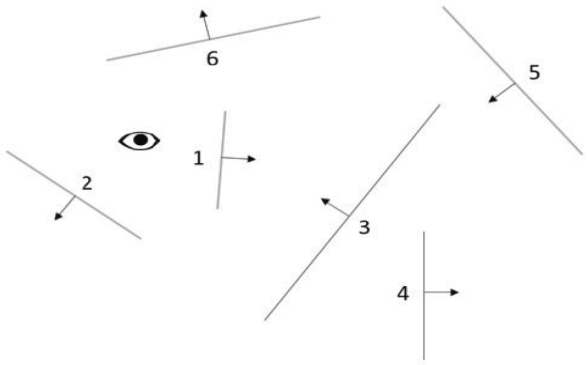
ICG 2018 Midterm

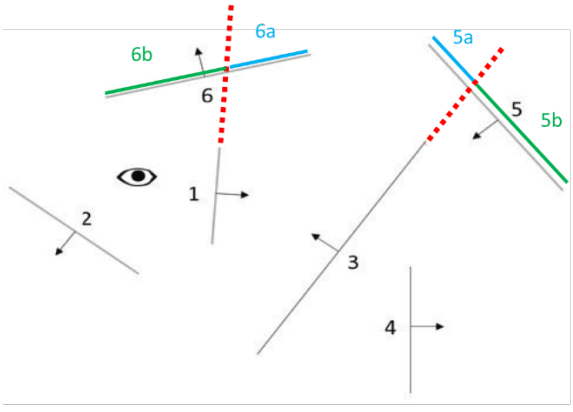
參考解答(不保證為標準答案)

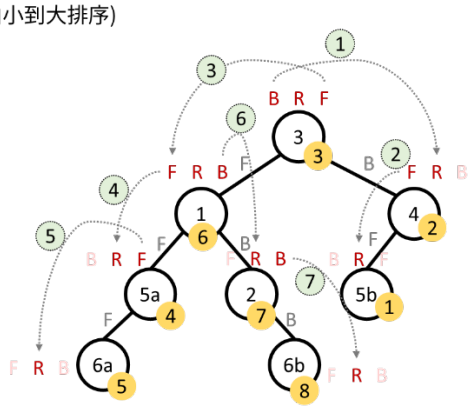
1. BSP Tree (10%)
   - Construct the Binary Space Partitioning (BSP) tree of the following figure. Please use the node “3” as the root, and choose smaller numbers as the sub-tree root node. (5%)
   - From the BSP tree of previous question, derive the display sequence in terms of the given viewing position in this figure. (5%)



解答：

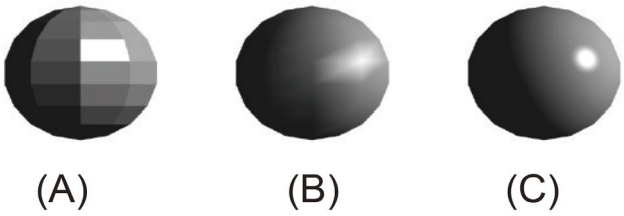
以上為a、下為b；若水平線才以左為a、以右為b(且數字由小到大排序)





順序：5b, 4, 3, 5a, 6a, 1, 2, 6b

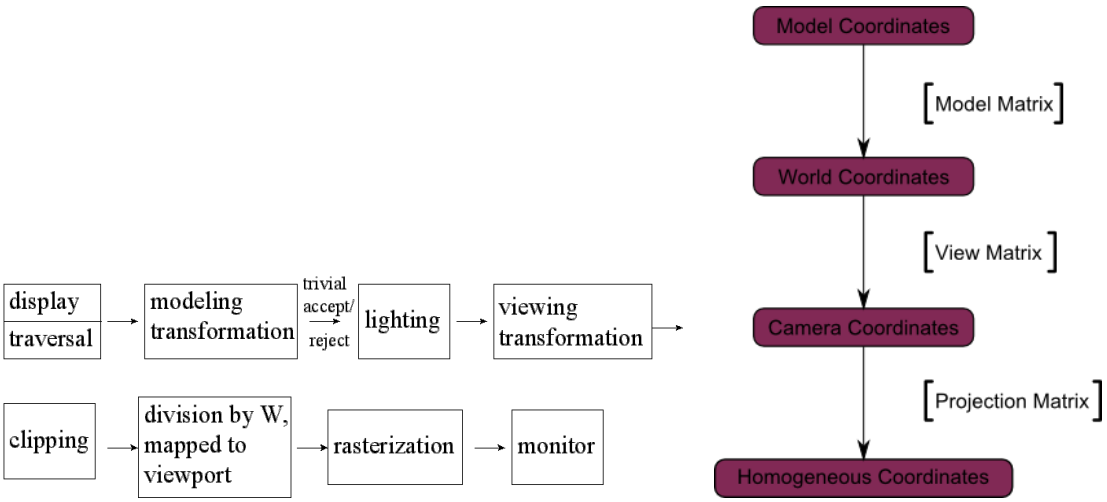
2. Shading (6%)
   - In homework 1, you implement three kinds of shading. Consider each figure below, determine which kind of shading it uses and explain your reason respectively. (6%)



解答：(A) Flat shading, (B) Gouraud shading, (C) Phong shading

3. GPU rendering pipeline (10%)
   - Describe the GPU rendering pipeline. (4%)
   - What is the meaning of model matrix, view matrix and projection matrix? Please describe them respectively. (6%)

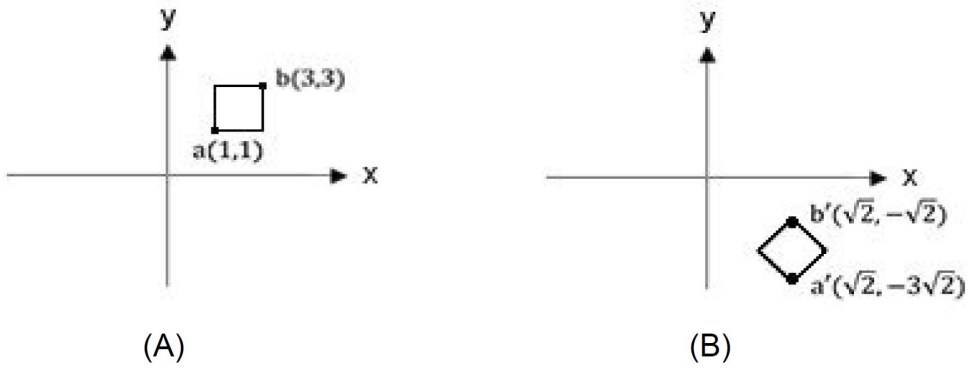
解答：



Model Matrix：從 Model Space 轉換到 World Space 的(運算)矩陣
View Matrix：從 World Space 轉換到 Camera Space 的(運算)矩陣
Projection Matrix：從 Camera Space 轉換到 Homogeneous Space 的(運算)矩陣

(src)

4. Transformation matrix (12%)
   For the following questions, consider as 2d plane. You need to write down all 3x3 matrices respectively and the matrix multiplication order.
   - For the figures below, find transform matrices to convert the square in (A) into the square in (B) (6%)



1. 先將原圖移至原心(中心點與原心重疊)
2. 進行旋轉
3. 再將旋轉後之圖片，移至結果位置(B 點)

