

第一届VASPKIT·并行科技杯程序设计比赛之 异质结建模

Nan-nan Zhang, zhangnn18@mails.tsinghua.edu.cn

第一届VASPKIT·并行科技杯程序设计比赛之异质结建模

[介绍和使用方法](#)

[测试项目](#)

[代码结构](#)

第一部分，读取输入文件：

第二部分，判断角度的mismatch，寻找最佳角度：

第三部分，判断边长的mismatch，寻找最佳边长：

第四部分，构建异质结：

介绍和使用方法

脚本名称：`heteroJ.py`

参数：

控制异质结的允许角度（mismatchA）和长度（mismatchL）的mismatch阈值，

file1和file2的名称，使用的晶格参数是file1的，如果想用file2的晶格参数交换file1和file2两个文件即可，

调整真空层的大小：15 Angstrom，

调整层间距：3 Angstrom，

```
mismatchA = 0.001
mismatchL = 0.05
file1 = "03_black_P_POSCAR_new"
file2 = "03_Graphene_POSCAR"
vacuum = 15 # Angstrom
distance = 3 # Angstrom
```

脚本使用：

基于python3，调用pymatgen库，读取和输出POSCAR文件。

```
import pymatgen
```

确定python3有pymatgen库。

直接运行：

```
python heteroJ.py
```

输出信息如下：

Begin

Author: Nan-nan Zhang
zhangnn18@mails.tsinghua.edu.cn
First released time: 2019/07/23
Last modified time: 2019/07/23
Version: v0.01

Functions:

Do Heterostructures Junction build from Two POSCAR.

Usage:

Set the mismatch Angle, mismatch Length at first lines of scripts
Input the file name file1 and file2
Set the Vacuum and distance at first lines of scripts.

#####

read structure 1
read structure 2

#####

Do angle check
mismatch angle3: 0.3333333333333333

#####

Oh! mismatch_angle NOT statisfied. Finding Supercell. Please wait a few second
mismatch = 0.0

Find a new supercellgroup: [[-2, 0, 0], [0, 1, 0], [0, 0, 1]] [[-2, -1, 0], [0, 1, 0], [0, 0, 1]]

abc : 8.760000 3.310000 17.068001 angles: 90.000000 90.000000
90.000000

abc : 4.274223 2.467724 15.000000 angles: 90.000000 90.000000
90.000000

mismatch = 0.0

Find a new supercellgroup: [[-1, 0, 0], [0, 1, 0], [0, 0, 1]] [[-2, -1, 0], [0, 1, 0], [0, 0, 1]]

abc : 4.380000 3.310000 17.068001 angles: 90.000000 90.000000
90.000000

abc : 4.274223 2.467724 15.000000 angles: 90.000000 90.000000
90.000000

#####

Angle match Use [[-1, 0, 0], [0, 1, 0], [0, 0, 1]] [[-2, -1, 0], [0, 1, 0], [0, 0, 1]]

#####

#####

Do Length check

mismatch abc1: 0.0247476558897371

mismatch abc2: 0.3413169381989234

#####

Oh! mismatch_Length NOT statisfied. Finding Supercell. Please wait a few second
Find a supercellgroup:

