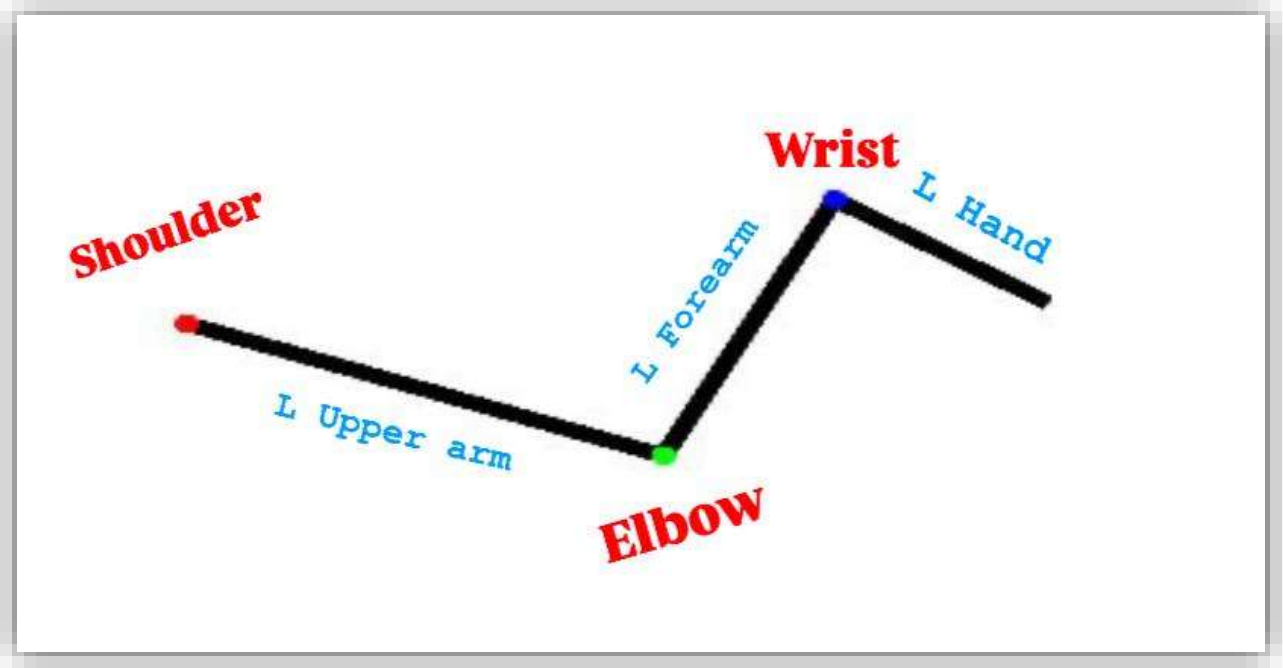
Computer Graphics

Lab 7

Name	Ranime Ahmed Elsayed Shehata
ID	21010531



Drawing and Controlling a 2D Robotic Arm

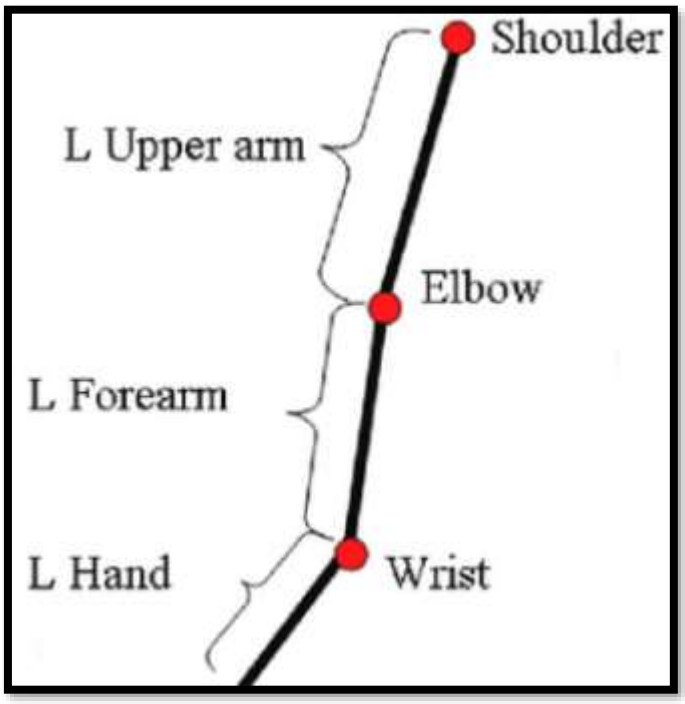
Problem Statement:

In this assignment, you will utilize the materials provided in Lab 7 to create and control a 2D robotic arm. This robotic arm will consist of three main components: the shoulder, elbow, and wrist. Your goal is to replicate the movements demonstrated in the attached video. Follow the detailed instructions below to successfully complete this assignment.