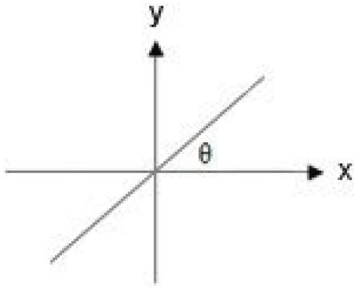
$$\begin{bmatrix}-1\\-1\\1\end{bmatrix}=\begin{bmatrix}1&0&-2\\0&1&-2\\0&0&1\end{bmatrix}\begin{bmatrix}1\\1\\1\end{bmatrix}\quad,\quad\begin{bmatrix}1\\1\\1\end{bmatrix}=\begin{bmatrix}1&0&-2\\0&1&-2\\0&0&1\end{bmatrix}\begin{bmatrix}3\\3\\1\end{bmatrix}$$

$$\begin{bmatrix}0\\-\sqrt{2}\\1\end{bmatrix}=\begin{bmatrix}\frac{1}{\sqrt{2}}&-\frac{1}{\sqrt{2}}&0\\\frac{1}{\sqrt{2}}&\frac{1}{\sqrt{2}}&0\\0&0&1\end{bmatrix}\begin{bmatrix}-1\\-1\\1\end{bmatrix}\quad,\quad\begin{bmatrix}0\\\sqrt{2}\\1\end{bmatrix}=\begin{bmatrix}\frac{1}{\sqrt{2}}&-\frac{1}{\sqrt{2}}&0\\\frac{1}{\sqrt{2}}&\frac{1}{\sqrt{2}}&0\\0&0&1\end{bmatrix}\begin{bmatrix}1\\1\\1\end{bmatrix}$$

$$\begin{bmatrix}\sqrt{2}\\-3\sqrt{2}\\1\end{bmatrix}=\begin{bmatrix}1&0&\sqrt{2}\\0&1&-2\sqrt{2}\\0&0&1\end{bmatrix}\begin{bmatrix}0\\-\sqrt{2}\\1\end{bmatrix}\quad,\quad\begin{bmatrix}\sqrt{2}\\-\sqrt{2}\\1\end{bmatrix}=\begin{bmatrix}1&0&\sqrt{2}\\0&1&-2\sqrt{2}\\0&0&1\end{bmatrix}\begin{bmatrix}0\\\sqrt{2}\\1\end{bmatrix}$$

$$\Rightarrow\begin{bmatrix}1&0&-2\\0&1&-2\\0&0&1\end{bmatrix}*\begin{bmatrix}\frac{1}{\sqrt{2}}&-\frac{1}{\sqrt{2}}&0\\\frac{1}{\sqrt{2}}&\frac{1}{\sqrt{2}}&0\\0&0&1\end{bmatrix}*\begin{bmatrix}1&0&\sqrt{2}\\0&1&-2\sqrt{2}\\0&0&1\end{bmatrix}=\begin{bmatrix}\frac{1}{\sqrt{2}}&-\frac{1}{\sqrt{2}}&-\sqrt{2}\\\frac{1}{\sqrt{2}}&\frac{1}{\sqrt{2}}&-4\sqrt{2}\\0&0&1\end{bmatrix}$$

- The simplest way to simulate a mirror is to copy objects which is symmetric to the mirror. Now given a mirror line with angle θ , please find transform matrices to do this. (6%) (Hint : use scale matrix with minus value)



解答 1：直接轉 2θ 度

$$\begin{bmatrix} a' \\ b' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(2\theta) & -\sin(2\theta) & 0 \\ \sin(2\theta) & \cos(2\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} a \\ b \\ 1 \end{bmatrix}$$

解答 2：先轉 90-θ 度，鏡射 y 軸(x 加負號)，再轉 θ-90 度回來

$$\begin{bmatrix} a' \\ b' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta-90) & -\sin(\theta-90) & 0 \\ \sin(\theta-90) & \cos(\theta-90) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \cos(90-\theta) & -\sin(90-\theta) & 0 \\ \sin(90-\theta) & \cos(90-\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} a \\ b \\ 1 \end{bmatrix}$$

5. Ray intersection (8%)
   - In ray tracing, we need to judge if the ray intersects with a shape or not. Given a ray in 3d $(x_t , y_t , z_t) = (v_x , v_y , v_z) \times t + (o_x , o_y , o_z)$ $t \geq 0$, and a square formed by 4 points (-r, -r, 0), (-r, r, 0), (r, r, 0), (r, -r, 0), where and $(v_x , v_y , v_z) =$ $(-1 , -1 , -1)$, r = 2.5. Check if the ray intersects with the square, and (o_x , o_y , o_z) = (4, 5, 6), calculate the intersection point if any one of them exists.

解答：四點之平面可視為 z=0，則求：(x, y, 0) = (-1, -1, -1)t + (4, 5, 6)即可
⇨ (x, y) = (-2, -1)
故假設 r >= 2 or r <= -2，則方形與射線之交點為(-2, -1, 0)，反之無交點

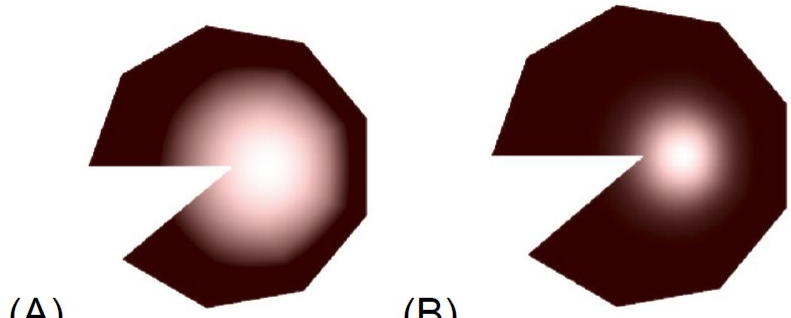
6. (5%) What is your term project for this semester? What are the technical difficulties involved in the project? (You can refer to the project listing).

解答：自行回答，只要有針對題目回答，基本上是送分題；可參考：
http://www.cmlab.csie.ntu.edu.tw/~cchank/term_project_list.pdf

7. Phong illumination model (4%)

$$I = k_a I_a + \sum_{lights} (k_d I_{m,s} \cos\theta + k_s I_{m,d} \cos^{\alpha}\theta)$$

- In Phong illumination model, the specular value α is also a parameter of material. Compare images below, which specular value is larger? (4%)



解答：B，因為金屬表面的反射光點會較為集中

8. Rendering equation (6%)

$$L_o(\mathbf{x}, \omega_o, \lambda, t) = L_e(\mathbf{x}, \omega_o, \lambda, t) + \int_{\Omega} f_r(\mathbf{x}, \omega_i, \omega_o, \lambda, t)\, L_i(\mathbf{x}, \omega_i, \lambda, t)\, (\omega_i \cdot \mathbf{n})\, \mathrm{d}\omega_i$$

- Can Phong illumination model solve the rendering equation? Why or why not? (6%)

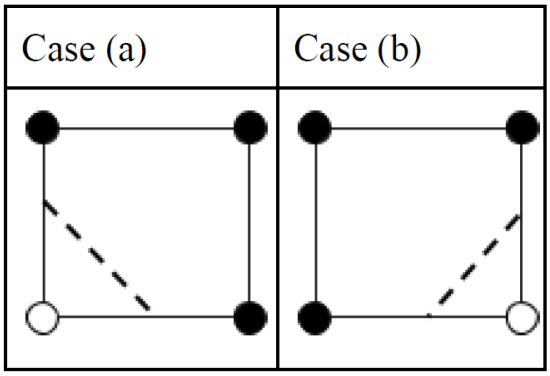
解答：無法，因為光線打到物體便停止，若要做 Rendering，光線需繼續多次反射，且不能假設物品表面為光滑