mismatch abc1: 0.0247476558897371

mismatch abc2: 0.007983604865589994

#####

Angle and Length match Use [[-1, 0, 0], [0, 3, 0], [0, 0, 1]] [[-2, -1, 0], [0, 4, 0], [0, 0, 1]]

```
#####

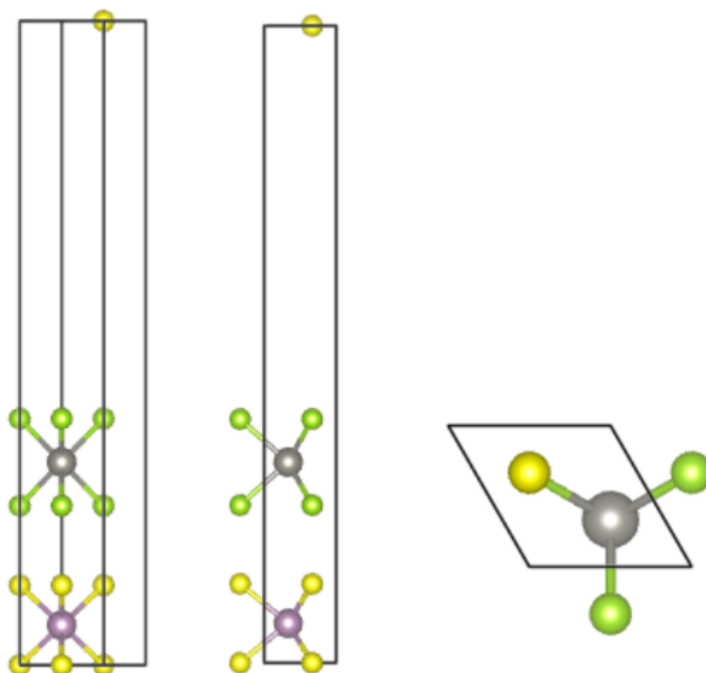
read new structure 1
abc   :   4.380000   9.930000  17.068001
angles:   90.000000   90.000000   90.000000
read new structure 2
abc   :   4.274223   9.870894  15.000000
angles:   90.000000   90.000000   90.000000

#####
output POSCAR03_black_P_POSCAR_new03_Graphene_POSCAR
#####

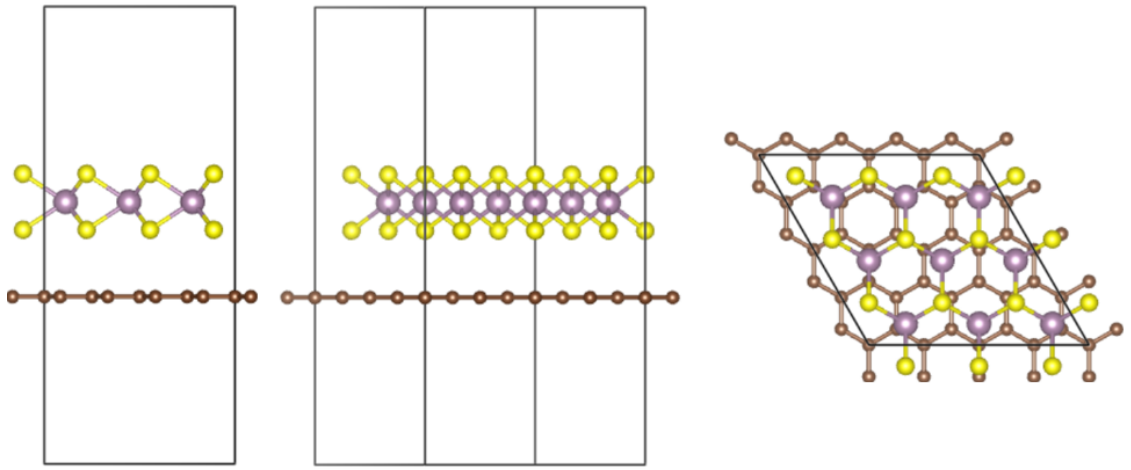
##### Summary #####
The final supercell for 03_black_P_POSCAR_new
[-1, 0, 0]
[0, 3, 0]
[0, 0, 1]
The final supercell for 03_Graphene_POSCAR
[-2, -1, 0]
[0, 4, 0]
[0, 0, 1]
Final mismatch of Angel: 0.0
Final mismatch of Length a and b: 0.0247476558897371   0.005987907478289188
Use 03_black_P_POSCAR_new as substrate
Distance between two slab is 3 Angstrom
Vacuum between two slab is 15 Angstrom
##### Summary #####
```

