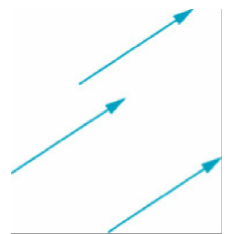
Scalar标量，系数，没有几何性质

Vector 向量，有大小magnitude, direction 方向

大小是两个绝对值



这些是相同的，向量没有起点

Point，空间中的一个坐标

两个点相减A-B可以制造一个指向A的向量

一个点+一个向量组成一个新向量（可以看成原点到这个点的向量+这个点为起点的向量）

坐标系

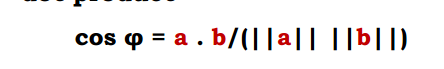
最基本的叫做orthonormal标准正交，三个basic vector垂直

x-y cartesian coordinate笛卡尔（就是XY平面坐标系）也是标准正交的一种

dot product

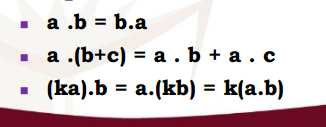


Dot product也叫scalar product，因为乘出来是个标量



It is used to find projection of a on b //projection投射

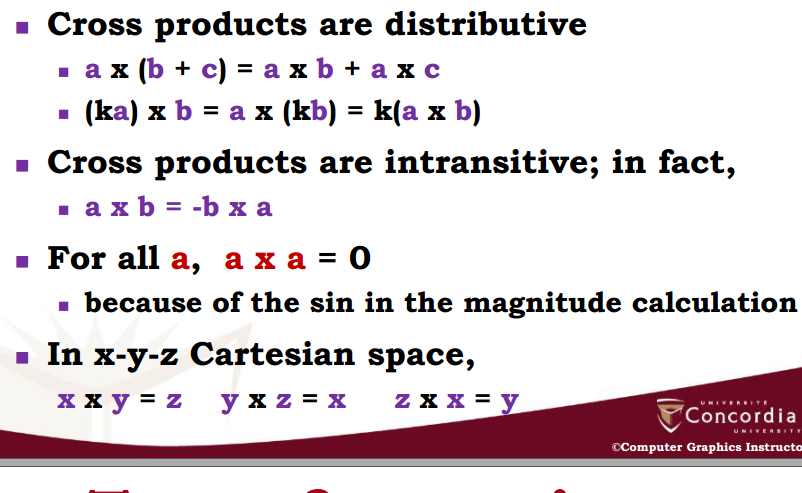




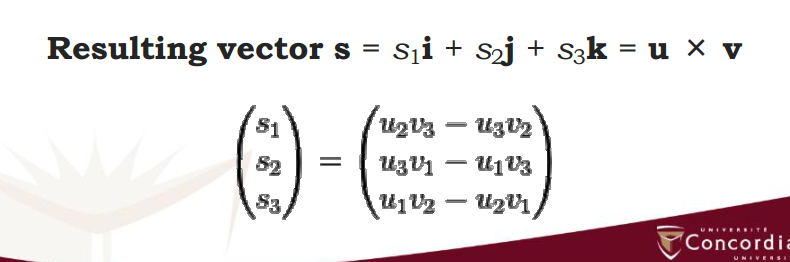
Cross product

only used for 3d vector

只对三维向量有效，大小就是这么求， 方向：垂直于两条向量

”

