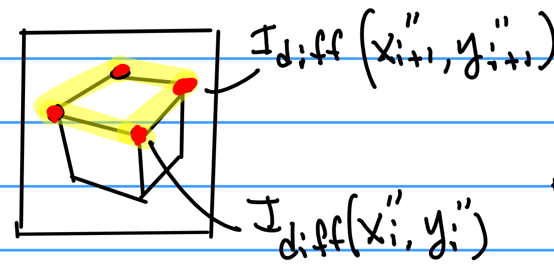


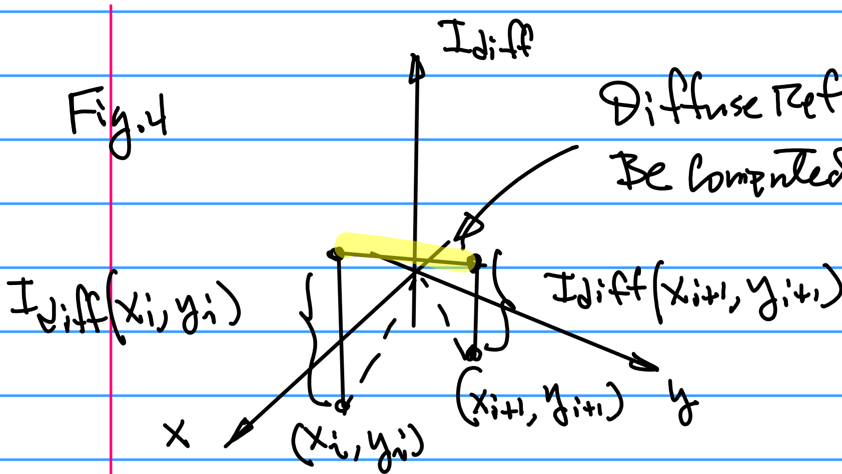
Fig 3.

Diffuse Reflections on 4 Corners Are Computed Already.

For Simplicity Purpose, with the understanding of working on the physical display, we use (x_i, y_i) not (x''_i, y''_i).

/ 2018F / 2020S-APL29-BilinearDiff1.jpg

Fig.4



Diffuse Reflection to Be Computed, on the Bounding line with $\vec{P_i}(I_{diff,i})$ and $\vec{P_{i+1}}(I_{diff,i+1})$ as Starting & Ending point.

x, y are from the Display Device

/ 2018F / 2020S-APL27-Bilinear1.jpg

Project the "yellow line" to x-z plane in Fig 4. So it is just a function of x, then compute Diffuse Reflection.

→ Similarly, project the "yellow line" to y-z plane, a function of y. then Compute Diffuse Reflection

↓

Combine both By Average Operation.

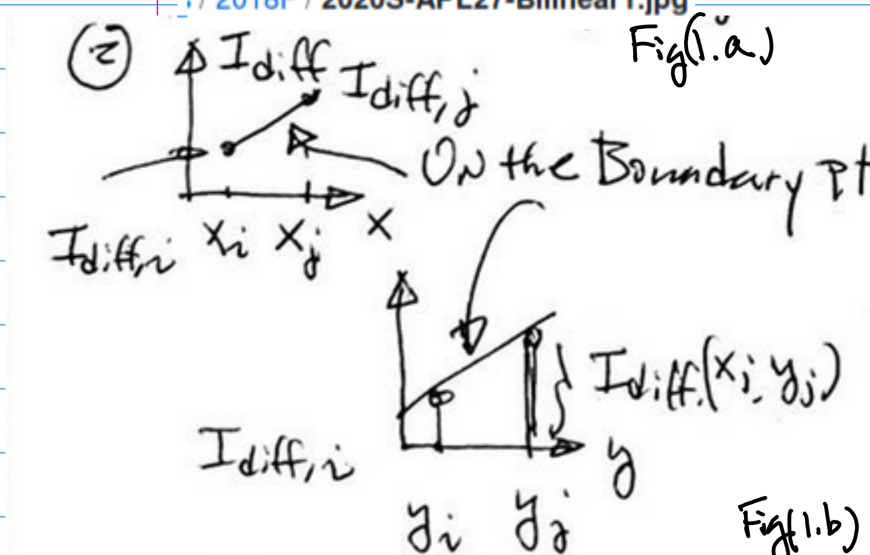
April 25 (Monday)

Topic: Boundary Line Computation for 3D G.E. Design.

