**CSE423-A2**

Name: F.M. Ashfaq  
ID: 20101327  
Section: 08

!pip install tensorflow==1.15.0

!pip install -q lucid>=0.2.3

!pip install -q moviepy

import numpy as np

import json

import moviepy.editor as mvp

from google.colab import files

import tensorflow.compat.v1.gfile as gfile

import lucid.misc.io.showing as show

from lucid.misc.gl.glcontext import create\_opengl\_context

# Now it's safe to import OpenGL and EGL functions

import OpenGL.GL as gl

from OpenGL.GLU import \*

# create\_opengl\_context() creates GL context that is attached to an

# offscreen surface of specified size. Note that rendering to buffers

# of different size and format is still possible with OpenGL Framebuffers.

#

# Users are expected to directly use EGL calls in case more advanced

# context management is required.

WIDTH, HEIGHT = 400,400

create\_opengl\_context((WIDTH, HEIGHT))

# OpenGL context is available here.

print(gl.glGetString(gl.GL\_VERSION))

print(gl.glGetString(gl.GL\_VENDOR))

#print(gl.glGetString(gl.GL\_EXTENSIONS))

# Let's render something!

# find zone

# convert to zone 0

# apply mid point algo

# convert back to original zone

def findZone(x1,y1,x2,y2):

dx=x2-x1

dy=y2-y1

if abs(dx)>abs(dy):

if dx>=0 and dy>=0:

return 0

if dx<0 and dy>=0:

return 3

if dx<0 and dy<0:

return 4

if dx>=0 and dy<0:

return 7

else:

if dx>=0 and dy>=0:

return 1

if dx<0 and dy>=0:

return 2

if dx<0 and dy<0:

return 5

if dx>=0 and dy<0:

return 6

# print(findZone(0,20,10,20))

def convertToZone0(z,x1,y1,x2,y2):

if z==0:

return [x1,y1,x2,y2]

if z==1:

return [y1,x1,y2,x2]

if z==2:

return [y1,-x1,y2,-x2]

if z==3:

return [-x1,y1,-x2,y2]

if z==4:

return [-x1,-y1,-x2,-y2]

if z==5:

return [-y1,-x1,-y2,-x2]

if z==6:

return [-y1,-x1,-y2,-x2]

if z==7:

return [x1,-y1,x2,-y2]

def convertToOriginal(z,x,y):

if z==0:

return [x,y]

if z==1:

return [y,x]

if z==2:

return [-y,x]

if z==3:

return [-x,y]

if z==4:

return [-x,-y]

if z==5:

return [-y,-x]

if z==6:

return [y,-x]

if z==7:

return [x,-y]

gl.glClear(gl.GL\_COLOR\_BUFFER\_BIT)

gl.glColor3f(0,1,0)

gl.glPointSize(5)

def DrawLine(x1,y1,x2,y2):

z=findZone(x1,y1,x2,y2)

# print(z)

converted\_coordinates=convertToZone0(z,x1,x2,y1,y2)

# print(converted\_coordinates)

x1=converted\_coordinates[0]

y1=converted\_coordinates[1]

x2=converted\_coordinates[2]

y2=converted\_coordinates[3]

dx=x2-x1

dy=y2-y1

D=2\*dy-dx

dNE=2\*(dy-dx)

dE=2\*dy

x=x1

y=y1

while(x<=x2):

#Draw(x,y)

# print(x,y) # this is zone 0 coordinates

pixel=convertToOriginal(z,x,y)

p1=pixel[0]

q1=pixel[1]

gl.glBegin(gl.GL\_POINTS)

gl.glVertex2f(p1/100,q1/100)

gl.glEnd()

x+= 1

if (D>0):

y+= 1 # 0.001

D+=dNE

else:

D+=dE

#id 20101327

# for 2

DrawLine(-30,-10,50,50) #x1,x2,y1,y2

DrawLine(-30,-10,30,30)

DrawLine(-30,-10,10,10)

DrawLine(-10,-10,30,50)

DrawLine(-30,-30,10,30)

#for 7

DrawLine(20,20,10,50)

DrawLine(-2,20,50,50)

