```
# -*- coding: utf-8 -*-
import os
import math
import sys
import re
import string
import numpy as np
#originalは岩上さん
#2021.04.12
#コメントとプログラムの見かけだけを変更岩上さんのプログラムと全く同じ動作をするはず。大野
#x:1,y:2:,z:3の直方体のメッシュを蜘蛛の巣メッシュに変える

#pointsファイルのパス
file_pass = 'points'
#変更したpointsを書き出すファイル
write_file = open("spiderweb_points","w")
#write_file = open("../hotRoom/constant/polyMesh/spiderweb_points","w")
#pointsファイルのpointが始まる行数と終わる行数
Number_of_lines_starti=1
Number_of_lines_point_starti=21
Number_of_lines_points_num=19
Number_of_lines_point_endi=Number_of_lines_points_num #19

#立方体の長さ x,y,z cuboidの設定のみに使用
l=10 #x
m=10 #y
n=10 #z
cuboid_len=np.array([l/2.,m,n/2]) #細かく残す直方体の形がここで決定できる。
#cuboid_len=np.array([l/2.,m,n/2]) の場合は、残す直方体の比率は[1:2:1]となる。
#p0:蜘蛛の巣に拡大する時の中心点
p0=np.array([l/2,m/2,n/2])
#直方体の何割を細かく残すかを決める変数　,拡大率を増やしていく範囲 gamma ~ gamma2
gamma=0.3
gamma2=0.7
#拡大率 一番細かいメッシュが最大で何倍大きくなるかを決める変数
beta=3.0

#=============ohnoここから
#係数の計算
def _factor(v,r,R):
        if r  < v and v < R :
                return ((beta - 1.)/(R - r)) *(v - r) + 1.
        elif v <= r  :
                return 1.0
        elif v >= R :
                return beta
#max法による係数の計算 (オリジナル)
def maxMethod_factor(p):  #pは正規化されている点情報
        p_max=[math.fabs(p[0]),math.fabs(p[1]),math.fabs(p[2])]
        p_max=max(p_max)
        alpha=_factor(p_max,gamma,gamma2)
        return alpha
#==============ohnoここまで
#平行移動
def translation_point(p,ptra):
```

```
 return p-ptra
#正規化の関数
def nomalize_point(p,c):
 return p/c
#正規化の逆関数
def inverse_nomalize_point(p,c):
 return p*c
###############mail#############
fp = open(file_pass,'r')
i=Number_of_lines_starti
for line in fp.readlines()  :
    if i == Number_of_lines_points_num :
                print "number of points : ",line
                line=line.translate(string.maketrans(',()','   '))
                Number_of_lines_point_endi=int(line)+20
                i+=1
    elif i>=Number_of_lines_point_starti and i<=Number_of_lines_point_endi :
  line=line.translate(string.maketrans(',()','   '))
  line=line.split(' ')
  #print line
  x = float( line[1] )
  y = float( line[2] )
  z = float( line[3] )
#======ohno ここから
  p=np.array([x,y,z])
                p=translation_point(p,p0)
                p= nomalize_point(p,cuboid_len)
                alpha =maxMethod_factor(p)
                p=alpha*p
                p=inverse_nomalize_point(p,cuboid_len)
                data="({0:1.9f} {1:1.9f} {2:1.9f})¥n".format(p[0]+p0[0],
                                                             p[1]+p0[1],
                                                             p[2]+p0[2])
#======ohno ここまで
                write_file.write(data)
                i+=1
    else :
  i+=1
  write_file.write(line)
write_file.close()
```