Example: Formulation of Bilinear Interpolation to Compute Boundary Line (yellow) in Fig. 4.

Project the "yellow" Line onto x_w-Z(I_{diff}) axis

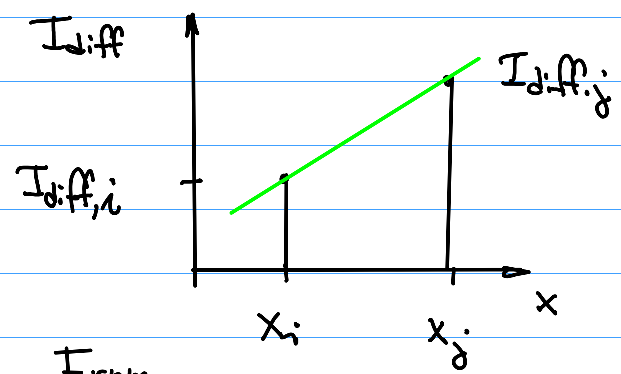
/ 2018F / 2020S-APL27-Bilinear1.jpg



Fig(1.a)

Fig(1.b)

Take care of the Diffuse Reflection on the Boundary Line w.r.t. x variable



Fig(1.c)

From,

$$\frac{x - x_2}{y - y_2} = \frac{x_1 - x_2}{y_1 - y_2} \quad \ldots (5)$$

then, derived the following Equations

$$\boxed{y = \frac{y_2 - y_1}{x_2 - x_1} x - \frac{y_2 - y_1}{x_2 - x_1} x_2 + y_2} \quad \ldots (1)$$

$$a y = b x + c, \quad y = \frac{b}{a} x + \frac{c}{a} \quad \ldots (2)$$

$$\frac{b}{a} = \frac{y_2 - y_1}{x_2 - x_1} \quad \ldots (2\text{-}b)$$

$$\frac{c}{a} = -\frac{y_2 - y_1}{x_2 - x_1} x_2 + y_2 \quad \ldots (2\text{-}c)$$

Once the diffuse Reflection(s) with respect to x independent variable is Computed, then the diffuse Reflection with respect to y independent Variable is Computed, then

$$\text{Actual Single pt. Diff. Ref} = \frac{1}{2}\left(I_{diff,x} + I_{diff,y}\right)$$

From 1-c.

(x_1, y_1):

$x_1 = x_i$

$y_1 = I_{diff,i}$

and

$x_2 = x_j$, $y_2 = I_{diff,j}$

Therefore,

2020S-APL29-BilinearDiff2.jpg

$$I_{diff,x} = \frac{I_{diff,j} - I_{diff,i}}{x_j - x_i} x - \frac{I_{diff,j} - I_{diff,i}}{x_j - x_i} x_j + I_{diff,j} \quad \ldots (3)$$

For I_{diff} w.r.t y. we have (Symmetric)

$$I_{diff,y} = \frac{I_{diff,j} - I_{diff,i}}{y_j - y_i} y - \frac{I_{diff,j} - I_{diff,i}}{y_j - y_i} y_j + I_{diff,j} \quad \ldots (4)$$

Finally, put them together,

$$I_{diff} = \frac{1}{2}\left(I_{diff,x} + I_{diff,y}\right) \quad \ldots (5)$$

Example:

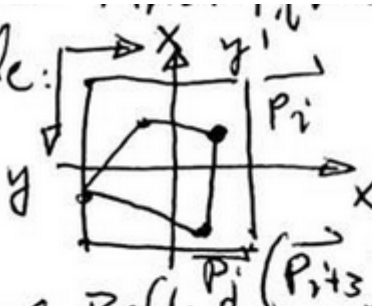


Fig.2

x-y physical Display → x'-y' Virtual Display

Diffuse Refle.

$I_{diff}(x_i, y_i)$

$I_{diff}(x_{i+3}, y_{i+3}) = I_{diff}(x_j, y_j)$

$(x_i, y_i) = (10, 12)$, $(x_j, y_j) = (8, -14)$

Given $\vec{I}_{diff}(10, 12) = (100, 0, 0)$,

$\vec{I}_{diff}(8, -14) = (80, 0, 0)$.

Find $I_{diff}(x, y)$ on the Boundary Line

April 27, Wed

Group I Classes

Group I classes are those classes which meet M, W, F, MTW, MWR, MTWF, MWRF, MTWRF, MW, WF, MWF, MF, TW, WR, MT.

