计算机图形学

第二章：光栅图形学算法

光栅图形学算法的研究内容

- 直线段的扫描转换算法
- 多边形的扫描转换与区域填充算法
- 直线裁剪算法
- 反走样算法
- 消隐算法

主要讲述的内容：

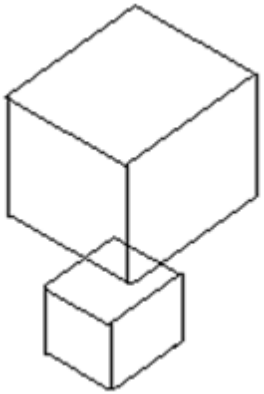
消隐的分类，如何消除隐藏线、隐藏面，主要介绍以下几个算法：

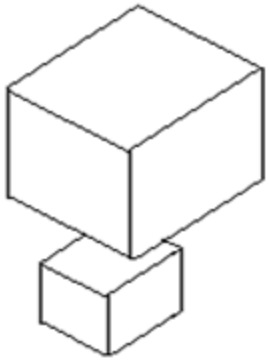
- Z缓冲区(Z-Buffer)算法
- 扫描线Z-buffer算法
- 区域子分割算法

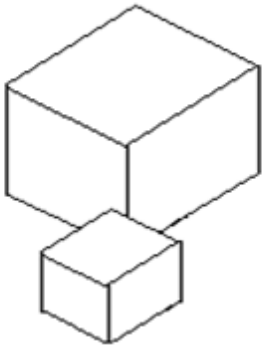
一、消隐

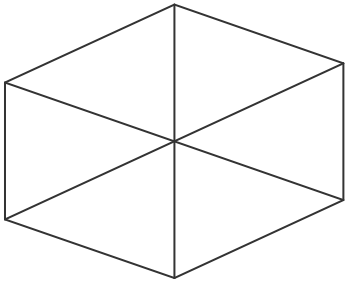
当我们观察空间任何一个不透明的物体时，只能看到该物体朝向我们的那些表面，其余的表面由于被物体所遮挡我们看不到

如果把可见和不可见的线都画出来，对视觉会造成多义性

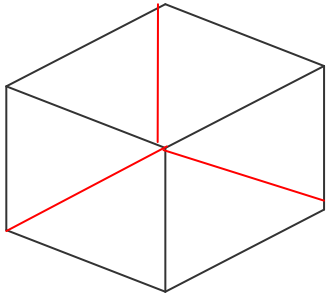




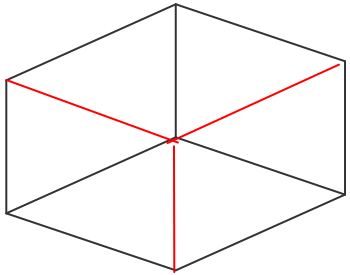




（a）



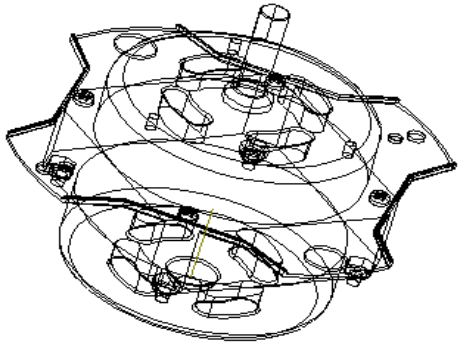
（b）



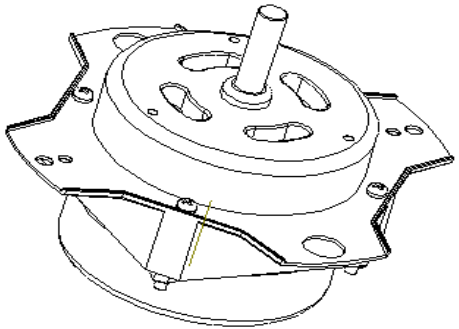
（c）

要消除二义性，就必须在绘制时消除被遮挡的不可见的线或面，习惯上称作消除隐藏线和隐藏面，简称为**消隐**。

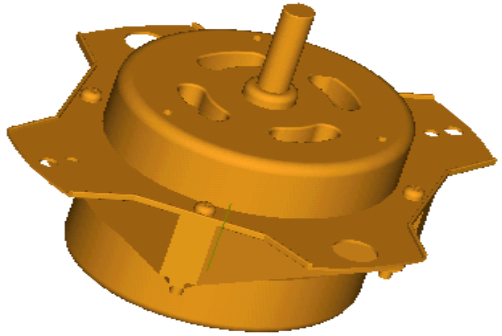
要绘制出意义明确的、富有真实感的立体图形，首先必须消去形体中的不可见部分，而只在图形中表现可见部分



