```
/*************************************************************************
                                    inlet model
1   profile term of inlet velocity
2   profile term of inlet k
3   profile term of inlet e
4   pollution
5   PFR
6   LMAA
7   VF  以VF讲解面数据的提取
*************************************************************************/

#include "udf.h"
#include "sg.h"

#define UH 7.84                     /*reference velocity*/
#define H 18.                       /*height of buildings*/
#define DELTA 250.                  /*doundary layer depth; DELTA=15*H */
#define ALPHA 0.25                  /*coefficient*/
#define Utau 0.305272               /*friction velocity*/
#define K 0.4                       /*von Karman constant*/
#define Cmu 0.09
#define M 0.00001
#define RHO 1.29
#define VOL 6220.8

/************************profile term of inlet velocity************************/

DEFINE_PROFILE(velocity_profile,t,i)
{
    real x[ND_ND];
    real h;
    face_t f;

    begin_f_loop(f,t)
    {
        F_CENTROID(x,f,t);
        h=x[2];

        if(h<=DELTA && h>=0)
        {
            F_PROFILE(f,t,i)=UH*pow(h/DELTA,ALPHA);
        }
        else
        {
```

```
                F_PROFILE(f,t,i)=UH;
            }
        }
        end_f_loop(f,t)
}

/*****************************profile term of inlet k*****************************/

DEFINE_PROFILE(k_profile,t,i)
{
        real x[ND_ND];
        real h;
        face_t f;

        begin_f_loop(f,t)
        {
            F_CENTROID(x,f,t);
            h=x[2];

            if(h<=DELTA)
            {
                F_PROFILE(f,t,i)=(Utau*Utau)/sqrt(Cmu);
            }
            else
            {
                F_PROFILE(f,t,i)=(Utau*Utau)/sqrt(Cmu);
            }

        }
        end_f_loop(f,t)
}

/*****************************profile term of inlet e*****************************/

DEFINE_PROFILE(e_profile,t,i)
{
        real x[ND_ND];
        real h;
        face_t f;

        begin_f_loop(f,t)
        {
            F_CENTROID(x,f,t);
            h=x[2];
```

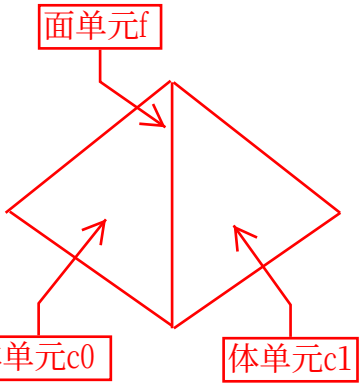
```
            if(h<=DELTA)
            {
                F_PROFILE(f,t,i)=(Utau*Utau*Utau)/(K*h);
            }
            else
            {
                F_PROFILE(f,t,i)=(Utau*Utau*Utau)/(K*DELTA);
            }

        }
        end_f_loop(f,t)
}

/************************pollution*************************/

DEFINE_SOURCE(Pullation_1,c,t,dS,eqn)
{
        real x[ND_ND];
        real source;
        C_CENTROID(x,c,t);

        if(x[0]>=151.2 && x[0]<=158.4 && x[1]>=81.0 && x[1]<=105.0 && x[2]<=18. && x[2]>=0.)
        {
            source = M;
        }
        else
        {
            source = 0;
        }

        dS[eqn]=0;

        return source;
}

/************************PFR*************************/

DEFINE_ON_DEMAND(PFR_1_udf)
{
        Domain *domain;
        Thread *t;
        cell_t c;
        real x[ND_ND];
```

```
	real cpt,cpa;
	real PFR;
	real SpatialAge;
	real vol;
	FILE *fp_pfr;
	fp_pfr=fopen("PFR.txt","a");
	domain=Get_Domain(1);
	thread_loop_c(t,domain)
	{
		begin_c_loop(c,t)
		{
			C_CENTROID(x,c,t);
			if(x[0]>=151.2 && x[0]<=158.4 && x[1]>=81.0 && x[1]<=105.0 && x[2]<=18. &&
x[2]>=0.)
			{
				cpt=cpt+C_YI(c,t,0)*C_VOLUME(c,t);
				vol=vol+C_VOLUME(c,t);
			}
			else
			{
				cpt=cpt;
				vol=vol;
			}
		}
		end_c_loop(c,t)
	}
	cpa=cpt/vol;
	PFR=(M*vol)/(cpa*RHO);
	SpatialAge=cpa/M;
	fprintf(fp_pfr,"%g\n",PFR);
	fclose(fp_pfr);
}

/*************************LMAA*****************************/

DEFINE_ON_DEMAND(LMAA_1_udf)
{
	Domain *domain;
	Thread *t;
	cell_t c;
	real x[ND_ND];
	real cpt,cpa;
	real SpatialAge;
	real vol;
```