Regular Class Start Times	Final Examination Days	Final Examination Ti
7:00 through 8:25 AM	Wednesday, May 18	7:15-9:30 AM
8:30 through 9:25 AM	Friday, May 20	7:15-9:30 AM
9:30 through 10:25 AM	Tuesday, May 24	7:15-9:30 AM
10:30 through 11:25 AM	Thursday, May 19	9:45 AM-12:00 PM
11:30 AM through 12:25 PM	Monday, May 23	9:45 AM-12:00 PM
12:30 through 1:25 PM	Wednesday, May 18	12:15-2:30 PM
1:30 through 2:25 PM	Friday, May 20	12:15-2:30 PM
2:30 through 3:25 PM	Tuesday, May 24	12:15-2:30 PM
3:30 through 4:25 PM*	Thursday, May 19	2:45-5:00 PM
4:30* through 5:25 PM*	Monday, May 23	2:45-5:00 PM

Now, DDA Algorithm.
(Digital Differential Algorithm)

Display Device has finite resolution. → "gaps" Problem

Example: For A finite Resolution Display Device Below,

Suppose $y=4x+1$ is given for $x=0$, $y=1$ Starting Point. Let draw a straight line for $x=0,1,2,\cdots,k$

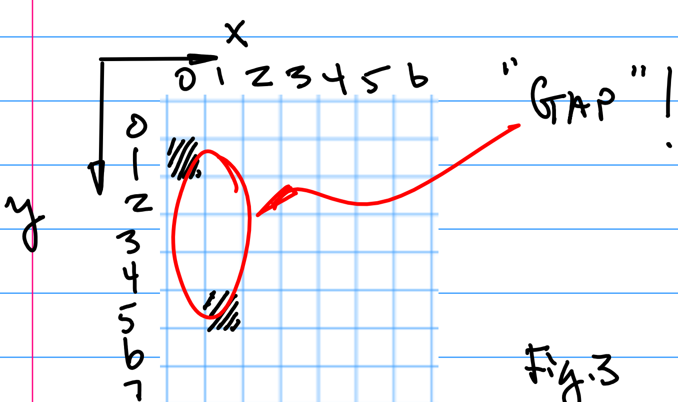


Fig.3

for $x=1$, $y=4x+1\Big|_{x=1}=5$

Note: Diffuse Reflection on the Boundary Lines is Carried After Transformation Pipeline.

Observation: The "gap" is due to the slop of $ay=bx+c$ (slop $\frac{b}{a}$) is greater than 1.

$$\left|\frac{b}{a}\right|>1$$

Increment x by 1 will Lead to increment y by a value greater than 1.

To solve this problem, we rewrite the equation as follows

$$\text{given } ay=bx+c \quad \cdots (1)$$

$$bx=ay-c$$

$$x=\frac{a}{b}y-\frac{c}{b} \quad \cdots (2)$$

$\because$ slop of $ay=bx+c$:

$\frac{b}{a}$ whose Absolute value is $\left|\frac{b}{a}\right|>1$

$\therefore$ Slop of Eqn (2), $\frac{a}{b}$ whose Absolute value $\left|\frac{a}{b}\right|<1$

The Algorithm (D.D.A: Digital Differential Algorithm) is Based on the Absolute Value of the slop must be less or Equal 1.

For the absolute value of the Slop ≤ 1, e.g.

For $ay=bx+c$ and $\left|\frac{b}{a}\right|\leq 1$, then

$$x_{k+1}=x_k+1 \quad \ldots (3\text{-}a)$$

$$y_{k+1}=y_k+\left|\frac{b}{a}\right| \quad \ldots (3\text{-}b)$$

from $ay=bx+c$

$$y=\frac{b}{a}x+\frac{c}{a}$$

Let $x=x_k$, $y_k=\frac{b}{a}x_k+\frac{c}{a}$

$$y_k=\frac{b}{a}x_k+\frac{c}{a}$$

for $x_{k+1}=x_k+1$, hence

$$y_{k+1}=\frac{b}{a}(x_k+1)+\frac{c}{a}$$

$$=\frac{b}{a}x_k+\frac{c}{a}+\frac{b}{a}$$

$$=y_k+\frac{b}{a}=y_k+\left|\frac{b}{a}\right|$$

For the slop $\left|\frac{b}{a}\right|>1$

From Eqn (2),

$$x=\frac{a}{b}y-\frac{c}{b}$$

$$\left|\frac{a}{b}\right|<1$$

Therefore, $x=\frac{a}{b}y-\frac{c}{b}$ can be dealt with By the Same Approach. e.g.

$$\begin{cases} y_{k+1}=y_k+1 & (4\text{-}a) \\ x_{k+1}=x_k+\left|\frac{a}{b}\right| & (4\text{-}b) \end{cases}$$

Example: Given a pair of Starting Point and Ending Point draw a straight line by using D.D.A.