Understanding the Components

Before you begin, it's important to understand the three components of the robotic arm:

1. **Shoulder**: This is the base of the arm that allows for the initial movement. It will enable the arm to move up and down as well as rotate.
2. **Elbow**: This joint connects the shoulder to the wrist and allows for bending and straightening of the arm.
3. **Wrist**: The wrist joint provides the final movement, allowing the end of the arm to rotate and position itself accurately.



Key Concepts to Consider

- **Degrees of Freedom**: Each joint (shoulder, elbow, wrist) has its own range of motion. Understanding how each joint moves will help you control the arm effectively.
- **Coordinate System**: Familiarize yourself with the coordinate system used in the lab materials. This will help you position the arm accurately in the 2D space.
- **Simulation**: The video provided demonstrates the desired movements. Pay close attention to how the arm moves in relation to the target points.

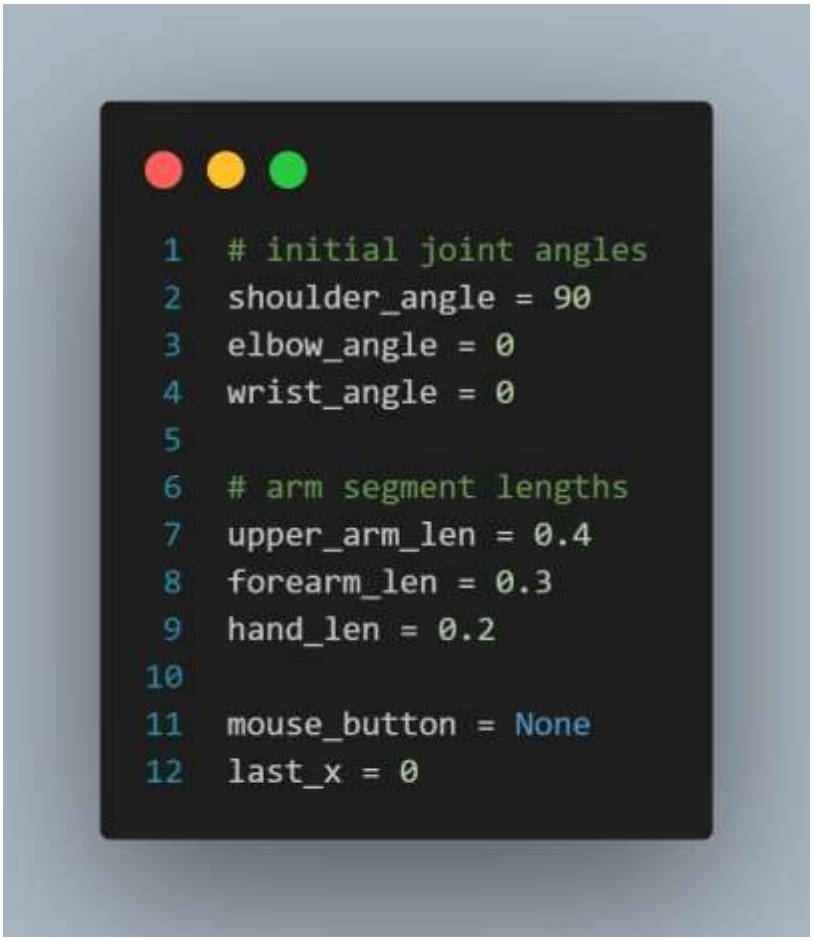
Code Snippet:

Code Explanation:

- **Imports:**

```
from OpenGL.GL import *
from OpenGL.GLU import *
from OpenGL.GLUT import *
import sys
import math
```

- **Global Variables:**



```
# initial joint angles
shoulder_angle = 90
elbow_angle = 0
wrist_angle = 0

# arm segment lengths
upper_arm_len = 0.4
forearm_len = 0.3
hand_len = 0.2

mouse_button = None
last_x = 0
```

