Code Listing for SurfaceDetect.xlsm MS Excel Macro for VSV Simulation post-processing

© Ocaya, r.o. — January 2018

```
1.  ' This program creates a single file with multiple .xyz type data to enable easy visual
    ization in VMD. Each .xyz type data becomes a frame when loaded into VMD. having one fi
    le allows a
2.  ' single file load into VMD which dramatically saves time in creating frames.
3.  '
4.  ' (c) Ocaya, R.O.
5.  ' Last modified 5 January 2018
6.  '
7.  Dim TotalAtoms, Adatoms As Integer
8.  Dim TotalSurfaceEnergy As Double
9.
10. Private Sub btnStart_Click()
11.   Dim i, j, k As Integer
12.   Dim t, x, y, z, v1, v2, v3 As Double
13.   Dim TimeStep, RowIndex As Long
14.   Dim oldname, filename As String
15.   Dim atomtype As String
16.
17.   TotalAtoms = Val(txtTotalAtoms.Text)
18.
19.   ' user specifies number of adatoms
20.   Adatoms = Val(txtAdatoms.Text)
21.
22.   ' find simulation time step from worksheet data
23.
24.   TimeStep = Sheet1.Cells(1 + TotalAtoms, 1) - Sheet1.Cells(1, 1)
25.   lblTimeStep.Caption = Str(TimeStep)     ' report the calculated value
26.
27.   ' count non-blank rows of the worksheet data. The last row index
28.   ' will give us the maximum simulation time, hence the number of
29.   ' rows in the data. This is then used to group data in the XYZ file
30.   i = 1
31.   While Sheet1.Cells(i, 1) <> ""
32.    i = i + 1
33.   Wend
34.   RowIndex = i - 2
35.
36.   ' report the simulation time found in the worksheet
37.   lblMaxSimTime.Caption = Str(Sheet1.Cells(RowIndex, 1))
38.
39.   atomtypeCu = "Cu"            ' we are interested in copper
40.   atomtypeNi = "Ni"            ' use this as a trick to highlight Cu as Ni
41.   filename = "out.xyz"
42.
43.   Open filename For Output As #1
44.   j = 0
45.   k = 1
46.   For i = 0 To RowIndex        ' default RowIndex = 37299
47.    If (i Mod TotalAtoms = 0) Then     ' count the number of particles in the file
48.     ' Close #1
```

```
49.     j = j + 10
50.     k = 1
51.     Print #1, TotalAtoms             ' this is the number of atoms in xyz file
52.     Print #1, "Created by Ocaya"
53.    End If
54.    t = Sheet1.Cells(i + 1, 1)
55.    x = Sheet1.Cells(i + 1, 2)
56.    y = Sheet1.Cells(i + 1, 3)
57.    z = Sheet1.Cells(i + 1, 4)
58.    v1 = Sheet1.Cells(i + 1, 5)
59.    v2 = Sheet1.Cells(i + 1, 6)
60.    v3 = Sheet1.Cells(i + 1, 7)
61.    ' Write #1, j, x, y, z, v1, v2, v3
62.    ' Write #1, Str(atomtype) + " " + Str(x) + " " + Str(y) + " " + Str(z)
63.    If k <= (TotalAtoms - Adatoms) Then '  default = 357 i.e. (total atoms-
   adatoms)=357
64.      Print #1, atomtypeCu, x, y, z     ' suppress commas by using Print rather than Wri
   te
65.     Else
66.      Print #1, atomtypeNi, x, y, z
67.    End If
68.    k = k + 1
69.   Next i
70.   Close #1
71.   Exit Sub
72. End Sub
73.
74. ' This subroutine detects a plane on the simulated output data and extracts velocity
75. ' and other parameters about the plane at a given time step. This allows the evolution

76. ' of the temperature and other parameters to be studied arbitrarily
77. '
78. Private Sub btnSurfaceDetect_Click()
79. Dim SimTimeLimit, UserSpecifiedTime, RowIndex As Long
80. Dim x1, y1, z1, x2, y2, z2, x3, y3, z3 As Double
81. Dim ux, uy, uz, vx, vy, vz, wx, wy, wz As Double
82. Dim sx, sy, sz As Double
83. Dim Wmag, Smag As Double  ' store the magnitudes of w and s vectors
84. Dim MinAngleRadians, DotProdRatio As Double
85. Dim i, j, StartIndex, StopIndex As Long
86. Dim Response
87.   ' count non-blank rows of the worksheet data. The last row index
88.   ' will give us the maximum simulation time, hence the number of
89.   ' rows in the data. This is then used to group data in the XYZ file
90.   i = 1
91.   While Sheet1.Cells(i, 1) <> ""
92.    i = i + 1
93.   Wend
94.   RowIndex = i - 2
95.
