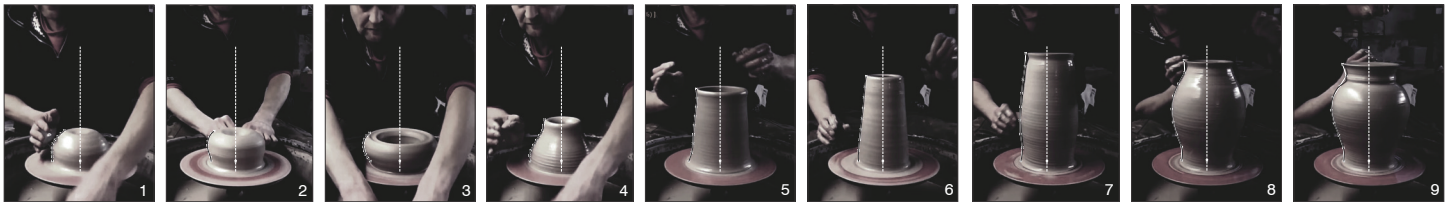
SYSTEM STALKER LAB: POTTERY
CODING SAMPLE

Generative Form-finding | Cambridge, Ontario | Group Academic Work: Di Wang | Sept '14




```
import math
import rhinoscriptsyntax as rs
import random

#Transformation1
T1AXtopDist=-17.4
T1AXbutDist=10.3
T1AXpkDist=1.4

T1AYtopDist=-1.4
T1AYbutDist=0
T1AYpkDist=-4.8

T1BXtopDist1=-5.1
T1BXtopDist2=-22.3
T1BXbutDist1=-4.9
T1BXbutDist2=-12.1
T1BXpkDist1=-3
T1BXpkDist2=-27.9

T1BYtopDist1=42.2
T1BYtopDist2=-5.7
T1BYbutDist=0

T1CXtopDist1=42.7
T1CXtopDist2=-28.4
T1CXbutDist1=-13
T1CXbutDist2=-16.1

T1CYtopDist1=98.2
T1CYtopDist2=-25.8
T1CYbutDist=0

#Transformation2
T2AtopAngle1=-14.8
T2AtopAngle2=-17.1
T2AbutAngle1=7.5
T2AbutAngle2=-6.3

T2BtopAngle1=8.9
T2BtopAngle2=-2.3
T2BbutAngle1=14.5
T2BbutAngle2=-4.5
T2BpkAngle1=0.8
T2BpkAngle2=-6.6

```

```
T2CtopAngle1=51
T2CtopAngle2=-3.6
T2CbutAngle1=21.8
T2CbutAngle2=-23
T2CpkAngle1=18.6
T2CpkAngle2=-24.2

#Transformation3
T3AtopDist1=0.711
T3AtopDist2=0.696

T3BtopDist1=1.02
T3BtopDist2=0.909
T3BpkDist1=1.116
T3BpkDist2=0.986

T3CtopDist1=1.012
T3CtopDist2=0.907
T3CpkDist1=0.668
T3CpkDist2=0.425

def Transformation1():
    T1XPoints=[0 for i in xrange(n)]
    T1YPoints=[0 for i in xrange(n)]

    #Randomly pick one type to transform
    t=random.randint(1,3)

    #Reciprocal
    if t==1:
        p=random.randint(1,n-2)
        T1XPoints[n-1]=T1AXtopDist
        T1XPoints[0]=T1AXbutDist
        T1XPoints[p]=T1AXpkDist
        T1YPoints[n-1]=T1AYtopDist
        T1YPoints[0]=T1AYbutDist
        T1YPoints[p]=T1AYpkDist
        d1X=(T1XPoints[n-1]-T1XPoints[p])/(n-1-p)
        d2X=(T1XPoints[p]-T1XPoints[0])/p
        d1Y=(T1YPoints[n-1]-T1YPoints[p])/(n-1-p)
        d2Y=(T1YPoints[p]-T1YPoints[0])/p

        for j in range (p,n):
            T1XPoints[j]=T1XPoints[p]+d1X*(j-p)
        for j in range (0,p):
            T1XPoints[j]=T1XPoints[0]+d2X*j
        for j in range (p,n):
            T1YPoints[j]=T1YPoints[p]+d1Y*(j-p)
        for j in range (0,p):
            T1YPoints[j]=T1YPoints[0]+d2Y*j

    #Misalignment
    if t==2:
        p=random.randint(1,n-2)
        T1XPoints[n-1]=random.uniform(T1BXtopDist2,T1BXtopDist1)
        T1XPoints[0]=random.uniform(T1BXbutDist2,T1BXbutDist1)
        T1XPoints[p]=random.uniform(T1BXpkDist2,T1BXpkDist1)
        T1YPoints[n-1]=random.uniform(T1BYtopDist2,T1BYtopDist1)
        T1YPoints[0]=T1BYbutDist
        d1X=(T1XPoints[n-1]-T1XPoints[p])/(n-1-p)
        d2X=(T1XPoints[p]-T1XPoints[0])/p
```

```
        dY=(T1YPoints[n-1]-T1YPoints[0])/(n-1)

        for j in range (p,n):