线框图



消隐图

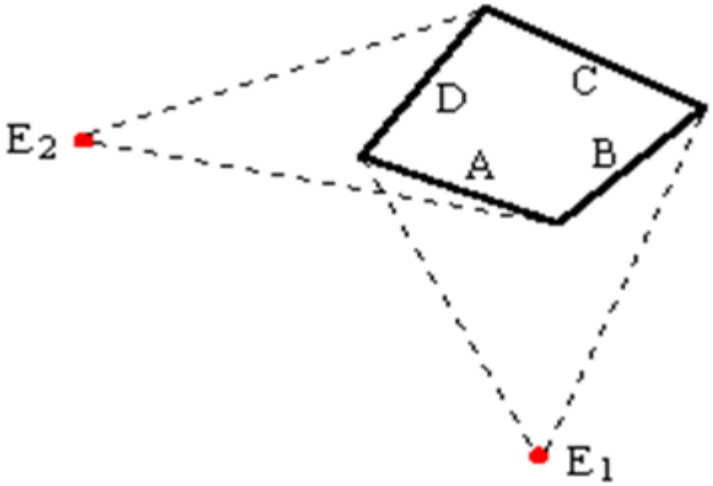


真实感图形

消隐包括消除“隐藏线”和“隐藏面”两个问题

到目前为止，虽然已有数十种算法被提出来了，但是由于物体的形状、大小、相对位置等因素千变万化，因此至今它仍吸引人们作出不懈的努力去探索更好的算法

消隐不仅与消隐对象有关，还与观察者的位置有关



消隐处理从原理上讲并不复杂，但是消隐处理的具体实现并不那么简单，它要求适当的算法及大量的运算。在60年代，消隐问题曾被认为是计算机图形学中的几大难题之一

二、消隐的分类

1、按消隐对象分类

（1）线消隐

消隐对象是物体上的边，消除的是物体上不可见的边

（2）面消隐

消隐对象是物体上的面，消除的是物体上不可见的面，通常做真实感图形消隐时用面消隐

2、按消隐空间分类

（1）物体空间的消隐算法

以场景中的物体为处理单元。假设场景中有k个物体，将其中一个物体与其余k-1个物体逐一比较，仅显示它可见表面以达到消隐的目的

此类算法通常用于线框图的消隐！

```
for（场景中的每一个物体）
  { 将该物体与场景中的其
    它物体进行比较，确定
    其表面的可见部分；
    显示该物体表面的可见
    部分；
  }
```

在物体空间里典型的消隐算法有两个： Roberts算法和光线投射法

Roberts算法数学处理严谨，计算量甚大。算法要求所有被显示的物体都是凸的，对于凹体要先分割成多个凸体的组合

Roberts算法基本步骤：

- 逐个的独立考虑每个物体自身，找出为其自身所遮挡的边和面（自消隐）；
- 将每一物体上留下的边再与其它物体逐个的进行比较，以确定是完全可见还是部分或全部遮挡（两两物体消隐）；
- 确定由于物体之间的相互贯穿等原因，是否要形成新的显示边等，从而使被显示各物体更接近现实

光线投射是求光线与场景的交点，该光线就是所谓的视线（如视点与像素连成的线）

一条视线与场景中的物体可能有许多交点，求出这些交点后需要排序，在前面的才能被看到。人的眼睛可以一目了然，但计算机做需要大量的运算

（2）图像空间的消隐算法

以屏幕窗口内的每个像素为处理单元。确定在每一个像素处，场景中的k个物体哪一个距离观察点最近，从而用它的颜色来显示该像素

```
for（窗口中的每一个像素）
  {确定距视点最近的物体，
   以该物体表面的颜色来显
   示像素；
  }
```

这类算法是消隐算法的主流！

因为最后看到的图像是在屏幕上的，所以就拿屏幕作为处理对象。针对屏幕上的像素来进行处理，算法的思想是围绕着屏幕的。对屏幕上每个象素进行判断，决定哪个多边形在该象素可见。