CMPE240, April 27, 22

Sol. From the Starting point $\vec{P_i}$ And ending Point $\vec{P_{i+1}}$, we Can find line equation as follows

$$y = 4x + 1$$

Since, the Slop $|4| > 1$, then

$$x = \frac{a}{b} y - \frac{c}{b}$$

where $a = 1$, $b = 4$, $c = 1$

$$\frac{a}{b} = \frac{1}{4} \text{ whose } \left|\frac{a}{b}\right| = \left|\frac{1}{4}\right| < 1$$

For $y_k = 1$ (Starting point $x = 0$, $y = 1$)

$$x_k = \frac{1}{4} y_k - \frac{1}{4} \Big|_{y_k = 1} = 0$$

For $y_{k+1} = y_k + 1 = 1 + 1 = 2$

$$x_{k+1} = x_k + \frac{1}{4} = 0 + \frac{1}{4} = \frac{1}{4} = 0.25$$
$$\simeq 0$$

For $y_{k+2} = y_{k+1} + 1 = 3$

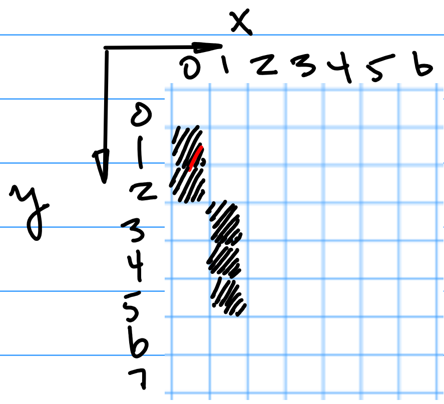
$$x_{k+2} = x_{k+1} + \frac{1}{4} = \frac{1}{4} + \frac{1}{4} = \frac{1}{2} = 0.5$$
$$\simeq 1$$

For $y_{k+3} = y_{k+2} + 1 = 3 + 1 = 4$

$$x_{k+3} = x_{k+2} + \frac{1}{4} = \frac{1}{2} + \frac{1}{4} = \frac{3}{4}$$
$$= 0.75 \simeq 1$$

For $y_{k+4} = y_{k+3} + 1 = 4 + 1 = 5$

$$x_{k+4} = x_{k+3} + \frac{1}{4} = \frac{3}{4} + \frac{1}{4} = 1$$



Programming / Implementation.

2018F-116-11diffuse20181114.cpp

2018F-117-12dda.cpp

2018F-118-13diffuseInterpolation20181127.cpp

$\vec{E}(x_e, y_e, z_e)$, $\rho = \sqrt{x_e^2 + y_e^2 + z_e^2}$, D focal length

```
float Xe = 200.0f, Ye = 200.0f, Ze = 250.0f;  //virtual camera
float Rho = sqrt(pow(Xe,2) + pow(Ye,2) + pow(Ze,2));
float D_focal = 100.0f;
//      point light source defined in line 160
```

Typedef struct for the Points in x_w-y_w-z_w, x_e-y_e-z_e, Perspective Projection

```
typedef struct {
    float X[UpperBD], Y[UpperBD], Z[UpperBD];
} pworld;

typedef struct {
    float X[UpperBD], Y[UpperBD], Z[UpperBD];
} pviewer;

typedef struct{
    float X[UpperBD], Y[UpperBD];
} pperspective;
```

Now, define diffuse Reflection intensity

```
typedef struct {
    float r[UpperBD], g[UpperBD], b[UpperBD];
} pt_diffuse;
```

Note: The Attendence of the Class is required/Mandatory. Next monday there will be Attendence Checking. please inform the other students.