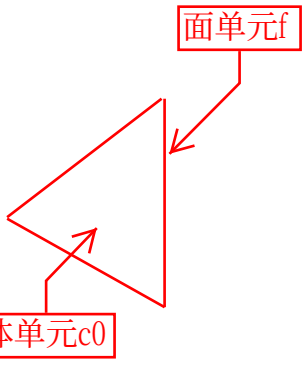
```
        FILE *fp_lmaa;
        fp_lmaa=fopen("LMAA.txt","a");
        domain=Get_Domain(1);
        thread_loop_c(t,domain)
        {
            begin_c_loop(c,t)
            {
                C_CENTROID(x,c,t);
                if(x[0]>=151.2 && x[0]<=158.4 && x[1]>=81.0 && x[1]<=105.0 && x[2]<=18. &&
x[2]>=0.)
                {
                    cpt=cpt+C_YI(c,t,0)*C_VOLUME(c,t);
                    vol=vol+C_VOLUME(c,t);
                }
                else
                {
                    cpt=cpt;
                    vol=vol;
                }
            }
            end_c_loop(c,t)
        }
```

1. 若面f为流场内部面，则f两侧有各有一个体单元，分别为c0和c1；
2. 若面f为流场边界面，则f只有一侧有一个体单元，为c0。
（t1和t2为c1和c2各自的线程）

内部面



边界面




```
}

/*************************VF*****************************/

DEFINE_ON_DEMAND(VF_1_udf)
{
    Domain *domain;
    Thread *t;
    face_t f;
    Thread *t0,*t1=NULL;
    cell_t c0,c1=-1;
    real delta_qp,qp;
    real x[ND_ND];
    real NV_VEC(A);
    real VF;
    real vol;
    real a;
    real u,v,w;
```

定义域指针

定义线程指针

定义面单元

定义面单元f两侧的体单元指针，初始值为NULL

定义面单元f两侧的体单元，初始值为-1

定义数组x

定义向量A（A为面单元f的面积向量，包含面积大小和法线方向）

```
real y;
real rho;
real xxx,yyy,zzz;
real xx,yy,zz;
FILE *fp_vf;
fp_vf=fopen("VF.txt","a");
domain=Get_Domain(1);
thread_loop_f(t,domain)
{
    begin_f_loop(f,t)
    {
        c0=F_C0(f,t);
        t0=F_C0_THREAD(f,t);
        F_AREA(A,f,t);
        a=NV_MAG(A);
        F_CENTROID(x,f,t);
        xxx=ROUND(x[0]*10.0);
        yyy=ROUND(x[1]*10.0);
        zzz=ROUND(x[2]*10.0);
        xx=xxx/10.0;
        yy=yyy/10.0;
        zz=zzz/10.0;
        if(xx==151.2 && yy>=81.0 && yy<=105.0 && zz>=0.0 && zz<=18.0)
        {
            if(BOUNDARY_FACE_THREAD_P(t))
            {
                if(NNULLP(THREAD_STORAGE(t,SV_Y)) &&
NNULLP(THREAD_STORAGE(t,SV_U)) && NNULLP(THREAD_STORAGE(t,SV_DENSITY)))
                {
                    u=F_U(f,t);
                    y=F_YI(f,t,0);
                    rho=F_R(f,t);
                }
                else
                {
                    u=C_U(c0,t0);
                    y=C_YI(c0,t0,0);
                    rho=C_R(c0,t0);
                }
            }
            else
            {
                c1 = F_C1(f,t);
                t1 = F_C1_THREAD(f,t);
```

打开VF.txt文件（没有的话会自动生成一个）

domain取整个流场区域

遍历流场域domain，得到面线程t（或者这样理解：遍历域domain中的所有面线程t）

遍历面线程t，得到面单元f（或者：遍历面线程中的所有面单元f）

体单元c0赋值

得到面积向量A

得到面积大小a

面f的重心坐标

对坐标进行简单的四舍五入操作（xx，yy，zz为四舍五入后得到的坐标）。为了防止出现坐标后带好几位小数点而无法精确匹配的情况，例如1.0000000001不等于1