9. Volume Rendering / Marching Cubes (12%)
   - Please give two examples for Volume Rendering applications. (4%)

解答：
Volume Rendering 定義：藉由 2D 資料，建模後視覺化成 3D 資料，且可能會用顏色加以辨識。故舉例可為：1. 醫療用(MRI, CT)、 2. 天氣模擬：(颱風、龍捲風)

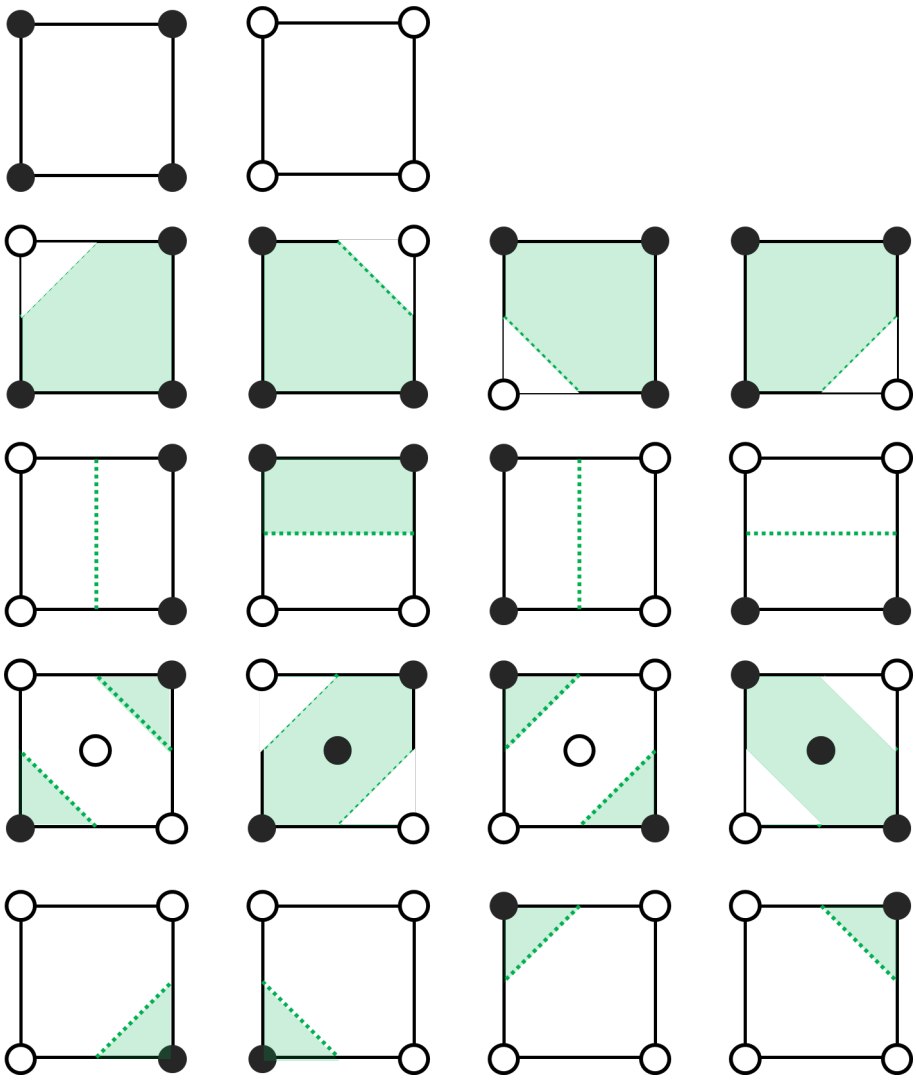
   - Volume Rendering use marching cubes to determine the contour of the volume. Now we consider the 2d condition, i.e. square marching. The square marching consists of two steps. First, apply the threshold of contour, and label the point with black if the value of the point is lower than threshold, otherwise label with white.

     Second, determine the contour by the case of 4 points' color. For example, in the case (a) and case (b), the contour is the dash line shown in the picture.

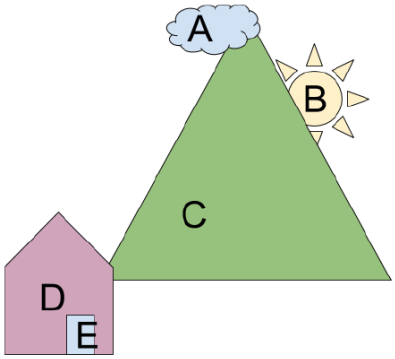


     - How many cases should square marching consider? Draw all of them. Each case should include 4 points and contour. Note that case(a) and case(b) are different. If there are any ambiguity of possible contour exists, just draw one of them. (8%)

解答：
以四點為例，正常只需考量 $2^4 = 16$ 種，但因為有些特定狀況，無法判定中心，故要再細分(第 5 橫排)



10. Painter's algorithm (4%)



E is a door of the house D.

- In image above, what is the drawing order if we use painter's algorithm? If multiple possible answers exist, answer one of them. (4%)

解答：Painter's Algorithm 是由後往前畫(有重疊則無法實作)，故基本上以下三組解答均正確：
B, C, A, D, E 或 B, C, D, E, A 或 B, C, D, A, E

11. (8%) It is said that Microsoft game machines X-Box (1, and 360) are costing more than their sale prices, and then use the game software revenue to compensate for the loss in hardware.
    (a) If this is true, what advantages for Microsoft in developing the hardware platform can be predicted from this business model? (4%)
    (b) In the above case, is it easy for companies from Korea or Taiwan to develop their own game machines or clones of these machines? Why and why not? (4%)

解答：Painter's Algorithm 是由後往前畫(有重疊則無法實作)，故基本上以下三組解答均正確：
(a) 其策略是以便宜的硬體，來推廣、搶佔市場，一個資本戰的概念
(b) 很難，因為需要極其大量的資金，以台、韓企業來說，執行上有困難

12. (15%) Modified Distribution Ray tracing
    (a) What's the weakness of ray-tracing as compared to the Rendering equation? (short answer) (5%)

解答：Ray tracing 假設物體表面光滑、且光線不會多次反射；故其缺點為：無法處理粗糙表面物體，及無法非常準確模擬現實世界之狀況(否則運算量過大、速度過慢)。

    (b) In the photo below, there are very bright spots under the glass sphere, called caustics. One way to improve the previous ray-tracing is to combine the rays (light packets) shooting from the lights, and store the (location, direction) information at the light-surface intersection points, called photon maps. (10%)

For caustics, we only store the photon map when rays hit the highly specular surface or pass through a transparent object and finally reach a diffuse surface.

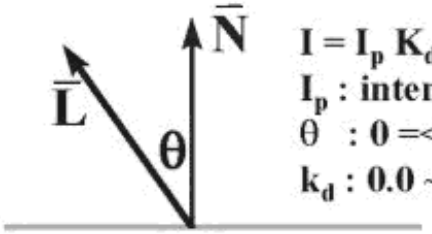
After the photon maps are created, we can use ray-tracing to shoot rays from the eye until it hits the surfaces with the photon maps. Why is this method successful in solving the Rendering Equation? Please give your own algorithm describing the previous solution.

