坐标系对象介绍

概要：*在进行坐标系变换时，常常使用矩阵或更一般的张量运算。然而，矩阵过于数学化且含义不明确，而张量过于抽象且难以掌握和量化。本文介绍了一种名为“coord”的专门为坐标系变换设计的数学对象，以替代矩阵和张量运算。*

1. 背景

坐标系是最为抽象和困难的数学物理对象之一，而张量运算更加抽象，需要经过多年的训练才能掌握。因此，需要一种简化的方法来处理坐标系运算的难题。虽然矩阵常用于坐标系变换，但作为数学对象，矩阵并非为坐标系变换设计，且存在冗余和便捷性问题。而张量过于抽象，不仅难以掌握，计算机也难以量化。因此，需要一种专门为坐标系变换设计的数学对象。

2. 设计

本文介绍了一种名为“coord”的数学对象，基于群论思路设计了四则运算。该对象定义了三维空间中的坐标系，包括一个原点、三个方向轴和三个缩放分量，分别对应位移、旋转和缩放三种变换。该对象还提供了构造坐标系对象的方法，如通过三个轴或欧拉角来构造坐标系。此外，该对象还提供了乘法和除法运算，用于向量从本坐标系变换到父坐标系下以及从父坐标系投影到本坐标系。最后，本文介绍了该对象在常见使用场景下的应用，如世界坐标系与局部坐标系相互转化、多节点层级下使用等。

a) 坐标系结构体的定义：

- 三维空间中的坐标系由一个原点加上三个方向轴和三个缩放分量组成，

分别对应**位移、旋转、缩放**三种变换(C++版本):

```
struct coord
{
    vec3 ux, uy, uz;        // 三个单位基向量
    vec3 scale;             // 缩放
    vec3 o;                 // 原点位置
}
```

构造一个坐标系对象(C++版本)

通过三个轴

```
coord C(vec3 ux, vec3 uy, vec3 uz)

coord C(vec3 ux, vec3 uy);
```

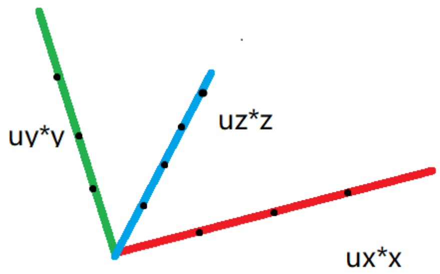
通过欧拉角

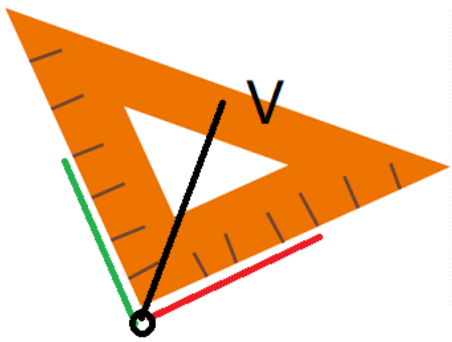
```
coord C(float angle, vec3 axis);

coord C(float pitch, float yaw, float roll);
```

（原点跟缩放可以直接设置）

b) 乘法：在某坐标系下定义向量





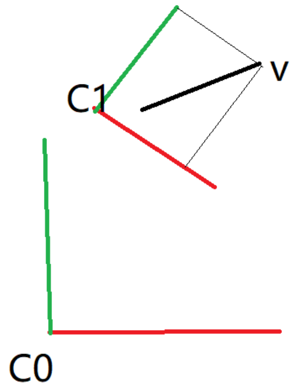
使用乘法实现向量从本坐标系变换到父坐标系下以及坐标系的合并运算

例如：V0 定义在世界坐标系 C0 下, V1 定义在 C1 坐标系

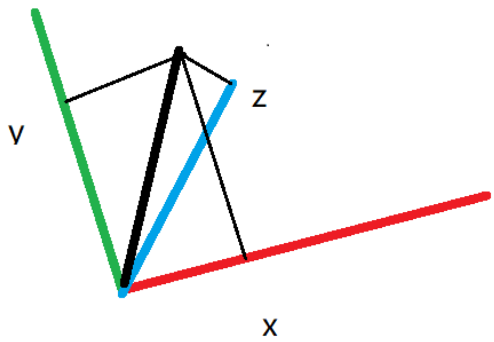
V0 = V1 * C1

C1,C2,C3 分别是三个场景节点下的本地坐标系，父->子方向：

V0 = V3 * C3 * C2 * C1 = V3 * (C3 * C2 * C1)