测试项目

1. POSCAR01_MoS2_POSCAR01_WSe2_POSCARwithVaccume

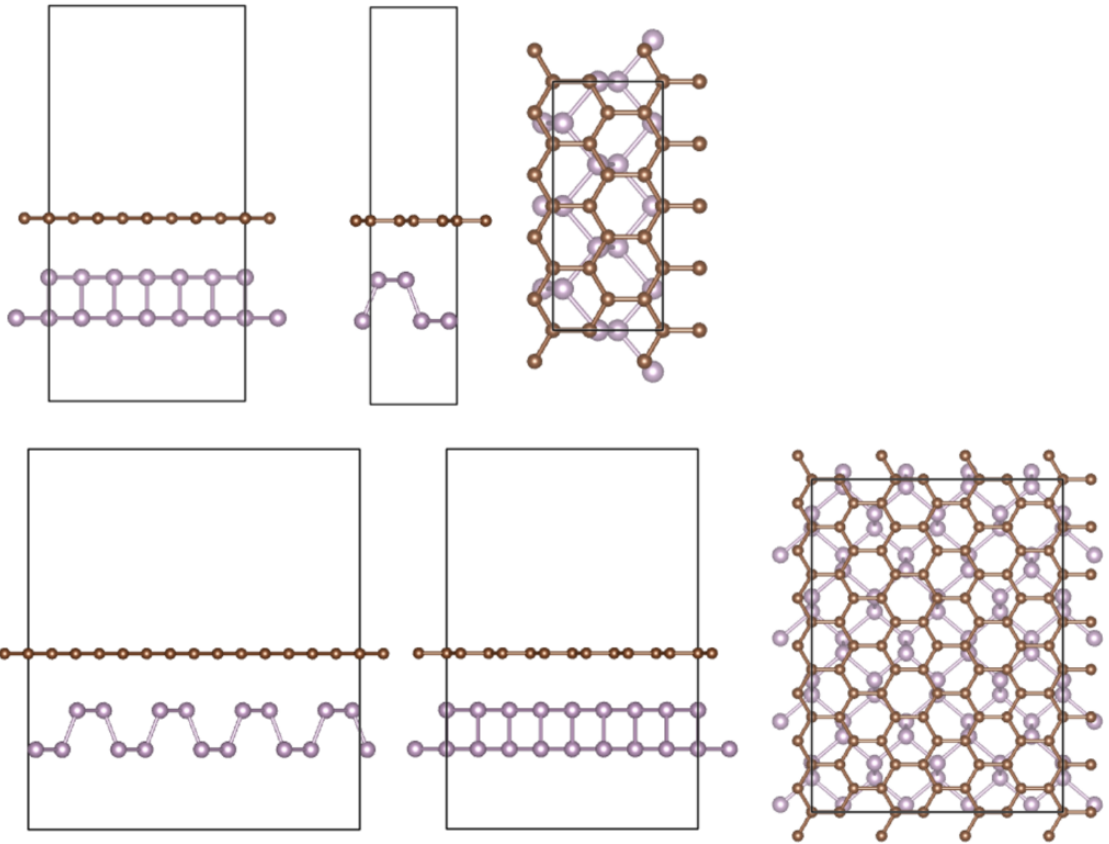


2. POSCAR02_Graphene_POSCAR02_MoS2_POSCARwithVaccume

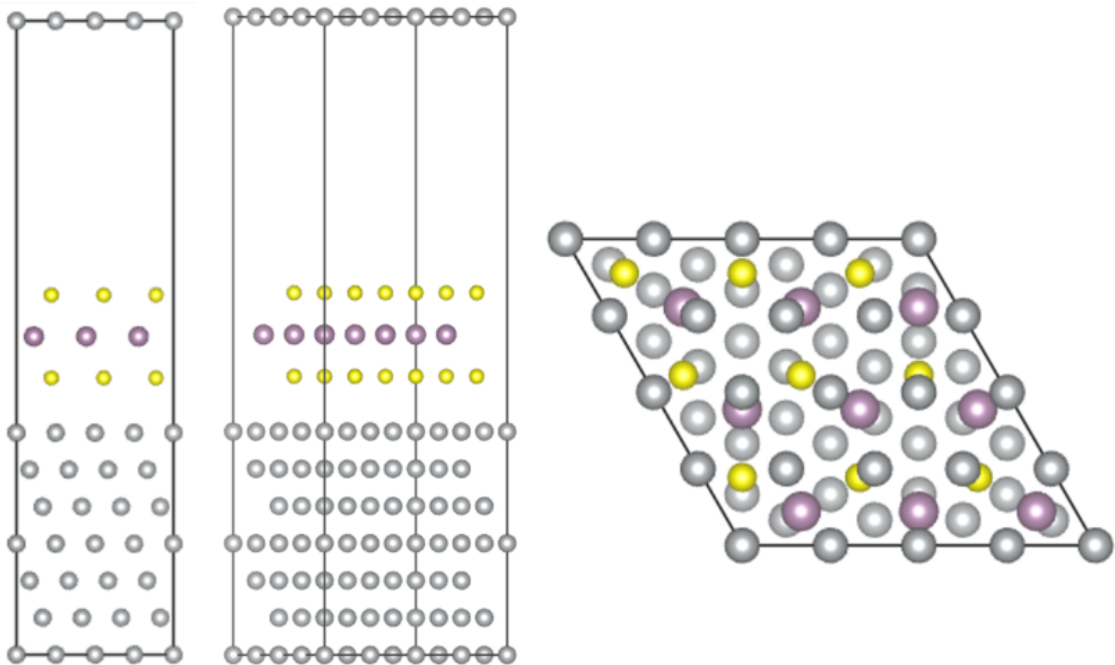


3. POSCAR03_black_P_POSCAR_new03_Graphene_POSCARwithVaccume

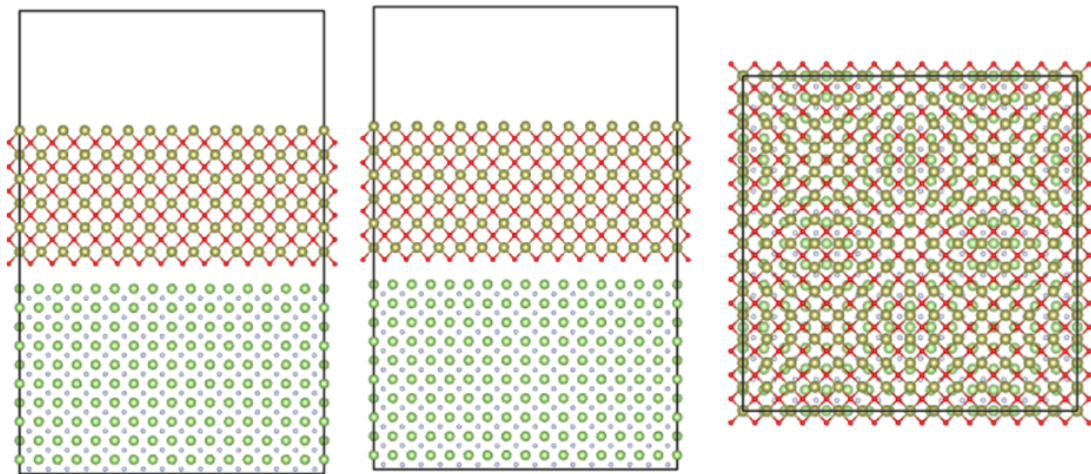
这个测试如果用默认的POSCAR生成第二种异质结，和原文对比发现两个材料的a, b的轴向正好反了，如果把其中一个材料的x, y对换以下，可得出和文献一样的结构。



4. POSCAR04_MoS2_POSCAR04_Ni111_POSCARwithVaccume



5. POSCAR05_GaN001_POSCAR05_HfO2001_POSCARwithVaccume



代码结构

第一部分，读取输入文件：

```
def readstruct(file, supercell=[1,1,1]):

    poscar1 = Poscar.from_file(file)      # 调用pymatgen读取POSCAR
    structure = poscar1.structure          # 整理成
    pymatgen.core.structure类的变量
    structure.make_supercell(supercell)
    #print("output lattice parameters:")
    #print(structure.lattice)              # 打印lattice参数(3*3
    matrix)
    strucPara = str(structure.get_sorted_structure()).split('\n')    # 提取夹角和
    边长

    #print(file, "\t parameters:")
    #print(strucPara[2])
    #print(strucPara[3])
    return strucPara[2], strucPara[3]
```

```

supercell1 = [[1, 0, 0], [0, 1, 0], [0, 0, 1]] # initial supercell1
supercell2 = [[1, 0, 0], [0, 1, 0], [0, 0, 1]] # initial supercell2
print("\nread structure 1")
abc1, angle1 = readstruct(file1, supercell1)
print("\nread structure 2")
abc2, angle2 = readstruct(file2, supercell2)

```

第二部分，判断角度的mismatch，寻找最佳角度：

```

print("Do angle check")
mismatch_angle3 = abs(angle1_clean[2] -
angle2_clean[2])/min((angle1_clean[2], angle2_clean[2]))
print("mismatch angle3:", mismatch_angle3)

if mismatch_angle3 < mismatchA:
    print('Good! mismatch_angle statisfied.')
else:
    print('Oh! mismatch_angle NOT statisfied. Finding Supercell. Please wait a
few second')
    judgesmallest = 100
    for a in range(-2, 3):
        for b in range(-2, 3):
            for i in range(-2, 3):
                for j in range(-2, 3):
                    if a == 0 or i == 0:
                        continue

                    abc1, angle1 = readstruct(file1, [[a, b, 0], [0, 1, 0], [0, 0,
1]])
                    abc2, angle2 = readstruct(file2, [[i, j, 0], [0, 1, 0], [0, 0,
1]])

                    abc1_clean = make_clean(abc1)
                    angle1_clean = make_clean(angle1)
                    abc2_clean = make_clean(abc2)
                    angle2_clean = make_clean(angle2)
                    mismatch_angle3 = abs(angle1_clean[2] - angle2_clean[2]) /
min(
                        (angle1_clean[2], angle2_clean[2]))
                    if mismatch_angle3 < mismatchA:
                        if abs(a)+abs(b)+abs(i)+abs(j) >= judgesmallest:
                            continue
                        judgesmallest = abs(a)+abs(b)+abs(i)+abs(j)
                        supercell1 = [[a, b, 0], [0, 1, 0], [0, 0, 1]]
                        supercell2 = [[i, j, 0], [0, 1, 0], [0, 0, 1]]
                        print("\nmismatch =", mismatch_angle3)
                        print("Find a new supercellgroup:", supercell1,
supercell2)

                        print(abc1, angle1)
                        print(abc2, angle2)
                        break
                    else:
                        continue