- **Drawing Joints:**
  ➔ Draws a filled circle using polygons to represent joints (shoulder, elbow, wrist).

```
def draw_circle(radius=0.01, segments=20):
    glBegin(GL_POLYGON)
    for i in range(segments):
        theta = 2.0 * math.pi * i / segments
        x = radius * math.cos(theta)
        y = radius * math.sin(theta)
        glVertex2f(x, y)
    glEnd()
```

- **Drawing the arm segment:**
  ➔ Draws a thin vertical rectangle representing one segment (bone) of the arm.

```
def draw_segment(length):
    glBegin(GL_POLYGON)
    glVertex2f(-0.008, 0)
    glVertex2f(0.008, 0)
    glVertex2f(0.008, -length)
    glVertex2f(-0.008, -length)
    glEnd()
```

- **Drawing 2D robotic arm using hierarchical transformations:**
  ➔ It draws:

1. **Upper Arm** (shoulder to elbow)
2. **Forearm** (elbow to wrist)
3. **Hand**

```python
def draw_arm():
    glClear(GL_COLOR_BUFFER_BIT)
    glLoadIdentity()

    # start positon of the arm
    glTranslatef(-0.2, 0.1, 0.0)

    # SHOULDER
    glPushMatrix()
    glRotatef(shoulder_angle, 0, 0, 1)
    glColor3f(0, 0, 0)
    draw_segment(upper_arm_len)
    glColor3f(1, 0, 0)
    draw_circle()

    # Move to elbow position
    glTranslatef(0, -upper_arm_len, 0)

    # ELBOW
    glRotatef(elbow_angle, 0, 0, 1)
    glColor3f(0, 0, 0)
    draw_segment(forearm_len)
    glColor3f(0, 1, 0)
    draw_circle()

    # Move to wrist position
    glTranslatef(0, -forearm_len, 0)

    # WRIST
    glRotatef(wrist_angle, 0, 0, 1)
    glColor3f(0, 0, 0)
    draw_segment(hand_len)
    glColor3f(0, 0, 1)
    draw_circle()

    glPopMatrix()
    glutSwapBuffers()
```

- **Mouse Button Callback:**
- **Key Press Callback:**
- **Mouse Movement Callback:**

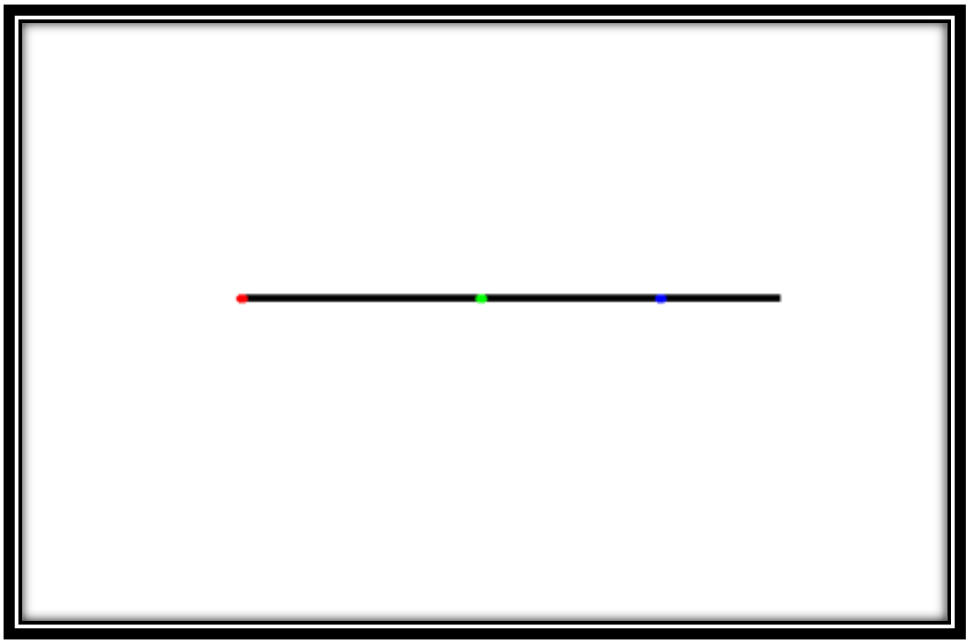
```
def mouse(button, state, x, y):
    global mouse_button, last_x
    if state == GLUT_DOWN:
        mouse_button = button
        last_x = x
    else:
        mouse_button = None

def keyboard(key, x, y):
    global shoulder_angle, elbow_angle, wrist_angle

    if key == b'r':
        shoulder_angle = 90
        elbow_angle = 0
        wrist_angle = 0
        glutPostRedisplay()

def motion(x, y):
    global shoulder_angle, elbow_angle, wrist_angle, last_x

    dx = x - last_x
    last_x = x

    speed = 0.2  # control sensitivity

    if mouse_button == GLUT_LEFT_BUTTON:
        shoulder_angle += dx * speed
    elif mouse_button == GLUT_RIGHT_BUTTON:
        elbow_angle += dx * speed
    elif mouse_button == GLUT_MIDDLE_BUTTON:
        wrist_angle += dx * speed
    elif keyboard == b'r':
        shoulder_angle = 90
        elbow_angle = 0
        wrist_angle = 0

    glutPostRedisplay()
```