            T1XPoints[j]=T1XPoints[p]+d1X*(j-p)
        for j in range (0,p):
            T1XPoints[j]=T1XPoints[0]+d2X*j
        for j in range (0,n):
            T1YPoints[j]=T1YPoints[0]+dY*j

    #Alignment
    if t==3:
        T1XPoints[n-1]=random.uniform(T1CXtopDist2,T1CXtopDist1)
        T1XPoints[0]=random.uniform(T1CXbutDist2,T1CXbutDist1)
        T1YPoints[n-1]=random.uniform(T1CYtopDist2,T1CYtopDist1)
        T1YPoints[0]=T1CYbutDist
        dX=(T1XPoints[n-1]-T1XPoints[0])/(n-1)
        dY=(T1YPoints[n-1]-T1YPoints[0])/(n-1)

        for j in range (0,n):
            T1XPoints[j]=T1XPoints[0]+dX*j
        for j in range (0,n):
            T1YPoints[j]=T1YPoints[0]+dY*j

    #Record new points
    for m in range(0,n):
        Xvec=[T1XPoints[m],0,0]
        Yvec=[0,T1YPoints[m],0]
        newPoint=rs.PointAdd(memory[m], Xvec)
        newPoint=rs.PointAdd(newPoint, Yvec)
        arrPoints[k][q][m]=newPoint


def Transformation2():
    T2newAngles=[0 for i in xrange(n)]
    T2vecs=[0 for i in xrange(n)]
    T2newPoints=[0 for i in xrange(n)]

    #Create vectors between old points
    for j in range(1,n-1):
        T2vecs[j]=rs.VectorCreate(memory[j+1],memory[j])

    #Randomly pick one type to transform
    t=random.randint(1,3)
    #Linear
    if t==1:
        T2newAngles[n-2]=random.uniform(T2AtopAngle2,T2AtopAngle1)
        T2newAngles[1]=random.uniform(T2AbutAngle2,T2AbutAngle1)
        d=(T2newAngles[n-2]-T2newAngles[1])/(n-2)
        for j in range(1,n-1):
            T2newAngles[j]=T2newAngles[1]+d*j
    else:
        p=random.randint(2,n-2)
        #Dynamic Small
        if t==2:
            T2newAngles[n-2]=random.uniform(T2BtopAngle2,T2BtopAngle1)
            T2newAngles[1]=random.uniform(T2BbutAngle2,T2BbutAngle1)
            T2newAngles[p]=random.uniform(T2BpkAngle2,T2BpkAngle1)
        #Dynamic Big
        if t==3:
            T2newAngles[n-2]=random.uniform(T2CtopAngle2,T2CtopAngle1)
            T2newAngles[1]=random.uniform(T2CbutAngle2,T2CbutAngle1)
```

```
            T2newAngles[p]=random.uniform(T2CpkAngle2,T2CpkAngle1)
        d1=(T2newAngles[n-2]-T2newAngles[p])/(n-p)
        d2=(T2newAngles[p]-T2newAngles[1])/(p-1)
        for j in range (p,n-2):
            T2newAngles[j]=T2newAngles[p]+d1*(j-p)
        for j in range (1,p):
            T2newAngles[j]=T2newAngles[1]+d2*(j-1)

    #Generate point on arcs
    for j in range (1,n-1):
        if T2newAngles[j]>=0:
            newVec = rs.VectorRotate(T2vecs[j],random.uniform(0,T2newAngles[j]), [0,0,1])
        else:
            newVec = rs.VectorRotate(T2vecs[j], random.uniform(T2newAngles[j],0), [0,0,1])
        T2newPoints[j+1] =rs.PointAdd(memory[j], newVec)

    #Record points
    for m in range (0,2):
        arrPoints[k][q][m]=memory[m]
    for m in range (2,n):
        arrPoints[k][q][m]=T2newPoints[m]

def Transformation3():
    T3oldDist=[0 for i in xrange(n)]
    T3newDist=[0 for i in xrange(n)]
    T3vec=[0 for i in xrange(n)]
    T3vecAngles=[0 for i in xrange(n-1)]

    h=memory[n-1][1]
    cen=[0,h/2,0]
    T3base=rs.Distance(cen,memory[0])

    for j in range (0,n):
        T3oldDist[j]=rs.Distance(cen,memory[j])
        T3vec[j]=rs.VectorCreate(memory[j],cen)
        T3vec[j]=rs.VectorUnitize(T3vec[j])
    for j in range (0,n-1):
        T3vecAngles[j]=rs.VectorAngle(T3vec[j],T3vec[j+1])
    #Calculate deviation
    t=Deviation(T3vecAngles)

    #Assign type based on deviation
    #Taper
    if t==1:
        T3newDist[n-1]=random.uniform(T3AtopDist2,T3AtopDist1)*T3base
        T3newDist[0]=T3base
        d=(T3newDist[n-1]-T3newDist[0])/(n-1)
        for j in range (0,n):
            T3newDist[j]=T3newDist[0]+d*j
    else:
        e1=round(n/3,0)
        e2=round((2*n)/3,0)
        #Expansion
        if t==2:
            p=random.randint(e1,e2)
            T3newDist[n-1]=random.uniform(T3BtopDist2,T3BtopDist1)*T3base
            T3newDist[p]=random.uniform(T3BpkDist2,T3BpkDist1)*T3base
            T3newDist[0]=T3base
        #Contraction
        if t==3:
            p=random.randint(e1,e2)
```

```
230             T3newDist[n-1]=random.uniform(T3CtopDist2,T3CtopDist1)*T3base
231             T3newDist[p]=random.uniform(T3CpkDist2,T3CpkDist1)*T3base
232             T3newDist[0]=T3base
233         d1=(T3newDist[n-1]-T3newDist[p])/(n-1-p)
234         d2=(T3newDist[p]-T3newDist[0])/p
235         for j in range (p,n):
236             T3newDist[j]=T3newDist[p]+d1*(j-p)
237         for j in range (0,p):
238             T3newDist[j]=T3newDist[0]+d2*j
239
240     #Record new points
241     for j in range (0,n):
242         T3newDist[j]=(T3newDist[j]+T3oldDist[j])/2
243         T3vec[j]=rs.VectorScale(T3vec[j],T3newDist[j])
244         arrPoints[k][q][j]=rs.PointAdd(cen,T3vec[j])
245
246 def CurveGeneration():
247     #This function mainly organizes curves to a grid
248     CopyList=[]
249     newCurve=rs.AddCurve(arrPoints[k][q],2)
250     for j in range (0,n):
251         CopyList.append(rs.AddPoint(arrPoints[k][q][j]))
252     CopyList.append(newCurve)
253     CopyList.append(rs.AddPoint([0,0,0]))
254     rs.CopyObjects(CopyList,[k*300,q*800+2000,0])
255     return newCurve
```