想要提取数据的面的坐标区域

判断是否为边界面

是边界面

判断面f上储存的浓度、速度、密度值是否为空。若为空，则将c0上的值作为面f上的值。

不是边界面（是内部面）

内部面时，存在体单元c1，对其进行赋值。

```
                u=(C_U(c0,t0)+C_U(c1,t1))/2;
                y=(C_YI(c0,t0,0)+C_YI(c1,t1,0))/2;
                rho=(C_R(c0,t0)+C_R(c1,t1))/2;
            }
            delta_qp=delta_qp+rho*a*((fabs(u)+u)/2)*y;
        }
        if(xx==158.4 && yy>=81.0 && yy<=105.0 && zz>=0.0 && zz<=18.0)
        {
            if(BOUNDARY_FACE_THREAD_P(t))
            {
                if(NNULLP(THREAD_STORAGE(t,SV_Y))                    &&
NNULLP(THREAD_STORAGE(t,SV_U)) && NNULLP(THREAD_STORAGE(t,SV_DENSITY)))
                {
                    u=F_U(f,t);
                    y=F_YI(f,t,0);
                    rho=F_R(f,t);
                }
                else
                {
                    u=C_U(c0,t0);
                    y=C_YI(c0,t0,0);
                    rho=C_R(c0,t0);
                }
            }
            else
            {
                c1 = F_C1(f,t);
                t1 = F_C1_THREAD(f,t);
                u=(C_U(c0,t0)+C_U(c1,t1))/2;
                y=(C_YI(c0,t0,0)+C_YI(c1,t1,0))/2;
                rho=(C_R(c0,t0)+C_R(c1,t1))/2;
            }
            delta_qp=delta_qp+rho*a*((fabs(u)-u)/2)*y;
        }
        if(xx>=151.2 && xx<=158.4 && yy==81.0 && zz>=0.0 && zz<=18.0)
        {
            if(BOUNDARY_FACE_THREAD_P(t))
            {
                if(NNULLP(THREAD_STORAGE(t,SV_Y))                    &&
NNULLP(THREAD_STORAGE(t,SV_V)) && NNULLP(THREAD_STORAGE(t,SV_DENSITY)))
                {
                    v=F_V(f,t);
                    y=F_YI(f,t,0);
                    rho=F_R(f,t);
```

将面单元f两侧c0和c1的值平均后作为f的取值。

对得到的值进行操作，得到自己想要的数据

另一个想要提取数据的面的坐标区域。以下内容类似，不再赘述

```
                }
                else
                {
                    v=C_V(c0,t0);
                    y=C_YI(c0,t0,0);
                    rho=C_R(c0,t0);
                }
            }
            else
            {
                c1 = F_C1(f,t);
                t1 = F_C1_THREAD(f,t);
                v=(C_V(c0,t0)+C_V(c1,t1))/2;
                y=(C_YI(c0,t0,0)+C_YI(c1,t1,0))/2;
                rho=(C_R(c0,t0)+C_R(c1,t1))/2;
            }
            delta_qp=delta_qp+rho*a*((fabs(v)+v)/2)*y;
        }
        if(xx>=151.2 && xx<=158.4 && yy==105.0 && zz>=0.0 && zz<=18.0)
        {
            if(BOUNDARY_FACE_THREAD_P(t))
            {
                if(NNULLP(THREAD_STORAGE(t,SV_Y))                                    &&
NNULLP(THREAD_STORAGE(t,SV_V)) && NNULLP(THREAD_STORAGE(t,SV_DENSITY)))
                {
                    v=F_V(f,t);
                    y=F_YI(f,t,0);
                    rho=F_R(f,t);
                }
                else
                {
                    v=C_V(c0,t0);
                    y=C_YI(c0,t0,0);
                    rho=C_R(c0,t0);
                }
            }
            else
            {
                c1 = F_C1(f,t);
                t1 = F_C1_THREAD(f,t);
                v=(C_V(c0,t0)+C_V(c1,t1))/2;
                y=(C_YI(c0,t0,0)+C_YI(c1,t1,0))/2;
                rho=(C_R(c0,t0)+C_R(c1,t1))/2;
            }
```

```
                delta_qp=delta_qp+rho*a*((fabs(v)-v)/2)*y;
            }
            if(xx>=151.2 && xx<=158.4 && yy>=81.0 && yy<=105.0 && zz==18.0)
            {
                if(BOUNDARY_FACE_THREAD_P(t))
                {
                    if(NNULLP(THREAD_STORAGE(t,SV_Y))                    &&
NNULLP(THREAD_STORAGE(t,SV_W)) && NNULLP(THREAD_STORAGE(t,SV_DENSITY)))
                    {
                    w=F_W(f,t);
                    y=F_YI(f,t,0);
                    rho=F_R(f,t);
                    }
                    else
                    {
                    w=C_W(c0,t0);
                    y=C_YI(c0,t0,0);
                    rho=C_R(c0,t0);
                    }
                }
                else
                {
                    c1 = F_C1(f,t);
                    t1 = F_C1_THREAD(f,t);
                    w=(C_W(c0,t0)+C_W(c1,t1))/2;
                    y=(C_YI(c0,t0,0)+C_YI(c1,t1,0))/2;
                    rho=(C_R(c0,t0)+C_R(c1,t1))/2;
                }
                delta_qp=delta_qp+rho*a*((fabs(w)-w)/2)*y;
            }
        }
        end_f_loop(f,t)
    }
    vol=7.2*24.0*18.0;
    qp=vol*M;
    VF=1+(delta_qp/qp);
    fprintf(fp_vf,"%g\n",VF);
    fclose(fp_vf);
}
```

最后进行一些操作，得到自己想要的数据

将得到的数据输入到VF.txt文件中

关闭文件（如果不关闭则得不到数据）