c) 除法：用某坐标系量度一个向量



使用除法从父坐标系投影到本坐标系，例如：

V0 定义在世界坐标系 C0 下, V1 定义在 C1 坐标系

V1 = V0 / C1, V0 = V1 * C1

V2 定义在 C2 坐标系下则：

V2 = V0 / C2 = V1 * C1 / C2

C1, C2, C3 分别是三个场景节点下的本地坐标系，父->子方向，

V3 = V0 / C1 / C2 / C3 = V0 / (C3 * C2 * C1)

d) 通常的使用场景

1. 比如一个世界空间下的向量 Vw，转化到本地坐标系 C 下

$$VL = Vw / C$$

反之： $Vw = VL * C$

2. 世界坐标系与局部坐标系相互转化

$$C = C3 * C2 * C1$$

$$Vw = VL * C, \quad VL = Vw / C$$

3. 多节点层级下使用

节点 5 坐标系下定义向量 V5，在节点 2 层次的父节点下：

$$V2 = V5 * C5 * C4 * C3$$

其中每个坐标系都是本地坐标系

反之： $V3 = V2 / C3 / C4 / C5$

4. 平级坐标系之间互换

```
C0{                    // C0 父坐标系下
    C1, C2;            // 两个平级子坐标系
}
```

向量从 C1 下转化为 C2 坐标系下

$$V2 = V1 * C1 / C2$$

5. 更多的运算

标量乘法：

C*k = C.scale * k

k 为浮点数字

四元数乘法

C*q = C.ux(q.angle, q.axis);

C.uy(q.angle, q.axis);

C.uz(q.angle, q.axis);

q 为四元数

与向量加法

C1 + o = C1'{C1.o + o, C1.u, C1.o}

o 为三维向量(vec3)

e) 坐标系加减法运算

在计算两个向量之差时：

DV = V1 – V2

设 V1 = v * C1, V2 = v * C2

DV = v * (C1 – C2)

这里定义一个加法运算用来进行向量基本加减法：

V1 +V2 = v * (C1 + C2)

= v * {C1.o+C2.o, C1.v + C2.v}.normalized

其中：C1.v = {C1.ux*C1.sx, C1.uy*C1.sy, C1.uz*C1.sz}

{ body }语法是临时构建结构，方便数学化代码

3. 坐标系求导

坐标系在空间上求导对应了三种常见形式：

梯度 $\nabla f = d\mathbf{F}(\mathbf{U} * df * \mathbf{C}uv) / d\mathbf{xyz}$

散度 $\nabla \cdot \mathbf{F} = d\mathbf{F} / d\mathbf{xyz} \cdot \mathbf{I}c$

旋度 $\nabla \times \mathbf{F} = d\mathbf{F} / d\mathbf{xyz} \times \mathbf{I}c$

4. 在微分几何领域的应用

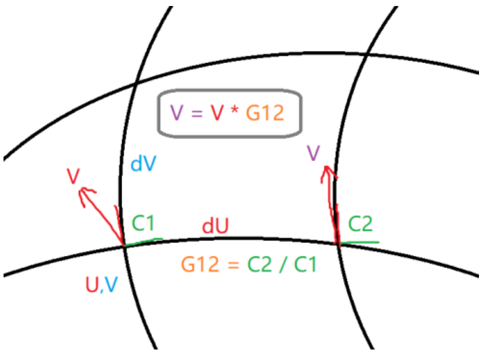
a) 向量的移动

定义一个自然坐标系(假设它是平直空间，向量可以随意移动而不变)下 V,在弯曲空间坐标系下观察 V，不同点上 V 是不同的，故而坐标系跟位置有关，取相邻两点(1), (2)点处有向量 V1, V2，对应坐标系 C1, C2，那么：

$V = V1 * C1 = V2 * C2 \Rightarrow$

$V2 = V1 * C1 / C2$，令 $G12 = C1 / C2 \Rightarrow$

$V2 = V1 * G12$



在任意微分向量方向上移动：

任意微分向量：$dW = Cw * dU$

所以在任意微分向量方向上移动：

$V2 = V1 * Gw = V1 * Gu * Cw$

其中：$Gu = G12 = C1 / C2$

单位微分向量 dU 方向上的两个坐标系对象之商。

b) 坐标系的指数(^)运算

坐标系在微分空间弧上平移时常伴随旋转，计算旋转对平移求导，由于旋转计算是使用除法（如：G12 = C1/C2），那么求导运算可采用指数位置进行，坐标系可表示为指数形式：C = e^(**V**),

