**作业05 AVX向量矩阵乘法**

背景知识：

在三维游戏中，车辆、房屋、人物等三维模型可均可表示为：若干个空间三角形，每个三角形包含3个边(线段)，每个线段包含起点和终点2个三维顶点，每个点是1个三维向量（x, y, z）。在C++中可用如下数据类型表示：

struct Face{

unsigned long long Lines[3];

};

struct Line{

unsigned long long Vectors[2];

};

struct Vector{

float V[3];

};

如下例：



一共包含4个顶点，6条边，4个三角形面。表示此模型需要：

Vector Vectors[4] = {

{0, 1, 2}, //Vector[0]的三个坐标xyz

{0, 1, 2}, //Vector[1]的三个坐标xyz

{0, 1, 2}, //Vector[2]的三个坐标xyz

{0, 1, 2} //Vector[3]的三个坐标xyz

};

Line Lines[6] = {

{0, 1}, //Vector[0]和Vector[1]构成的线0

{1, 2}, //Vector[1]和Vector[2]构成的线1

{2, 3}, //Vector[2]和Vector[3]构成的线2

{3, 0}, //Vector[3]和Vector[0]构成的线3

{0, 2}, //Vector[0]和Vector[3]构成的线4

{1, 3} //Vector[1]和Vector[3]构成的线5

};

Face Faces[4] = {

{0, 1, 4},

{2, 3, 4},

{0, 5, 3},

{1, 3, 5}

};

游戏中模型的移动和转动，是仅通过对Vectors中的数据进行坐标变换即可，而无需改变线和面的数据。变换公式如下：





为了将旋转、平移等操作归一化为向量与矩阵乘法，故将Vector扩展为4维。因旋转、平移、缩放、透视等三维变换可以先期合成矩阵，合成后矩阵功能分区如下图：



可以认为向量序列和给定矩阵相乘得到新向量序列是共性计算算子，并不需知道给定矩阵中各个位置取值代表的功能。为了更好的利用SIMD，更新Vector数据类型定义为齐次坐标：

struct Vector{

float V[4];

};

请充分利用AVX执行的并行性，编写程序实现批量向量与矩阵相乘，结果存储回矩阵

void VectorTrans(Vector VecSet [], float TransMatirx[4][4], unsigned int SetLen){

}

C++实现，提交文件名：学号\_05\_01.cpp