!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 = 1000,1000

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))

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 = 1000,1000

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))

#**TASK1**

import random

randomlistx = random.sample(range(-100, 100), 50)

randomlisty = random.sample(range(-100, 100), 50)

randomlistx\_new = []

randomlisty\_new = []

for i in randomlistx:

randomlistx\_new.append(i/100)

for i in randomlisty:

randomlisty\_new.append(i/100)

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

count1 = 0

count2 = 0

while count1 < 50:

while count2 < 50:

gl.glColor3f(0,.5,0)

gl.glBegin(gl.GL\_POINTS)

gl.glVertex2f(randomlistx\_new[count1],randomlisty\_new[count2])

count1 += 1

count2 += 1

gl.glEnd()

#gl.glColor3f(145,0,0)

#gl.glBegin(gl.GL\_QUADS)

#gl.glVertex2f(.4,0)

#gl.glVertex2f(-.4,0)

#gl.glVertex2f(0,.4)

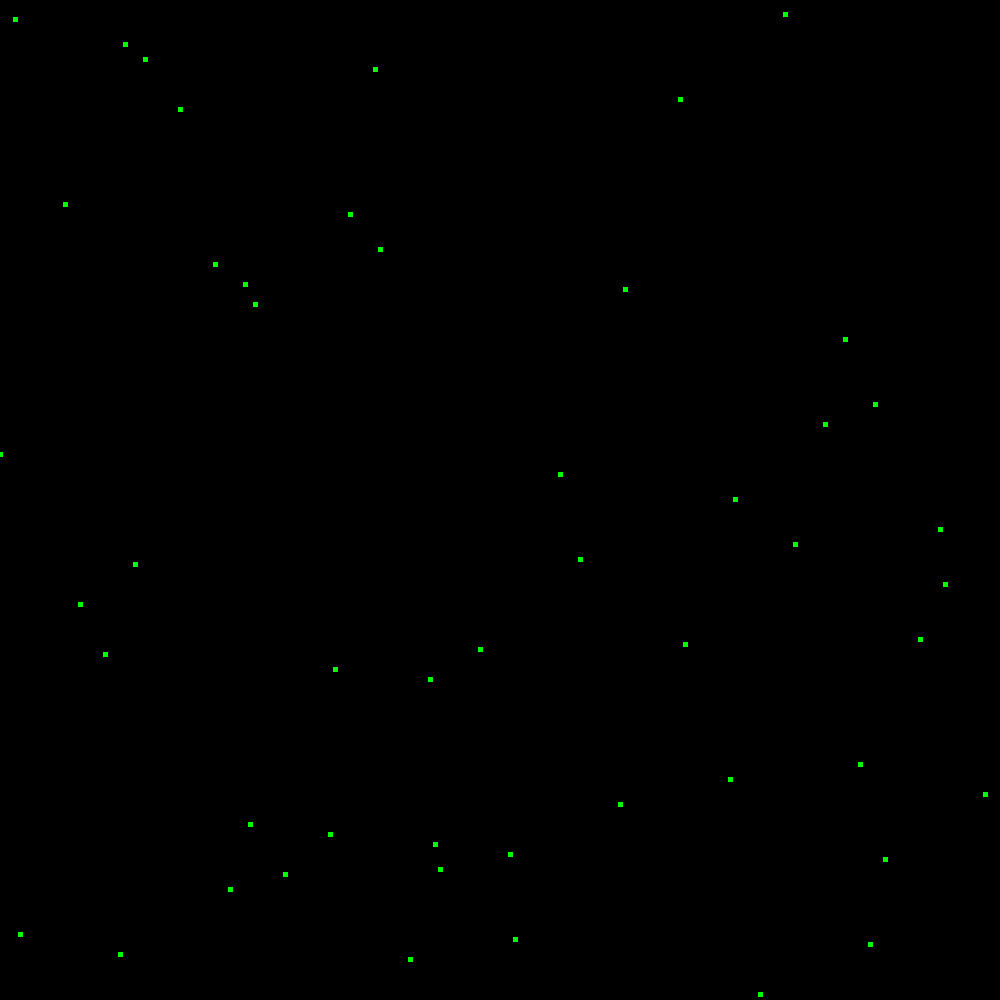
#gl.glVertex2f(0,-.4)

#gl.glEnd()

img\_buf = gl.glReadPixelsub(0, 0, WIDTH, HEIGHT, gl.GL\_RGB, gl.GL\_UNSIGNED\_BYTE)

img = np.frombuffer(img\_buf, np.uint8).reshape(HEIGHT, WIDTH, 3)[::-1]

show.image(img/255.0)



#**TASK2**

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

gl.glColor3f(0,.5,0)

gl.glBegin(gl.GL\_POINTS)

#doorknob

gl.glVertex2f(.05,-.45)

gl.glEnd()

gl.glColor3f(145,0,0)

gl.glBegin(gl.GL\_LINES)

#Roof

gl.glVertex2f(.4,.2)

gl.glVertex2f(-.4,.2)

gl.glVertex2f(.4,.2)

gl.glVertex2f(0,.8)

gl.glVertex2f(-.4,.2)

gl.glVertex2f(0,.8)

#

#body

gl.glVertex2f(.4,-.6)

gl.glVertex2f(.4,.2)

gl.glVertex2f(-.4,-.6)

gl.glVertex2f(-.4,.2)

gl.glVertex2f(.4,-.6)

gl.glVertex2f(-.4,-.6)

#

#door

gl.glVertex2f(-.1,-.6)

gl.glVertex2f(-.1,-.3)

gl.glVertex2f(.1,-.6)

gl.glVertex2f(.1,-.3)

gl.glVertex2f(-.1,-.3)

gl.glVertex2f(.1,-.3)

#

#window1

gl.glVertex2f(-.2,0)

gl.glVertex2f(-.2,-.15)

gl.glVertex2f(-.2,0)

gl.glVertex2f(-.35,0)

gl.glVertex2f(-.35,0)

gl.glVertex2f(-.35,-.15)

gl.glVertex2f(-.2,-.15)

gl.glVertex2f(-.35,-.15)

#

#winmdow2

gl.glVertex2f(.2,0)

gl.glVertex2f(.2,-.15)

gl.glVertex2f(.2,0)

gl.glVertex2f(.35,0)

gl.glVertex2f(.35,0)

gl.glVertex2f(.35,-.15)

gl.glVertex2f(.2,-.15)

gl.glVertex2f(.35,-.15)

gl.glEnd()

img\_buf = gl.glReadPixelsub(0, 0, WIDTH, HEIGHT, gl.GL\_RGB, gl.GL\_UNSIGNED\_BYTE)

img = np.frombuffer(img\_buf, np.uint8).reshape(HEIGHT, WIDTH, 3)[::-1]

show.image(img/255.0)