取微分空间弧 d**u** 两端 **u**1, **u**2 处坐标系：

C1 = e^(**R1**)，C2 = e^(**R2**)

⇨ G21 = C2 / C1 = e^(**R**2 – **R**1), 令 d**R** = **R**2 – **R**1, d**u** = **u**2 - **u**1

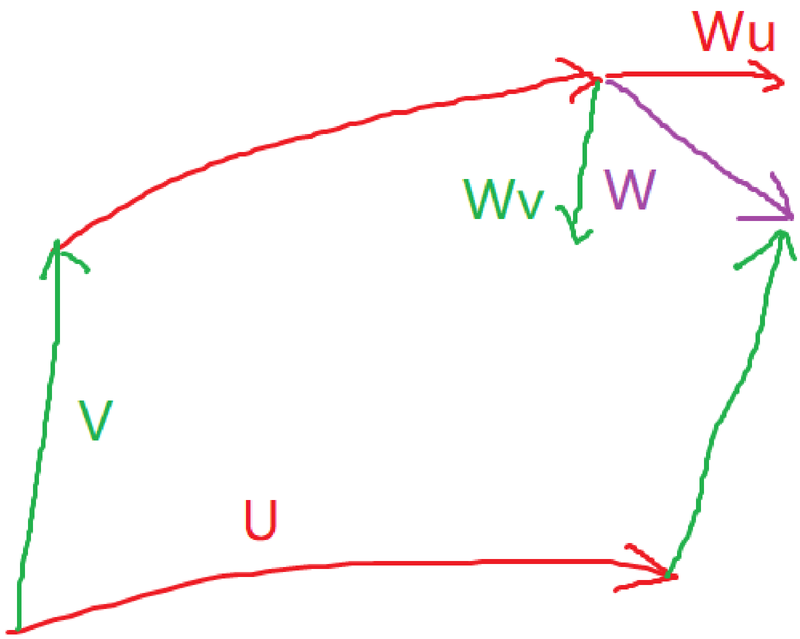
⇨ 导数 C’ = e^ (d**R** / d**u**)

⇨ G21 = C’ ^ d**u**, 当 d**u** = 1 时，G21 = C’

坐标系在曲面上平移可以使用公式:

C2 = C1 * C’ ^ (**u**2 – **u**1)

c) 曲率计算



在弯曲坐标系下，自然坐标系 x,y 轴的平行线投影得到的 u,v 曲线上 G12 分别在两个方向上对应 Gu,Gv，从(u1,v1)到(u2,v2) 计算两个路径的差别再加上修正项可得曲率公式为:

Ruv = Gu*Gv - Gv*Gu * Gu^Wu * Gv^Wv

其中：W = Wu + Wv = Gu * Wu + Gv * Wv

5. 与李群，李代数的结合

旋转矩阵 R 是李群的一个元素，我们的坐标系的乘法运算跟旋转矩阵是等价的，所以我们的坐标系 C 也是李群的一个元素，乘法为运算，0 元素为 ONE:

C∈{ONE, (*)}

向量叉乘对应的坐标系

v1 X v2 = v*C1 X (v * C2)

令 v = vec3.ONE 即 vec3(1,0,0)，那么 $C * v = C$，所以上式可以写为：

$$v1 \times v2 = v*(C1 \times C2)$$

我们定义一个坐标系叉乘运算,用李代数括号表达：

$$[C1, C2] = C1 * C2 - C2 * C1;$$

6. 结论

本文介绍了一种专门为坐标系变换设计的数学对象“coord”，其具有明确的含义和易于掌握的特点，并在常见使用场景下具有广泛的应用价值。坐标系对象能够简化局部空间下的线性运算，特别是在计算向量移动和旋转操作时，可以根据群论规则设计运算，形成统一的形式。坐标系对象适用于需要大量线性运算的约束计算等场景。通过坐标系对象，向量、矩阵和张量可以结合在一起，将李群和李代数对象统一成一个环，形成了统一而连贯的形式。