解答(本題為開放式題目)：
Caustics 是結合 Ray tracing 與高中物理，在可能會發生錯誤的狀況下，特別處理。

Illumination model

1) **Ambient light** (漫射)(or 環境反射)
   I = Ia * ka * Obj(r, g, b)
   Ia : intensity of ambient light
   ka : 0.0 ~ 1.0, Obj(r, g, b): object color

2) **Diffuse reflection** (散射)(or 漫反射)
   I = Ip (r, g, b) * Kd * Obj(r, g, b) * COS(**θ**)
   Ip (r, g, b): light color



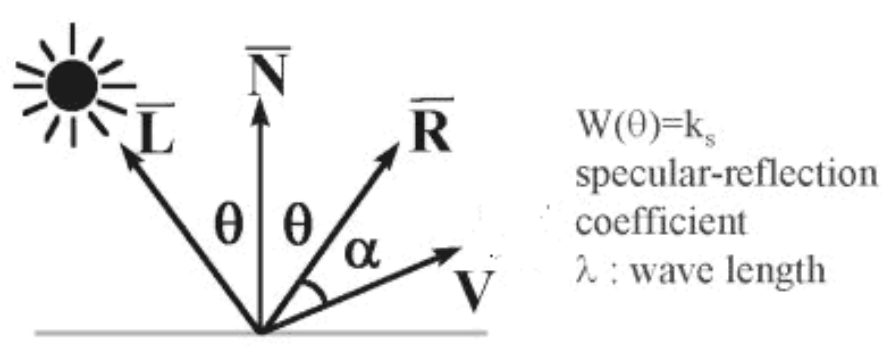
$I = I_p K_d \cos\theta$
I_p : intensity of light source
θ : $0 =< \theta <= 90$
k_d : 0.0 ~ 1.0 (material dependent)

3) **Light source attenuation**

$$I = I_a k_a + fatt\ I_p K_d(\bar{N}.\bar{L}) \quad fatt = \frac{1}{d_L^2}$$

Specular reflection (似鏡面反射)

- **$I = K_s * I_p(r, g, b) * \cos^n(\alpha)$,**
  K_s = specular-reflection coef.



- Phong illumination model

  color of object = Obj (R, G, B)
  = (Or, Og, Ob) or (light frequency)
  where 0.0 =< Od =< 1.0

Polygon shading : linear interpolation

a. flat shading : constant surface shading.
b. Gouraud shading: color interpolation shading.
c. Phong shading: vertex normal interpolation shading

The Display Order of Binary Space Partition Trees(BSP tree)

if Viewer is in front of root, then

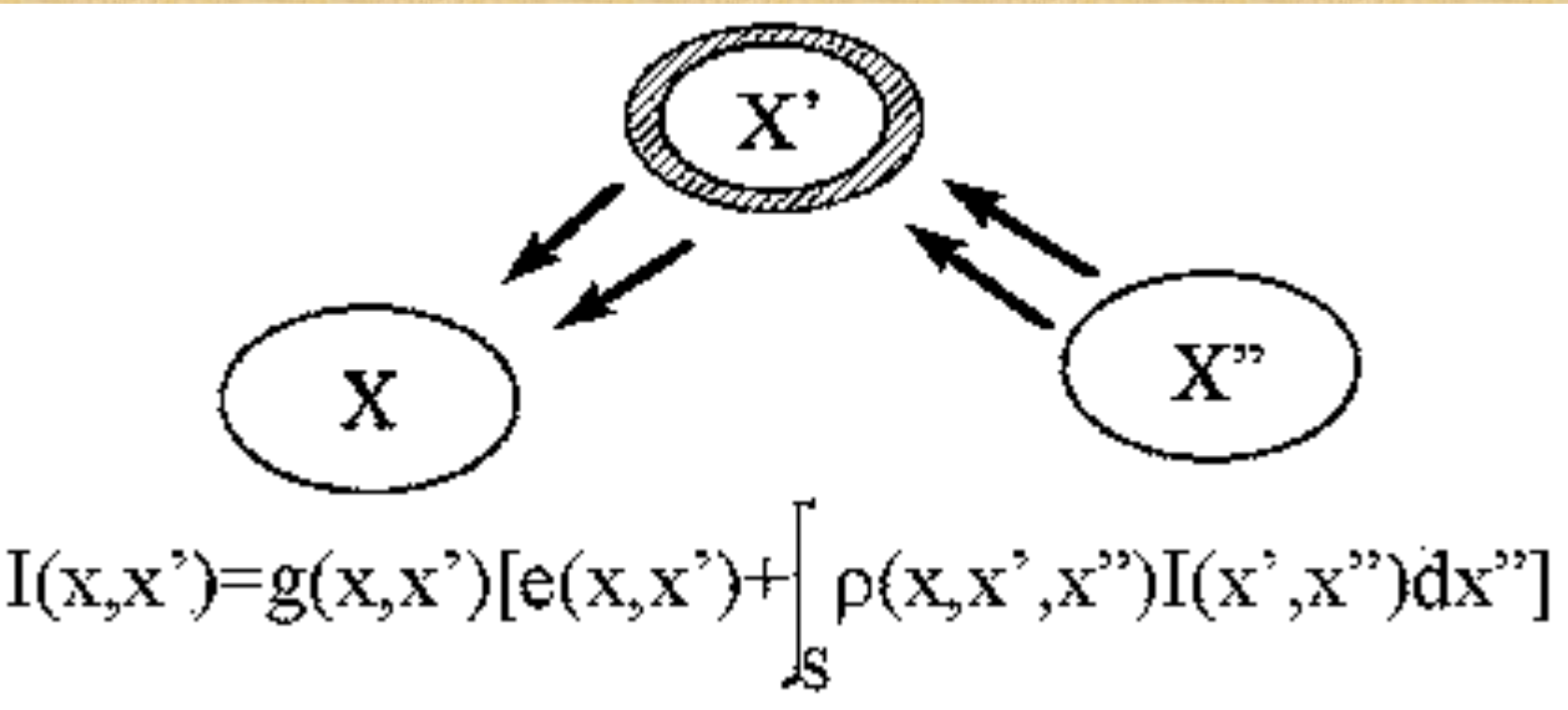
- Begin {display back child, root, and front child}
- BSP_displayTree(tree->backchild)
- displayPolygon(Tree->root)
- BSP_displayTree(tree->frontchild)
- end

else

- Begin
- BSP_displayTree(tree->frontchild)
- displayPolygon(Tree->root)
- BSP_displayTree(tree->backchild)
- end

The Rendering Equation: Jim Kajiya, 1986

(渲染方程式)



I(x, x'): light (intensity) from patch x' to patch x

g(x,x'): visibility geometry, o: invisible, 1: visible

e(x, x'): the rate at which light is emitted from patch x' to x, when x' is an emitter.