Axa=0因为夹角=0



怎么算

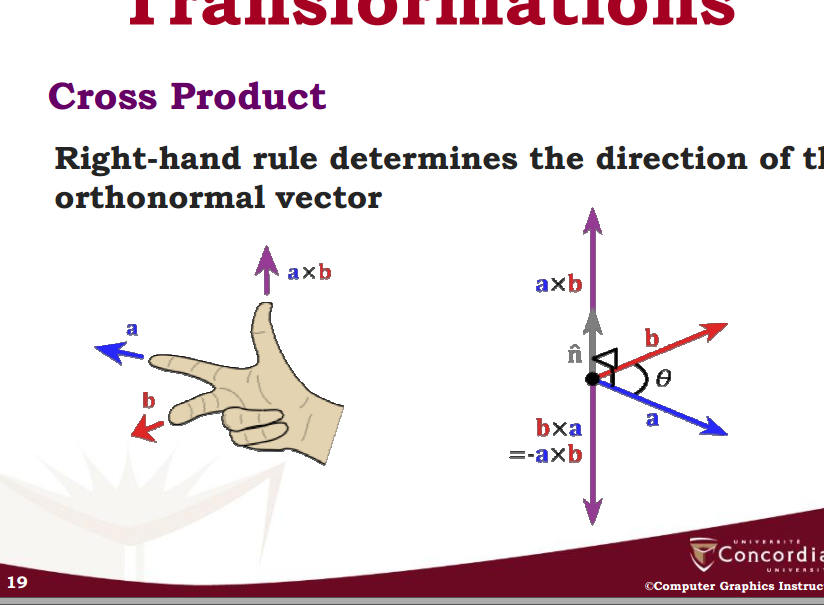
|S1 U1 V1|

|S2 U2 V2|

|S3 U3 V3|

然后切割就完事儿了，S2加负号

S1代表i，S2代表j，S3代表k



Axb方向永远是大拇指，倒过来加负数

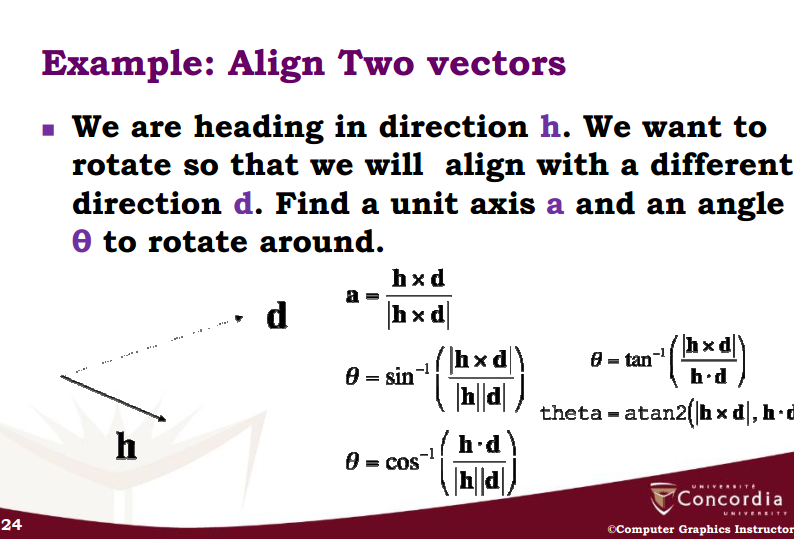
Normal vector

Normal vector就是垂直于面的一个vector，一个标准Normal //unit normal就是一个大小为一的normal vector

­­­­­­­­­­­­如果一个平面上面有三个点abc，那么就是这个平面的normal factor之一（两个向量），但是这三个点不能在一条直线上

利用一个unit normal来旋转一个向量

Ex



第一步不是公式，hxd做出一个垂直向量，因为是单位所以除了他的大小

两边同时加上绝对值

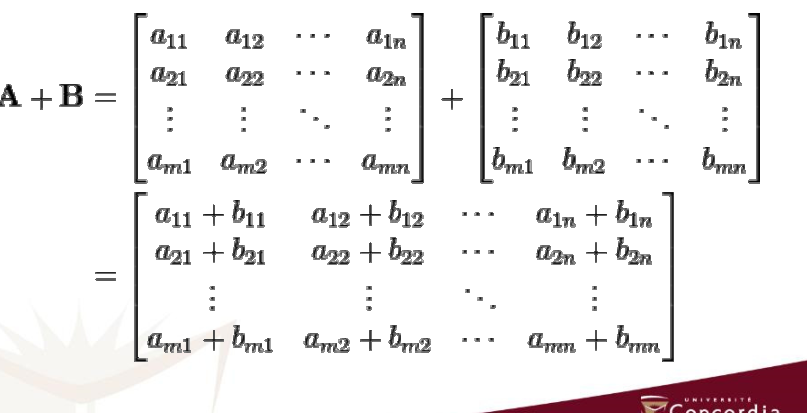
|HXD|=|H||D|SINO