$$\vec{K_d} = (K_r, K_g, K_b)$$

```
//------------diffuse reflection----------*
pt_diffuse  diffuse;   //diffuse.r[3]

//-------reflectivity coefficient----------*
#define     Kdr     0.8
#define     Kdg     0.0
#define     Kdb     0.0

```

$$\|\vec{r}\| = \|\vec{P_s} - \vec{P_i}\| = \sqrt{(x_s - x_i)^2 + (y_s - y_i)^2 + (z_s - z_i)^2}$$

```
//--------compute distance-----------------*
float distance[UpperBD];
for (int i=48; i<=49; i++) {
distance[i] = sqrt(pow((world.X[i]-world.X[45]),2)+        //intersect pt p7
                    pow((world.Y[i]-world.Y[45]),2)+
                    pow((world.X[i]-world.X[45]),2) );
//std::cout << "distance[i]  " << distance[i] << std::endl;
}

for (int i=4; i<=5; i++){
distance[i] = sqrt(pow((world.X[i]-world.X[45]),2)+        //pt p4 of projection plane
                    pow((world.Y[i]-world.Y[45]),2)+
                    pow((world.X[i]-world.X[45]),2) );
//std::cout << "distance[i]  " << distance[i] << std::endl;
}
```

CMPE240. April 27, Wed, 22

$\vec{n} \cdot (\vec{P_s} - \vec{P_i})$ (Note $\vec{P_i} - \vec{P_s}$ as Ray Eqn)
See Part-II, pp.1.

```
    //--------compute angle--------------------*
    float angle[UpperBD], tmp_dotProd[UpperBD], tmp_mag_dotProd[UpperBD];

    for (int i=48; i<=49; i++){
/*
    tmp_dotProd[i] =  (world.X[i]-world.X[45])*world.X[47]+   //...[47] for normal vector
                (world.Y[i]-world.Y[45])*world.Y[47]+         //...[45] for pt light source
                (world.Z[i]-world.Z[45])*world.Z[47];

*/
    tmp_dotProd[i] = world.Z[i]-world.Z[45];
    std::cout << " tmp_dotProd[i]  " <<  tmp_dotProd[i] << std::endl;

    tmp_mag_dotProd[i] = sqrt(pow((world.X[i]-world.X[45]),2)+        //[45] pt light source
                    pow((world.Y[i]-world.Y[45]),2)+
                    pow((world.Z[i]-world.Z[45]),2) );
    std::cout << " tmp_mag_dotProd[i]  1 " <<  tmp_mag_dotProd[i] << std::endl;

    angle[i] = tmp_dotProd[i]/ tmp_mag_dotProd[i];
```

Compute color Intensity in X_w-Y_w-Z_w

```
    //compute color intensity
    diffuse.r[i] = Kdr *  angle[i] /  pow(distance[i],2) ;
    diffuse.g[i] = Kdg *  angle[i] /  pow(distance[i],2) ;
    diffuse.b[i] = Kdb *  angle[i] /  pow(distance[i],2) ;

```

Post Processing: Adding offset value — full Dynamic Range

```
    for (int i=4; i<=5; i++) {
    r = display_scaling*diffuse.r[i]+display_shifting;
    g = diffuse.g[i]; b = diffuse.b[i] ;
```

$I_{diff,x}$

```
// y - y1 = (y2-y1)/(x2-x1) (x-x1)
// for x direction
// independent variable is .X and function y is diffuse reflection intensity
newx_rDiff_Pt = rDiff_Point[4] +
                (rDiff_Point[5] - rDiff_Point[4])/(perspective.X[5]-perspective.X[4])*
                (mid_x - perspective.X[4]);
newx_gDiff_Pt = 0.0; newx_bDiff_Pt = 0.0;
```

$I_{diff,y}$

```
// for y direction
// independent variable is .Y and function y is diffuse reflection intensity
newy_rDiff_Pt = rDiff_Point[4] +
                (rDiff_Point[5] - rDiff_Point[4])/(perspective.Y[5]-perspective.Y[4])*
                (mid_y - perspective.Y[4]);
newy_gDiff_Pt = 0.0; newy_bDiff_Pt = 0.0;
```

$$I_{diff} = \frac{1}{2}\left(I_{diff,x} + I_{diff,y}\right)$$

```
// combination of both
new_rDiff_Pt = (newx_rDiff_Pt +  newy_rDiff_Pt)/2.0;
new_gDiff_Pt = 0.0; new_bDiff_Pt = 0.0;
```