96.   ' now extract the vectors from user input
97.   x1 = Val(txt_x1)
98.   y1 = Val(txt_y1)
99.   z1 = Val(txt_z1)
100.         x2 = Val(txt_x2)
101.         y2 = Val(txt_y2)
102.         z2 = Val(txt_z2)
103.         x3 = Val(txt_x3)
104.         y3 = Val(txt_y3)
105.         z3 = Val(txt_z3)
106.
```

```
107.          ' now construct u and v vectors
108.          ux = x2 - x1
109.          uy = y2 - y1
110.          uz = z2 - z1
111.
112.          vx = x3 - x1
113.          vy = y3 - y1
114.          vz = z3 - z1
115.
116.          ' construct w vector by cross product w=u^v
117.          wx = uy * vz - vy * uz
118.          wy = uz * vx - vz * ux
119.          wz = ux * vy - vx * uy
120.
121.          lblCross.Caption = "(" + Str(Format(wx, "#.0000")) + ", " + Str(Format(wy, "#.
    0000")) + ", " + Str(Format(wz, "#.0000")) + ")"
122.          Wmag = (wx ^ 2 + wy ^ 2 + wz ^ 2) ^ 0.5
123.
124.         SimTimeLimit = Str(Sheet1.Cells(RowIndex, 1))
125.         UserSpecifiedTime = Val(txtSpecifyTimeStep.Text)
126.
127.         If UserSpecifiedTime > SimTimeLimit Then
128.            Response = MsgBox("Specified timestep exceeds available simulation time", vb
    OKCancel, "Time step specification error")
129.         Else
130.         ' now locate and store parameters of all atoms on the specified plane at the ti
    me step
131.         ' specified by the user
132.           i = 1
133.          While Sheet1.Cells(i, 1) <> UserSpecifiedTime
134.           i = i + 1
135.          Wend
136.
137.          lblSimTime.Caption = Str(i)
138.
139.          ' now calculate the worst case tolerance of normalized dot product of w and s
    to determine
140.          ' the farthest location the examined atom can be to be considered still on the
     plane.
141.
142.          MinAngleRadians = Cos(WorksheetFunction.Pi * Val(txtMinAngle.Text) / 180)
143.          TotalAtoms = Val(txtTotalAtoms.Text)
144.          StartIndex = i
145.          StopIndex = StartIndex + TotalAtoms - 1 ' the -
    1 corrects the index to last atom
146.
147.          lblStopIndex.Caption = Str(StopIndex)
148.
149.          ' now locate all atoms that lie on the plane at the given timestep
150.          j = 1
151.
152.          For i = StartIndex To StopIndex
153.           ' now calculate the vector due to the arbitrary point P
154.           sx = Sheet1.Cells(i, 2) - x1     ' calculate sx = xi - x1
155.           sy = Sheet1.Cells(i, 3) - y1     ' calculate sy = yi - y1
156.           sz = Sheet1.Cells(i, 4) - z1     ' calculate sz = zi - z1
157.           Smag = (sx ^ 2 + sy ^ 2 + sz ^ 2) ^ 0.5
158.
159.           '
160.           DotProdRatio = (wx * sx + wy * sy + wz * sz) / (Wmag * Smag)
```

```
161.          If Abs(DotProdRatio) < MinAngleRadians Then ' if the atom is close enough to
    the plane
162.            Sheet2.Cells(j, 1) = Sheet1.Cells(i, 2) ' copy the x-
    position of atom into new sheet
163.            Sheet2.Cells(j, 2) = Sheet1.Cells(i, 3) ' copy the y-
    position of atom into new sheet
164.            Sheet2.Cells(j, 3) = Sheet1.Cells(i, 4) ' copy the z-
    position of atom into new sheet
165.
166.            ' Sheet2.Cells(j, 4) = Sheet1.Cells(i, 5) ' copy the x-
    velocity of atom into new sheet
167.            ' Sheet2.Cells(j, 5) = Sheet1.Cells(i, 6) ' copy the y-
    velocity of atom into new sheet
168.            ' Sheet2.Cells(j, 6) = Sheet1.Cells(i, 7) ' copy the z-
    velocity of atom into new sheet
169.           v1 = Sheet1.Cells(i, 5)
170.           v2 = Sheet1.Cells(i, 6)
171.           v3 = Sheet1.Cells(i, 7)
172.
173.           ' calculate velocity magnitude and put into next column
174.           Sheet2.Cells(j, 4) = (v1 ^ 2 + v2 ^ 2 + v3 ^ 2) ^ 0.5
175.
176.           Sheet2.Cells(j, 5) = i ' include index for debugging only
177.           lblCross.Caption = "(" + Str(Format(sx, "#.0000")) + ", " + Str(Format(sy, "
    #.0000")) + ", " + Str(Format(sz, "#.0000")) + ")"
178.
179.           j = j + 1  ' increment line counter for new worksheet
180.          End If
181.         Next i
182.        End If
183.       End Sub
184.