SINO乘过去，=hxd|（a\*|h||d|），然后hxd/a实际上就是hxd的大小（x出来的东西除以单位向量）

Matrix

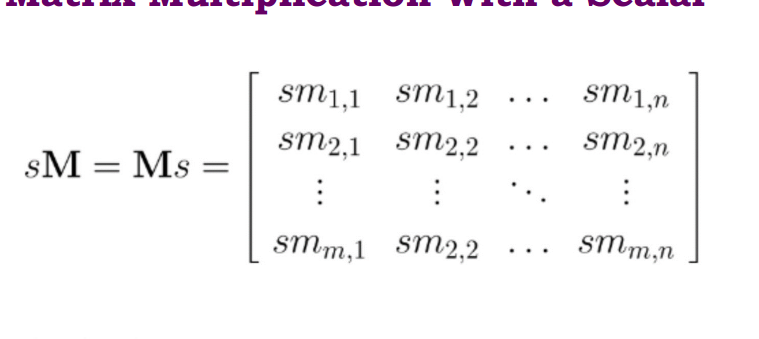
一组长方形数字

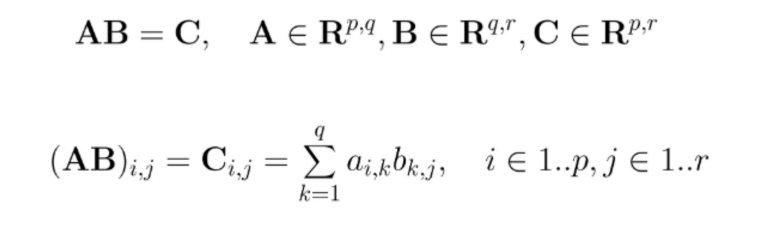
在图形里，通常M=n=4或m=n=3



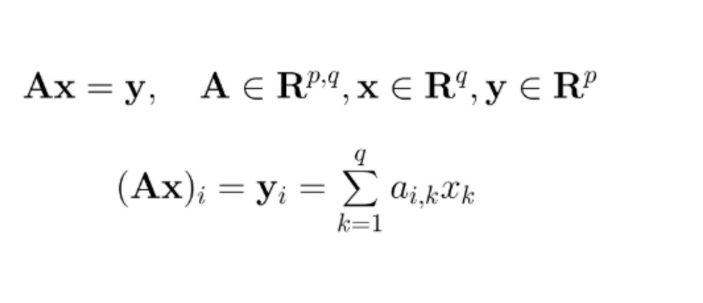
+法一定要行列数量一致，对应相加

乘法就是直接乘系数

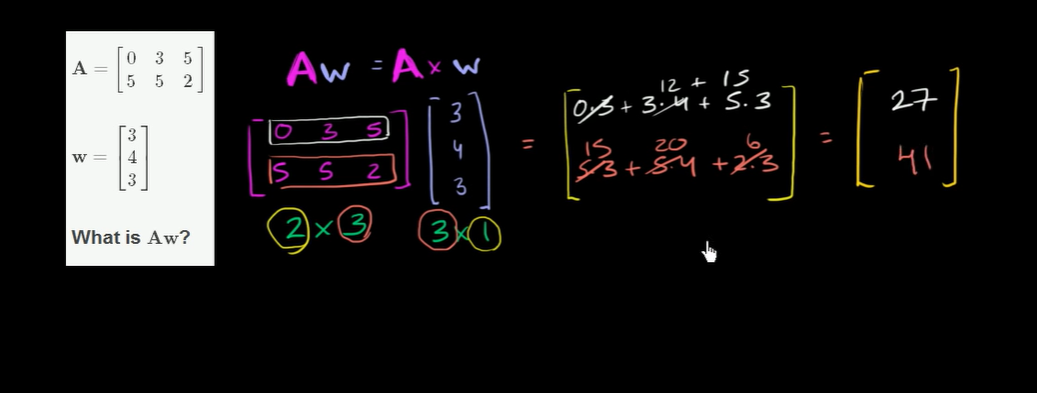




矩阵相乘，A要a行b列，B要b行C列，C的第A行第B列等于A的第A行所有列与B的第B列所有行对应相乘

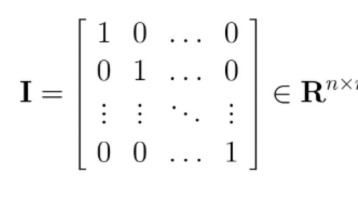


矩阵向量相乘

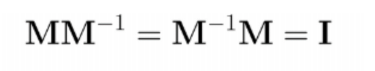


向量是竖着的矩阵

单位矩阵 identity matrix，一般用I表示



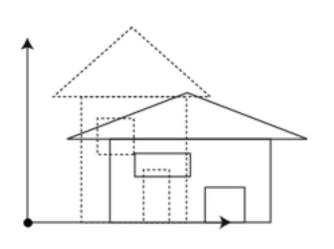
Matrix inverse，他与他的INVERSE✖起来是单位矩阵