- OpenGL Initialization & Entry Point:

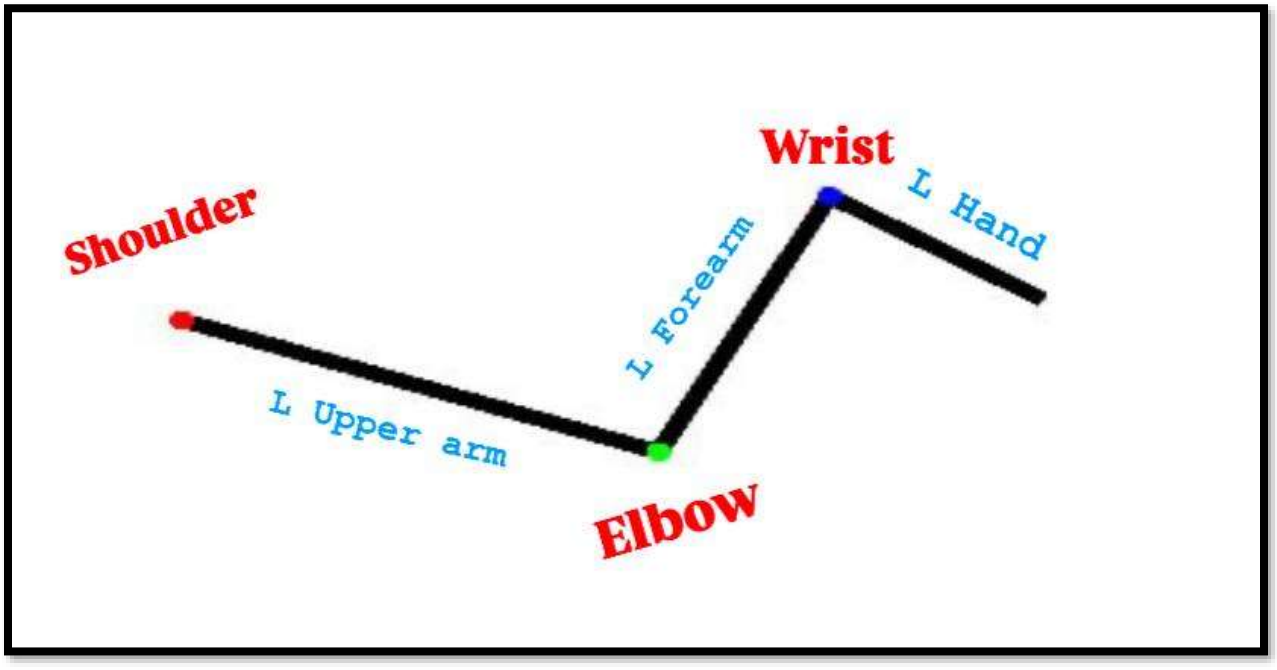
```
def init():
    glClearColor(1.0, 1.0, 1.0, 1.0)
    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    gluOrtho2D(-1.0, 1.0, -1.0, 1.0)
    glMatrixMode(GL_MODELVIEW)

def main():
    glutInit(sys.argv)
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB)
    glutInitWindowSize(1000, 750)
    glutCreateWindow(b"2D Robotic Arm")
    init()
    glutDisplayFunc(draw_arm)
    glutMouseFunc(mouse)
    glutMotionFunc(motion)
    glutKeyboardFunc(keyboard)
    glutMainLoop()

if __name__ == "__main__":
    main()
```

Screen Output:

Initial Position:



After movement:



How to Control Motion:

Input	Action
Left mouse drag	Rotate **shoulder**
Right mouse drag	Rotate **elbow**
Middle mouse drag	Rotate **wrist**
Press `r`	**Reset** all joint angles