ρ (x, x', x''): patch's reflectivity,

Ray Tracing Algorithm

```
Trace(ray)
    For each object in scene
        Intersect(ray, object)

    If no intersections
        return BackgroundColor

    For each light
        For the object in scene
            Intersect(ShadowRay, object)
            Accumulate local illumination
            Trace(ReflectionRay)
            Trace(TransmissionRay)
            Accumulate global illumination
```

Error Diffusion Dithering: example

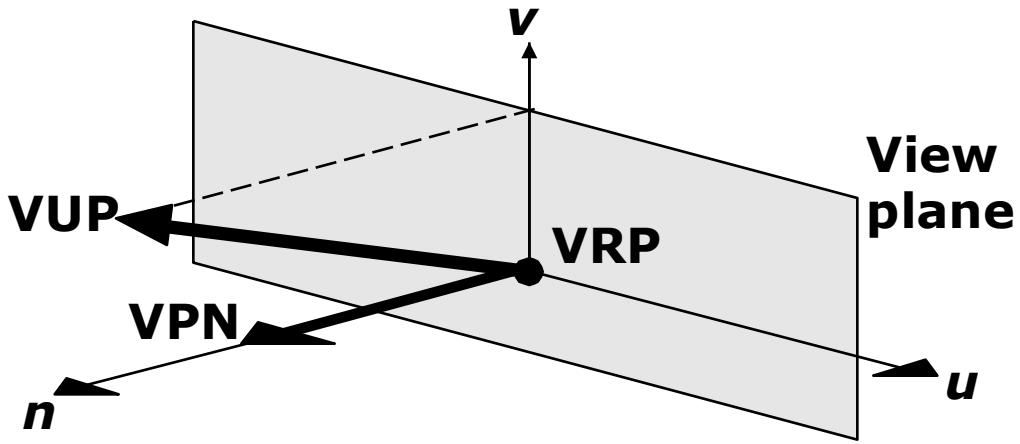
- How to approximate such an array if we have elements of the form 2xN only?
  (i.e. 0,2,4,6,8,10, etc.) (initial approximation
- by lower bound)

$$\begin{matrix} 7 & 7 & 5 \\ 3 & 1 & 3 \\ 5 & 8 & 7 \end{matrix}$$

$$\begin{matrix} 6 & 6 & 4 \\ 2 & 2 & \end{matrix} \qquad \begin{matrix} 1 & 1 & +1 \\ 1 & 1 & x \\ x & x & x \end{matrix} \qquad \begin{matrix} 6 & 7+3/8 & 5+33/64 \\ 3+3/8 & 1+82/64 & x \\ x & x & x \end{matrix}$$

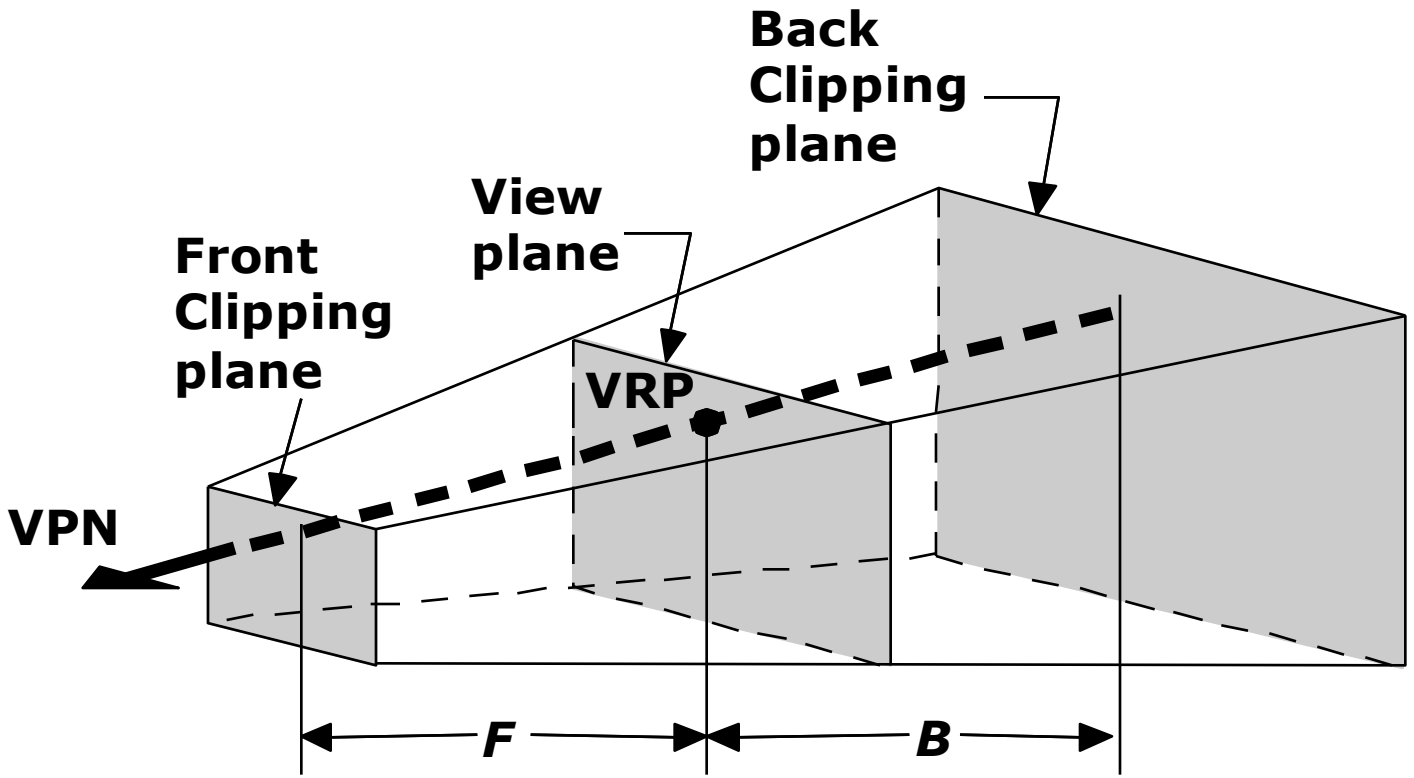
error intermediate calculation

Specification of an Arbitrary 3D View



- VRP: view reference point
- VPN: view-plane normal
- VUP: view-up vector

Truncated View Volume for an
Perspective Projection
(how to do 3D clipping?), PRP: projection reference point,
DOP: direction of projection