linear transformations线性变换

我们通过让matrix相乘来完成transformation

1.scaling 缩放

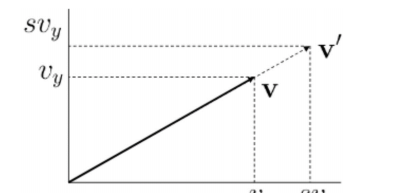
缩放有点难描述，有点像你改WINDOWS窗口大小，你可以随便改，但不可能改出一个菱形来

缩放分为两种

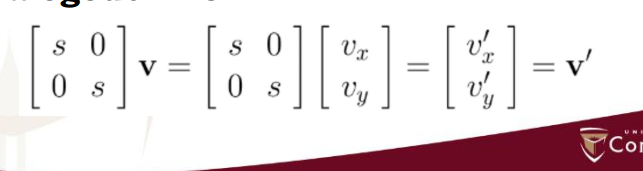
uniform scaling 均匀缩放

就是成比例缩放，

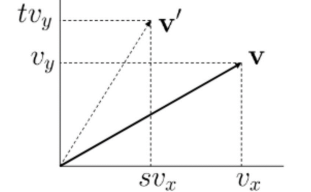
2D里面就是向量延长缩短

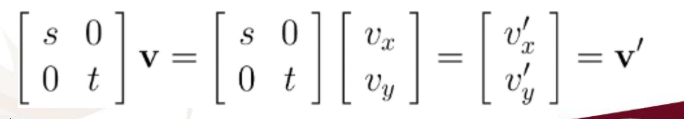


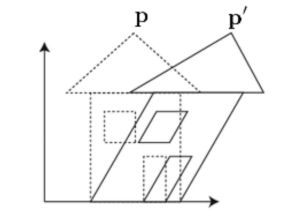
3D向量就是向量乘以相同的常亮



non-uniform

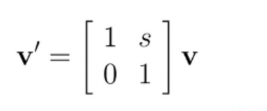
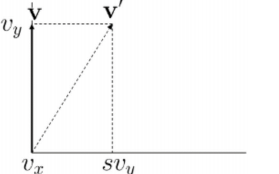
向量可以改方向长度