```

第三部分，判断边长的mismatch，寻找最佳边长：

```

print("Do Length check")

```

```

print("mismatch abc1:", mismatch_abc1)
print("mismatch abc2:", mismatch_abc2)

if mismatch_abc1 < mismatchL and mismatch_abc2 < mismatchL:
    print('Good! mismatch_Length statisfied.')
else:
    print('Oh! mismatch_Length NOT statisfied. Finding Supercell. Please wait a few second')
    judgeSmallest = 100
    for a in range(1, 9):
        for b in range(1, 9):
            for i in range(1, 9):
                for j in range(1, 9):
                    mismatch_abc1 = abs(abc1_clean[0]*a-
                    abc2_clean[0]*i)/min(abs(abc2_clean[0])*a,abs(abc2_clean[0]*i))
                    mismatch_abc2 = abs(abc1_clean[1]*b-
                    abc2_clean[1]*j)/min(abs(abc2_clean[1])*b,abs(abc2_clean[1]*j))
                    if mismatch_abc1 < mismatchL and mismatch_abc2 < mismatchL:
                        if abs(a) + abs(b) + abs(i) + abs(j) >= judgeSmallest:
                            continue
                        judgeSmallest = abs(a) + abs(b) + abs(i) + abs(j)
                        print("\nFind a supercellgroup:")
                        print("mismatch abc1:", mismatch_abc1)
                        print("mismatch abc2:", mismatch_abc2)
                        Lengthsuperlist = [a,b,i,j]

```

第四部分，构建异质结：

```

def makeJunction(file1, file2, supercell1, supercell2, vacuum=15, distance=2):
    poscar1 = Poscar.from_file(file1)    # 调用pymatgen读取POSCAR
    structure1 = poscar1.structure        # 整理成
    pymatgen.core.structure类的变量
    structure1.make_supercell(supercell1)
    poscar2 = Poscar.from_file(file2)    # 调用pymatgen读取POSCAR
    structure2 = poscar2.structure        # 整理成
    pymatgen.core.structure类的变量
    structure2.make_supercell(supercell2)

    # find the biggest z, find the smallest z
    zlist1 = []
    for i in range(len(structure1.species)):
        zlist1.append(float(structure1.frac_coords[i][-1]))
    zmax1,zmin1 = max(zlist1), min(zlist1)
    thick1 = zmax1-zmin1
    zlist2 = []
    for i in range(len(structure2.species)):
        zlist2.append(float(structure2.frac_coords[i][-1]))
    zmax2,zmin2 = max(zlist2), min(zlist2)
    thick2 = zmax2 - zmin2

    abc1, angle1 = readstruct(file1, supercell1)
    abc1_clean = make_clean(abc1)

    thicktot = (thick1 + thick2)*float(abc1_clean[-1]) + distance + vacuum

    for i in range(len(structure2.species)):

```

```

newcoordlist = []
#print(structure2.frac_coords[i])

newcoordlist.append(structure2.frac_coords[i][0])
newcoordlist.append(structure2.frac_coords[i][1])
newcoordlist.append(structure2.frac_coords[i][-1] + zmax1 - zmin2 +
distance/float(abc1_clean[-1]))
#print(structure2.frac_coords[i],newcoordlist)
structure1.append(species=structure2.species[i], coords=newcoordlist,
coords_are_cartesian=False)
print("\n#####")
print("output " + "POSCAR" + file1 + file2)
print("#####\n")
open("POSCAR" + file1 + file2,
"w").write(str(Poscar(structure1).get_string(direct=False)))
# modify z to vacuum thick
f = open("POSCAR" + file1 + file2, 'r', encoding='utf-8')
f_new = open("POSCAR" + file1 + file2 + "withVaccume", 'w', encoding='utf-
8')
count = 0
for line in f:
    count += 1
    if count == 5:
        line = "0.0 0.0 " + str(thicktot) + "\n"
        f_new.write(line)
        continue
    f_new.write(line)
f.close()
f_new.close()

makeJunction(file1, file2, supercell1, supercell2, vacuum, distance)

```