185.       Private Sub cmdSurfaceEnergy_Click()
186.       Dim i, k, RowIndex As Long
187.       Dim SpecifiedTime, StopTime, TimeStep As Double
188.         i = 1
189.         While Sheet1.Cells(i, 1) <> ""
190.          i = i + 1
191.         Wend
192.         RowIndex = i - 2
193.
194.         StartTime = Sheet1.Cells(1, 1)
195.         StopTime = Sheet1.Cells(RowIndex, 1)
196.         TimeStep = StartTime
197.
198.         k = 1
199.         ' For SpecifiedTime = StartTime To StopTime Step TimeStep
200.           For SpecifiedTime = 0.5 To 205.5 Step TimeStep
201.
202.          Sheet26.Cells(k, 1) = k
203.          Sheet26.Cells(k, 2) = SpecifiedTime
204.          Sheet26.Cells(k, 3) = SurfaceEnergy(SpecifiedTime)
205.          k = k + 1
206.
207.         Next SpecifiedTime
208.
209.       End Sub
210.
211.       Private Function SurfaceEnergy(ByVal SpecifiedTime As Double) As Double
212.       Dim SimTimeLimit As Long
213.       Dim i, j, StartIndex, StopIndex As Long
```

```
214.        Dim x1, y1, z1, x2, y2, z2, x3, y3, z3 As Double
215.        Dim ux, uy, uz, vx, vy, vz, wx, wy, wz As Double
216.        Dim sx, sy, sz As Double
217.        Dim Wmag, Smag As Double  ' store the magnitudes of w and s vectors
218.        Dim MinAngleRadians, DotProdRatio As Double
219.        Dim totEnerg As Double
220.
221.          ' now extract the vectors from user input
222.          x1 = Val(txt_x1)
223.          y1 = Val(txt_y1)
224.          z1 = Val(txt_z1)
225.          x2 = Val(txt_x2)
226.          y2 = Val(txt_y2)
227.          z2 = Val(txt_z2)
228.          x3 = Val(txt_x3)
229.          y3 = Val(txt_y3)
230.          z3 = Val(txt_z3)
231.
232.          ' now construct u and v vectors
233.          ux = x2 - x1
234.          uy = y2 - y1
235.          uz = z2 - z1
236.
237.          vx = x3 - x1
238.          vy = y3 - y1
239.          vz = z3 - z1
240.
241.          ' construct w vector by cross product w=u^v
242.          wx = uy * vz - vy * uz
243.          wy = uz * vx - vz * ux
244.          wz = ux * vy - vx * uy
245.
246.          Wmag = (wx ^ 2 + wy ^ 2 + wz ^ 2) ^ 0.5
247.
248.          i = 1
249.          ' do the calculations only at the time specified in the argument list
250.          While Sheet1.Cells(i, 1) <> SpecifiedTime
251.           i = i + 1
252.          Wend
253.
254.          MinAngleRadians = Cos(WorksheetFunction.Pi * Val(txtMinAngle.Text) / 180)
255.          TotalAtoms = Val(txtTotalAtoms.Text)
256.          StartIndex = i
257.          StopIndex = StartIndex + TotalAtoms - 1 ' the -
    1 corrects the index to last atom
258.
259.          totEnergy = 0
260.          ' now locate all atoms that lie on the plane at the given timestep
261.          j = 1
262.
263.          For i = StartIndex To StopIndex
264.           ' now calculate the vector due to the arbitrary point P
265.           sx = Sheet1.Cells(i, 2) - x1     ' calculate sx = xi - x1
266.           sy = Sheet1.Cells(i, 3) - y1     ' calculate sy = yi - y1
267.           sz = Sheet1.Cells(i, 4) - z1     ' calculate sz = zi - z1
268.           Smag = (sx ^ 2 + sy ^ 2 + sz ^ 2) ^ 0.5
269.
270.           ' if Smag=0 then you are testing the same point, take a smart guess
271.           If Smag = 0 Then DotProdRatio = 0.00001 Else DotProdRatio = ((wx * sx + wy *
    sy + wz * sz) / (Wmag * Smag))
272.
```

```
273.          If Abs(DotProdRatio) < MinAngleRadians Then ' if the atom is close enough to
    the plane
274.            v1 = Sheet1.Cells(i, 5)
275.            v2 = Sheet1.Cells(i, 6)
276.            v3 = Sheet1.Cells(i, 7)
277.            ' find square of velocity magnitude; it is proportional to surface energy
278.            totEnerg = totEnerg + (v1 ^ 2 + v2 ^ 2 + v3 ^ 2)
279.          End If
280.         Next i
281.        ' return the calculated surface energy for
282.        ' detected atoms at the stated time step
283.        SurfaceEnergy = totEnerg
284.       End Function
285.
286.       Private Sub UserForm_Initialize()
287.        txtAdatoms.Text = 1
288.        txtTotalAtoms.Text = 2281
289.        TotalAtoms = Val(txtTotalAtoms.Text)
290.       End Sub
```