shearing

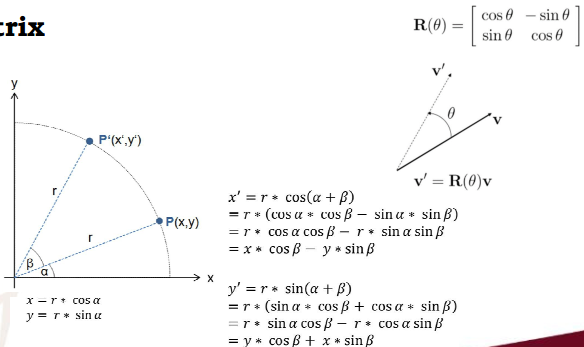


高度不变X乱变

只有X能变Y不能变

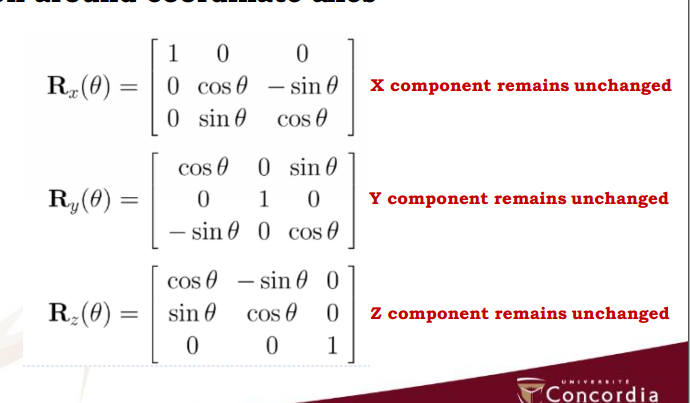


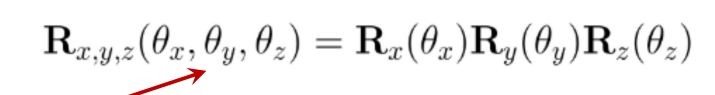
rotation 旋转



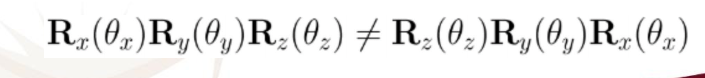
旋转你最后会发现与角A无关

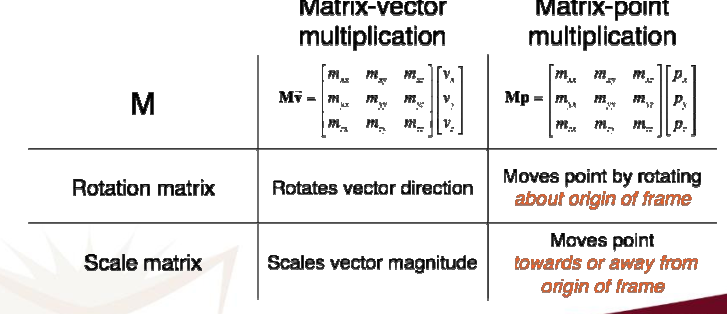
3D公式

注意着三个都要X(X,Y,Z)  


最后再把这三个连起来，

注意顺序

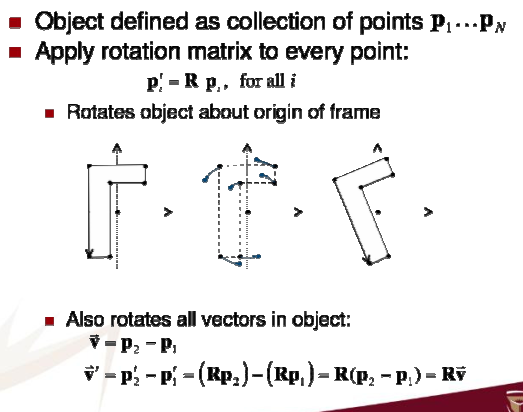




向量的旋转与拉伸都是通过×一个matrix， 旋转是改变向量方向，scale是改变向量大小

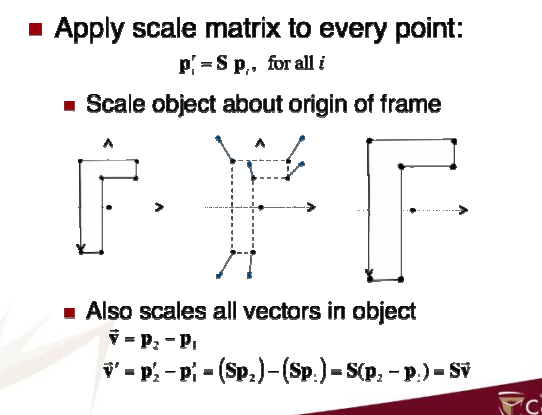
点的旋转与scale也是X一个matrix，旋转会让这个点随着这个frame的origin旋转，scale会沿着frame的Origin这条虚线靠近或远离

