Task 1

```
!pip install tensorflow==1.15.0
!pip install -q lucid>=0.2.3
!pip install -q moviepy
```

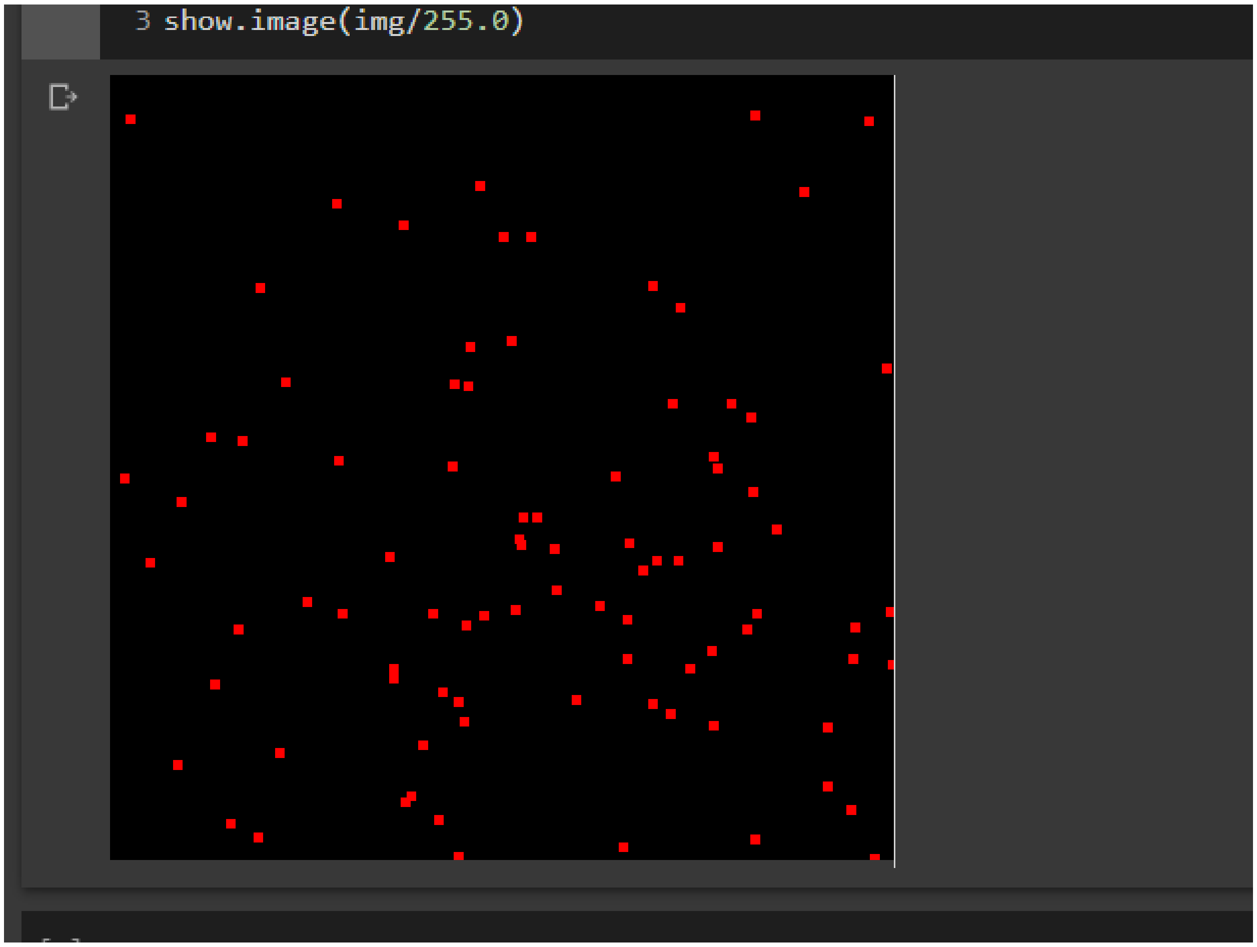
```
import numpy as np
import json
import moviepy.editor as mvp
from google.colab import files
import tensorflow.compat.v1.gfile as gfile
import lucid.misc.io.showing as show
```

```
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))
```

```
import random
gl.glClear(gl.GL_COLOR_BUFFER_BIT)
gl.glColor3f( 255, 0, 0)
gl.glPointSize(5)
gl.glBegin(gl.GL_POINTS)
for i in range(50):
    if i<13:
        gl.glVertex2f(-random.random(),random.random())
    if i<25:
        gl.glVertex2f(-random.random(),-random.random())
    if i<37:
        gl.glVertex2f(random.random(),-random.random())
    else:
        gl.glVertex2f(random.random(),random.random())
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)
```



```
show.image(img/255.0)
```

Task 2

```
!pip install tensorflow==1.15.0
!pip install -q lucid>=0.2.3
!pip install -q moviepy
```