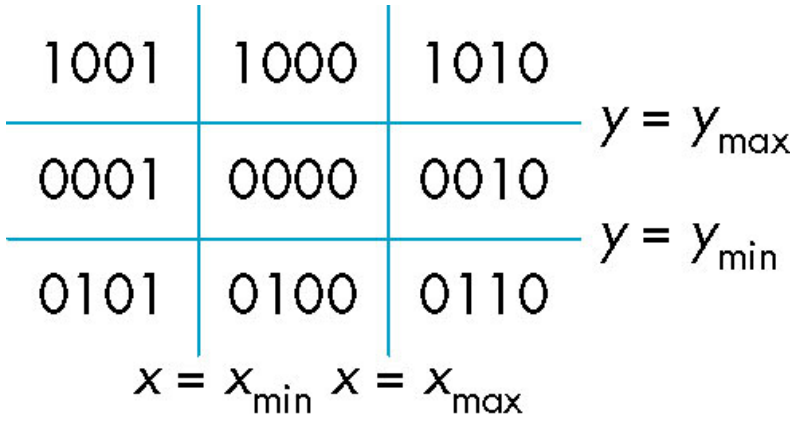


Defining Outcodes

- For each endpoint, define an outcode

$$b_0 b_1 b_2 b_3$$

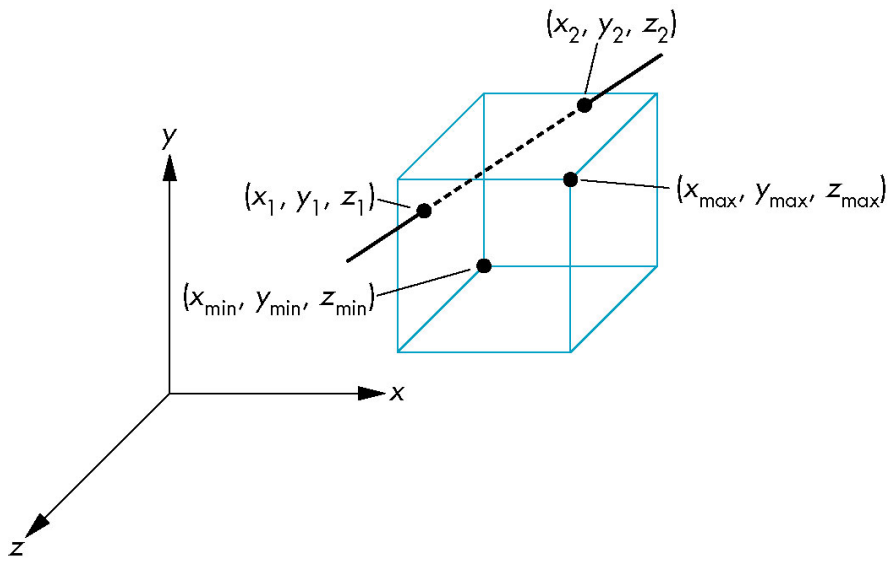
$b_0 = 1$ if $y > y_{max}$, 0 otherwise
$b_1 = 1$ if $y < y_{min}$, 0 otherwise
$b_2 = 1$ if $x > x_{max}$, 0 otherwise
$b_3 = 1$ if $x < x_{min}$, 0 otherwise

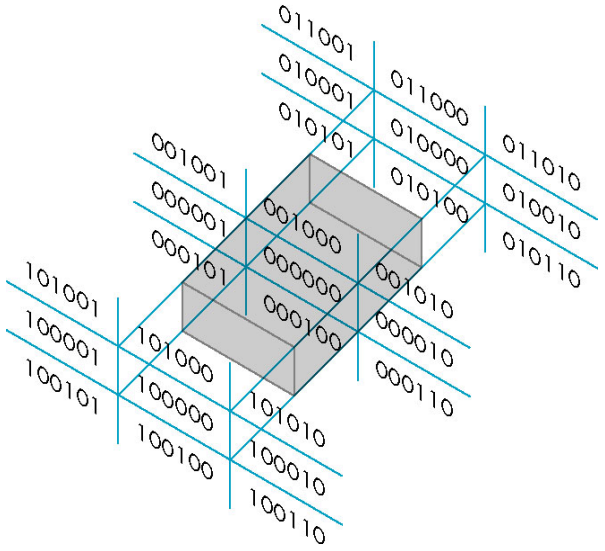


- Outcodes divide space into 9 regions
- Computation of outcode requires at most 4 subtractions

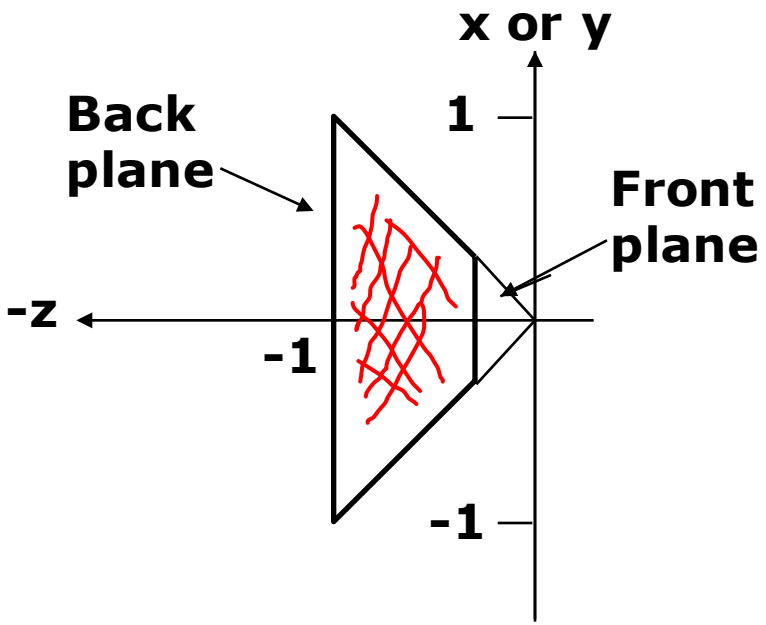
Cohen Sutherland in 3D

- Use 6-bit outcodes
- When needed, clip line segment against planes





Canonical View Volume for Perspective Projection: 3D clipping is easier this way!



- $x = z, y = z, z = -z_{min}$
- $x = -z, y = -z, z = -1$

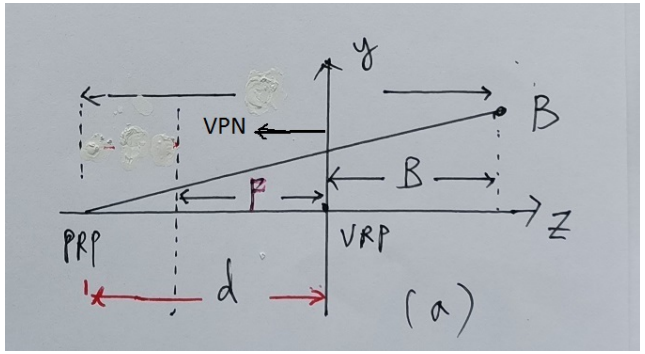
The Extension of the Cohen-Sutherland Algorithm

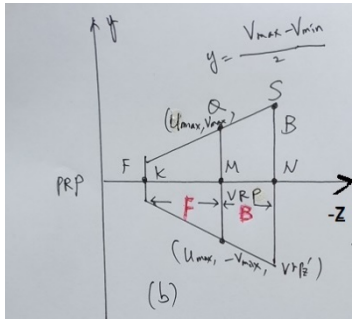
- bit 1 – point is above view volume $y > -z$
- bit 2 – point is below view volume $y < z$
- bit 3 – point is right of view volume $x > -z$
- bit 4 – point is left of view volume $x < z$
- bit 5 – point is behind view volume $z < -1$
- bit 6 – point is in front of view volume $z > -z_{min}$

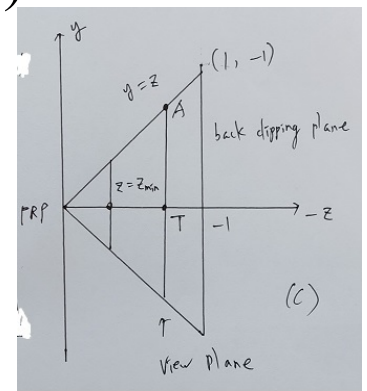
The Steps of Implementation of Perspective Projection

- Translate the VRP to the origin
- Rotate VRC such that the VPN becomes the z axis
- Translate such that the PRP is at the origin
- Shear such that the DOP becomes parallel to the z axis
- Scale such that the view volume becomes the canonical perspective view volume

$$N_{per} = S_{per} \quad SH_{per} \quad T(PRP) \quad R \quad T(VRP)$$







$$N_{per} = S_{per} \quad SH_{per} \quad T(\ PRP) \quad R \quad T(\ VRP)$$

- $N_{per} = S_{per} * SH_{per} * T(-PRP) * R * T(-VRP)$
  $T(-VRP) = T(0,0,0)$ where no oblique proj.