旋转一个object



Object实际上也是一组点的集合，你只要对每个点都旋转就成

你把它看成vector也一样，因为一个vector是两个点的差值



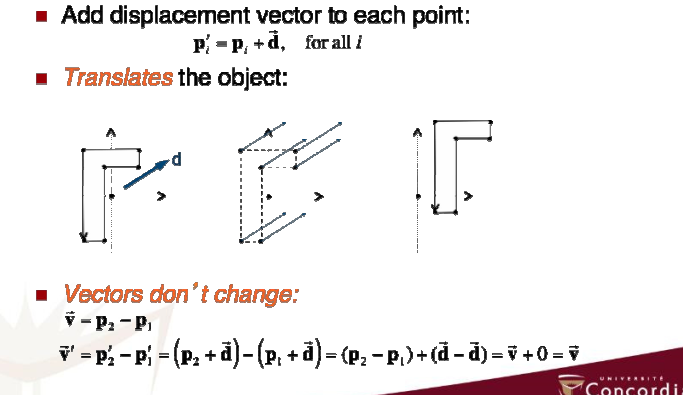
Scale也是如此

为什么方向不一样：因为每个点是绕着原点延伸或缩短

Vector同理

Translates 平移

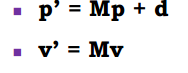
给所有点加上一个向量



对vector来说是没有意义的，因为这本质上是坐标的改变，而vector不考虑起点

所以对于Point与VECTOR来说,加入了平移以后，通用公式不一样

M代表着旋转以及拉伸变化，d代表displacement translate变化



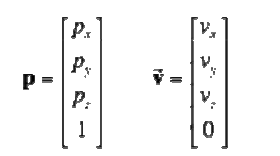
因此我们引入齐次坐标

Homogeneous coordinates齐次坐标

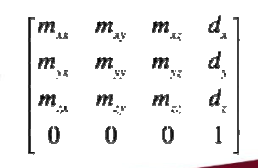
加上第四个坐标，

用1来代替3D VERTEX

用0来代替3D VECTOR

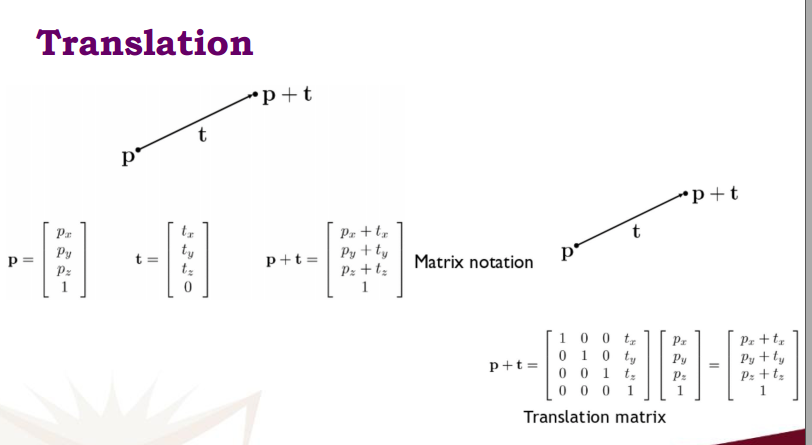


把M d融合成一个4x4 matrix



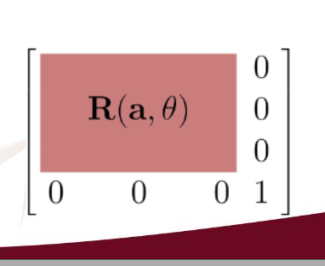
这样对于p来说，前面三个还是原来的，而又加上了d

而v就完全脱离了d

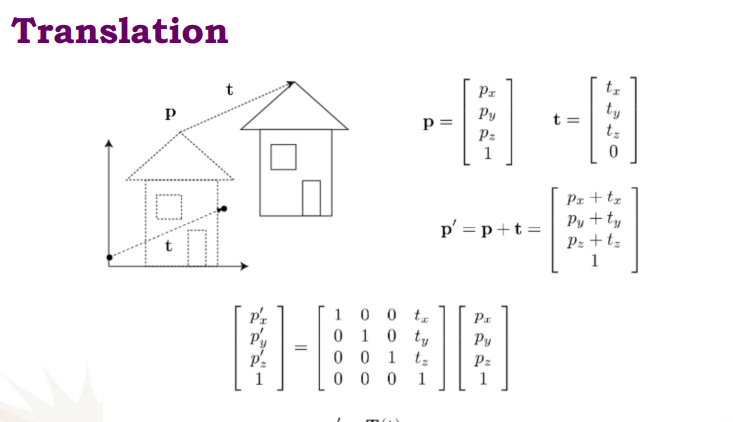


传统加法与matrix乘法

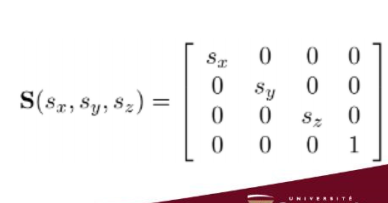
具体的·



Rotation 全部填0，右下角填1



Translation 斜着填1，最右好好填



Scaling， 全部0，右下角1’

Shear，