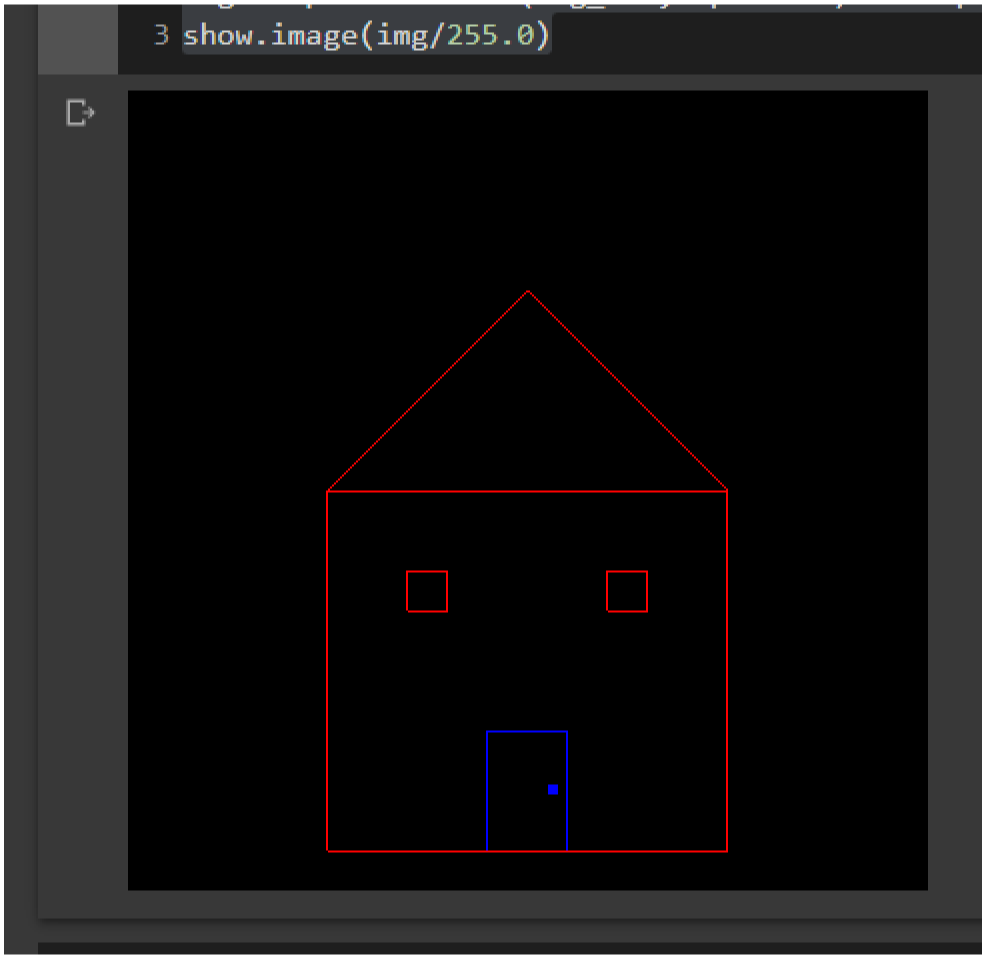
```
import numpy as np
import json
import moviepy.editor as mvp
from google.colab import files
import tensorflow.compat.v1.gfile as gfile
import lucid.misc.io.showing as show
```

```
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.
```

```
import random
gl.glClear(gl.GL_COLOR_BUFFER_BIT)
gl.glColor3f( 255, 0, 0)
gl.glPointSize(5)
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(.5, 0)
gl.glVertex2f(0, .5)
gl.glEnd()
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(0, .5)
gl.glVertex2f(-.5, 0)
gl.glEnd()
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(.5, 0)
gl.glVertex2f(-.5, 0)
gl.glEnd()
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(.5, 0)
gl.glVertex2f(.5, -.9)
gl.glEnd()
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(-.5, 0)
gl.glVertex2f(-.5, -.9)
gl.glEnd()
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(.5, -.9)
gl.glVertex2f(-.5, -.9)
gl.glEnd()
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(-.2, -.2)
gl.glVertex2f(-.3, -.2)
gl.glEnd()
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(-.2, -.3)
gl.glVertex2f(-.3, -.3)
gl.glEnd()
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(-.3, -.2)
gl.glVertex2f(-.3, -.3)
gl.glEnd()
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(-.2, -.2)
gl.glVertex2f(-.2, -.3)
gl.glEnd()
#----------------------------------
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(.2, -.2)
gl.glVertex2f(.3, -.2)
gl.glEnd()
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(.2, -.3)
gl.glVertex2f(.3, -.3)
gl.glEnd()
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(.3, -.2)
gl.glVertex2f(.3, -.3)
gl.glEnd()
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(.2, -.2)
gl.glVertex2f(.2, -.3)
gl.glEnd()
#---------------------------
gl.glColor3f( 0, 0, 255 )
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(.1, -.6)
gl.glVertex2f(.1, -.9)
gl.glEnd()
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(-.1, -.6)
gl.glVertex2f(-.1, -.9)
gl.glEnd()
gl.glBegin(gl.GL_LINES)
gl.glVertex2f(-.1, -.6)
gl.glVertex2f(.1, -.6)
gl.glEnd()
gl.glBegin(gl.GL_POINTS)
gl.glVertex2f(.06, -.75)
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)
```



```
show.image(img/255.0)
```