  $T(-PRP) = T((0,0,d))$

- $SH_{per}=1$, $R=R_y(180^o)=$

$$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

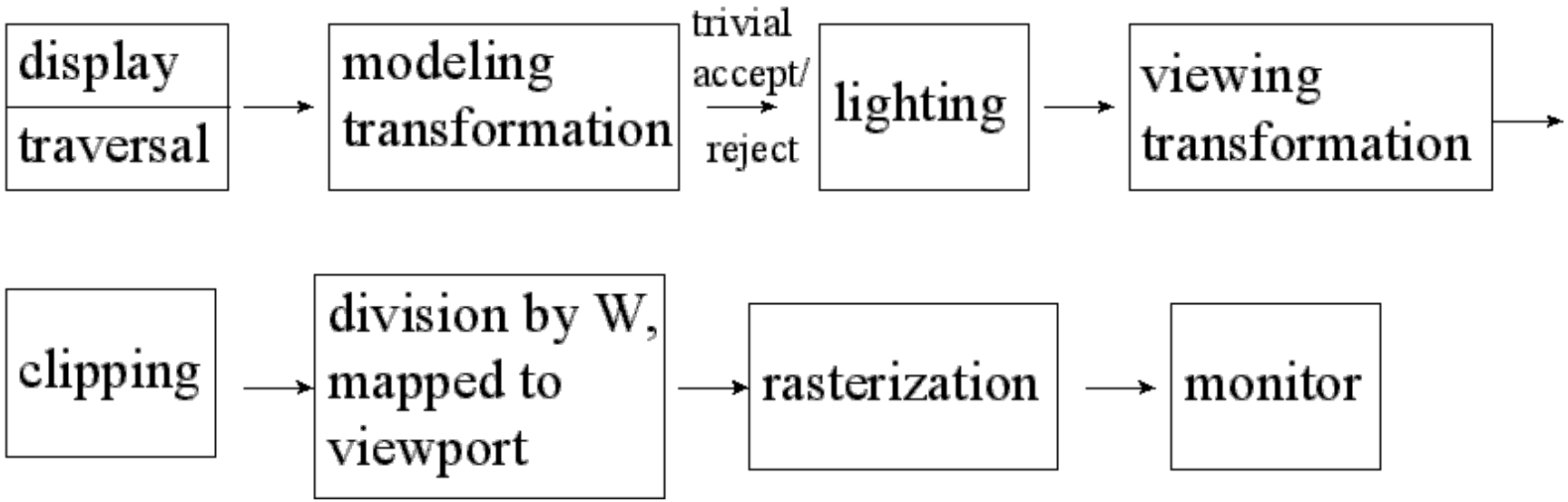
$$N_{per} = S_{per} \quad SH_{per} \quad T(\ PRP) \quad R \quad T(\ VRP)$$

- S_{per}(scaling)=

$$S \ \frac{2\text{vrp}_z'}{(U_{max} - U_{min})(\text{vrp}_z'+B)}, \frac{2\text{vrp}_z'}{(V_{max} - V_{min})(\text{vrp}_z'+B)}, \frac{1}{\text{vrp}_z'+B}$$

- Assuming back clipping plane = (U_{max}, V_{max}, B)
- Front clipping plane = (U_{min}, V_{min}, Z_{min} = F)
- For example, let U_{max} = 5 = V_{max}, B=10, F=-d+2
  U_{min} = -5 = V_{min}, vrp_z‘ = d, where d=8

Standard Graphics Pipeline



Computer Graphics、Image Processing、Computer Vision、Pattern Recognition 差別：

		輸出	
		描述	影像
輸入	描述		Computer Graphics
	影像	Computer Vision Pattern Recognition	Image Processing

例：

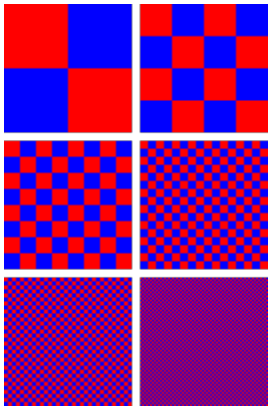
- CG：畫一隻狗
- CV、PR：人臉辨識
- IP：修圖(Lightroom、Photoshop、美圖秀秀...)

4k(UHD)： 3840×2160
2k(QHD)： 2560×1440
1080p(FHD)： 1920×1080
720p： 1280×720

顏色(色彩深度)：早期 8 bit/pixel：256 色，一直到近期的 24 bit/pixel(RGB 各 8 bits)，稱為 True Color(真彩色)

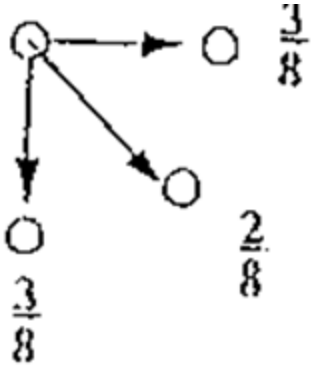
(助註：一般常見螢幕是 8 bits，較少見 for 攝影師、設計師的才會到 10 bits，且價格相當高；現階段高端相機也不乏可錄製 10 bits 影片的機型，但照片則可以拍攝 RAW 檔，高達 14 bits 以上；延伸關鍵字：10 bits、444/422/420、外錄、超採樣...)

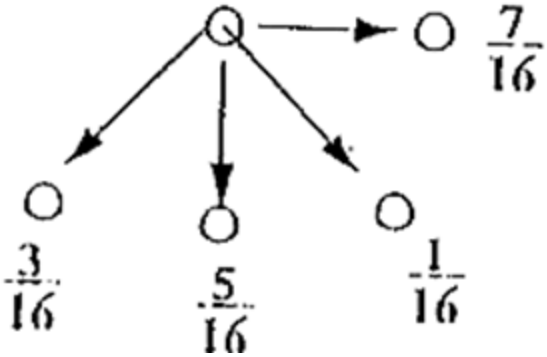
Dithering(抖動著色)：利用人眼視覺特性(10m 外距離 0.2mm 兩點視為同一點)，用兩種顏色混合成另一種單一顏色



視覺：10 公尺外，距離 0.2mm 之兩點，將視為同一點(範例：紙鈔)

(Floyd-Steinberg) Error Diffusion Dithering(差錯擴散)：





原始：

7	7	5
3	1	3
5	8	7

標示說明：

暗紅色底表示正在執行的 Pixel，綠色=小於等於它的最大偶數，藍色=多出來的誤差，且要分配給右、下、右下；紅色表依比例分配出去之誤差。以下為各步驟：

步驟 1：

6+1	7+3/8	5
3+3/8	1+2/8	3
5	8	7

6	59/8	5
27/8	10/8	3
5	8	7

步驟 2：

6	6+11/8	5+33/64
27/8	10/8+33/64	3+22/64
5	8	7

6	6	353/64
27/8	113/64	214/64
5	8	7

步驟 3：

6	6	4+97/64
27/8	113/64	214/64+291/512
5	8	7

6	6	4
27/8	113/64	2003/512
5	8	7

步驟 4：

6	6	4
2+11/8	113/64+33/64	2003/512
5+33/64	8+22/64	7

6	6	4
2	146/64	2003/512
353/64	534/64	7

步驟 5：

6	6	4
2	2+18/64	2003/512+54/512
353/64	534/64+54/512	7+36/512

步驟 6：

6	6	4
2	2	2057/512
353/64	4326/512	3620/512

6	6	4
2	2	4+9/512
353/64	4326/512	3620/512+27/4096

步驟 7：

6	6	4
2	2	4
4+97/64	4326/512+291/512	28987/4096

6	6	4
2	2	4
4	4617/512	28987/4096

步驟 8：

6	6	4
2	2	4
4	8+521/512	28987/4096+1563/4096

6	6	4
2	2	4
4	8	30550/4096

步驟 9：

6	6	4
2	2	4
4	8	6+5974/4096

6	6	4
2	2	4
4	8	6

故 Error：

1	1	1
1	-1	-1
1	0	1

應用：印象派畫風(Impressionism)、影印機、螢幕...等

原圖中的平形線，不論如何旋轉、縮放與平移，結果仍維持平行，但角度、長短則不然(可能改變)

Shear 剪力：

二維：

$$\begin{bmatrix} 1 & a & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x+ay \\ y \\ 1 \end{bmatrix}$$

三維：

$$SH_{xy}(SH_x, SH_y) = \begin{bmatrix} 1 & 0 & SH_x & 0 \\ 0 & 1 & SH_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

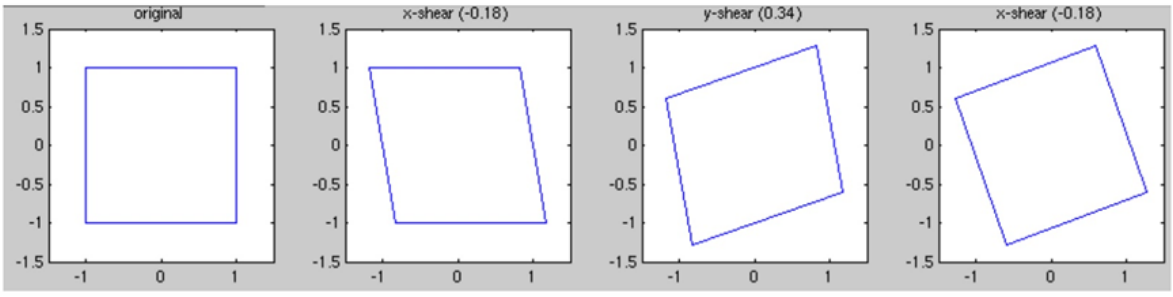
曾經的期中考題：

Q：如何旋轉一圖形？

A：利用多次剪力拉伸即可

Example

Let Θ=20°. Then α~-0.18 and β~0.34. Here you can see what a square looks like as the three shears are applied:



三角形表示法 ***

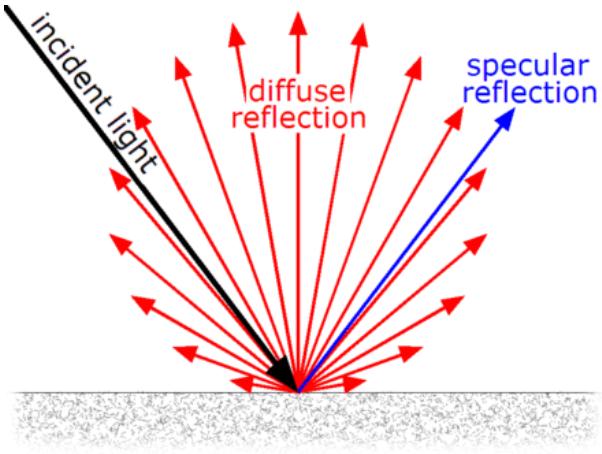
單一三角形會用 3 組 6 個數值表示，每一組表示其中一個頂點，而 6 個數值則代表：該點之『xyz 向量』，加上『xyz 的平面法向量(Surface Normal Vector，且是正交後的結果，故平方和為 1)』

若加入顏色，則會在前面再加 2 組 3 數值，第一組表正面顏色、第二組表反面顏色(朝向觀查者的那一面定為正面，反面通常會以是否燈光來表示)；3 數值則為其 RBG 數值(通常以 8bits 表示：255 為全白、0 為全黑)

以上方式的優點為 debug 容易，但缺點為資料量過大。
例：[補圖]

為解決多組三角形邊緣重覆資料的狀況，可以 Winged-edge Structure 表示(優缺點相反：資料量較少、但較難 debug)

Ambient Light 與 Diffuse Reflection 的差異：

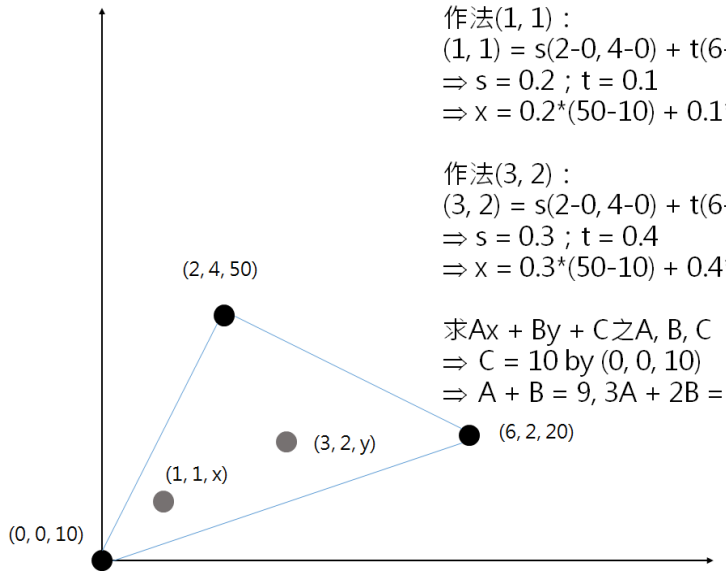


1. 上圖展示了一道光線(黑箭頭)照到物體表面，反射後產生的反射光，若等角度又射(藍箭頭)，則稱為 Specular Reflection；若有不等角度反射(紅箭頭)，則稱為 Diffuse Reflection。
2. Ambient Light 雖然在概念上與 Diffuse Reflection 類似，但其定義為：當光線在眾多物體表面不斷反射，進而產生高度難以計算之環境光源時，我們便稱此環境光源為 Ambient Light

平方反比定律(與公式中的 fatt 有關)：

1. 球狀(最常見)：距離加倍，光線亮度會衰減平方倍，在攝影上稱之為『平方反比定律』，常用在閃光燈的使用上(以上條件為光線為球狀)
2. 若光線改以柱狀、條狀，則光線是成反比、而非平方反比(聚光燈、日光燈可能接近此種方式)
3. 平行光：若是沒有能量衰減，能量並不會減少，則距離不影響亮度(理論上)

- flat shading：三角形的頂點沒有法向量，三角形整個面才有法向量，打光時整個三角形只呈現一種顏色。
- Gouraud shading：三角形的頂點都有各自的法向量，打光時三個頂點有各自的顏色，接著做雙線性內插（bilinear interpolation）來求得顏色，使整個三角形有漸層的顏色變化。
- Phong shading：三角形的頂點都有各自的法向量，先對三角形整個面作法向量的雙線性內插，接著打光來求整個三角形的顏色。



作法(1, 1)：
(1, 1) = s(2-0, 4-0) + t(6-0, 2-0) + (0, 0)
⇒ s = 0.2 ; t = 0.1
⇒ x = 0.2*(50-10) + 0.1*(20-10) + 10 = 19

作法(3, 2)：
(3, 2) = s(2-0, 4-0) + t(6-0, 2-0) + (0, 0)
⇒ s = 0.3 ; t = 0.4
⇒ x = 0.3*(50-10) + 0.4*(20-10) + 10 = 26

求Ax + By + C之A, B, C
⇒ C = 10 by (0, 0, 10)
⇒ A + B = 9, 3A + 2B = 16 => A = -2, B = 11