**Documentary Research Report on Makeblock Robot**

**Introduction**

Makeblock is a versatile robotics platform that allows users to build and program robots for various tasks. The Makeblock robots are designed for educational purposes, providing hands-on experience in coding, engineering, and robotics. This report explores several core tasks that can be performed using Makeblock robots, including moving in a square pattern, basic movement, detecting colours with QUAD RGB sensors, obstacle detection with ultrasonic sensors, and implementing line-following capabilities.

**Task 1: Basic Movement**

**Objective:** To program the mBlock to execute a continuous circular movement, allowing students to grasp the principle of motor control and trajectory management.

**Overview:** Moving in a circular pattern requires careful coordination of the robot’s motors to maintain a smooth and consistent curvature. This task provides students with practical experience in adjusting motor speeds and understanding the mechanics of circular motion.

**Methodology:**

**Setup:**

1.Ensure the mBlock is assembled and connected to the computer via USB or Bluetooth.

2.Open the mBlock software and create a new project

**Sample Code(mBlock)**

import event, time, cyberpi, mbot2 import time

@event.is\_press('b')

def is\_btn\_press():

mbot2.forward(100, 4)

time.sleep(2)

mbot2.straight(-50)

time.sleep(2)

mbot2.turn(-90)

mbot2.forward(50, 2)

time.sleep(2)

mbot2.turn(90)

mbot2.forward(50, 2)

**Task 2: Move in a Rectangle**

**Objective:** To program the Makeblock robot to move in a square pattern, demonstrating its movement capabilities and programming skills.

Overview: This task involves instructing the robot to move to four corners of a square, helping students understand basic movement commands and coordinates system.

**Methodology:**

**1.Setup:** Ensure the Makeblock robot is assembled and connected to a computer or tablet using the Makeblock app or mBlock software.

**2.Programming:** Use the visual programming interface or Python to create a routine that commands the robot to move in a square.

**Sample Code(mBlock)**

import event, time, cyberpi, mbot2, mbuild

import time

@event.is\_press('b')

def is\_btn\_press():

mbot2.straight(50)

mbot2.turn(90)

mbot2.straight(25)

mbot2.turn(90)

mbot2.straight(50)

mbot2.turn(90)

mbot2.straight(25)

mbot2.turn(90)

time.sleep(4)

**Expected Outcome:** The robot successfully navigates a square pattern, demonstrating accurate movement and control.

**Task 3: QUAD RGB Sensors to detect colours**

**Objective:** To integrate QUAD RGB sensors with the Makeblock robot for colour detection tasks.

**Overview: QUAD** RGB sensors allow the robot to detect and respond to different colours in its environment. This task teaches students about sensor integration and colour recognition.

**Methodology:**

1. Sensor Setup: Attach QUAD RGB sensors to the robot
2. Programming: Use the mBlock software to program the robot to change behaviour based on detected colours.

**Sample Code(mBlock)**

from makeblock import \*

import cyberpi from cyberpi import \*

import mbot2 import time import random

testInProgress = False displayStartMessage = False

def BeginRGBSensorTest(): cyberpi.display.clear()

# Global variables for display purposes  
global testInProgress  
global displayStartMessage   
testInProgress = True  
  
testSensor = "l2"  
  
rValue = 0  
gValue = 0  
bValue = 0  
  
# Sensor Test  
  
while (True):  
 cyberpi.display.clear()  
 #cyberpi.console.print("Object detected " + str(cyberpi.ultrasonic2.get(1)) + "cm from sensor\nPress \"A\" to end test.")  
 rValue = cyberpi.quad\_rgb\_sensor.get\_red(testSensor, 1)  
 gValue = cyberpi.quad\_rgb\_sensor.get\_green(testSensor, 1)  
 bValue = cyberpi.quad\_rgb\_sensor.get\_blue(testSensor, 1)  
  
 cyberpi.console.print("Red Value: " + str(rValue) + "\nGreen Value: " + str(gValue) + "\nBlue Value: " + str(bValue) + "\nPress \"A\" to end test.")  
  
 if cyberpi.controller.is\_press("a"):  
 break  
 time.sleep(1)  
  
testInProgress = False  
displayStartMessage = False  
cyberpi.display.clear()

# **Main loop**

while True:

if not testInProgress:  
   
 if not displayStartMessage:  
 cyberpi.display.clear()  
 cyberpi.console.print("Please press button \"A\" to begin Sensor test.")  
 displayStartMessage = True  
  
 if cyberpi.controller.is\_press("a"):  
 BeginRGBSensorTest()  
  
time.sleep(0.1)

**Expected Outcome:** The robot moves or turns based on the colour detected by the QUAD RGB sensors, demonstrating effective sensor integration.

**Task 2: Ultrasonic Sensors for Obstacle Detection**

**Objective:** To use ultrasonic sensors for obstacle detection, allowing the Makeblock robot to navigate its environment safely.

**Overview:** Ultrasonic sensors measure the distance to nearby objects, enabling the robot to avoid collisions. This task enhances understanding of distance measurement and safe navigation in robotics.

**Methodology:**

**1.Sensor Setup:** Attach ultrasonic sensors to the robot.

**2.Programming:** Use mBlock to write a routine that commands the robot to stop or change direction when an obstacle is detected.

**Sample Code(mBlock)**

import event, time, cyberpi, mbuild, mbot2

@event.is\_press('b')

def is\_btn\_press():

while True:

if mbuild.ultrasonic2.get (1) < 15:

mbot2.turn(-97)

mbot2.forward(50)

**Expected Outcome:** The robot detects obstacles and avoids collisions by stopping or changing direction, demonstrating effective use of ultrasonic sensors.

**Task 5: Line Following**

**Objective:** To implement line-following capabilities in the Makeblock robot using infrared sensors.

**Overview:** Line following is a common robotics task where the robot is programmed to follow a path marked by a line. This task teaches students about feedback control and sensor response.

**Methodology:**

1. **Sensor Setup:** Attach infrared sensors to the robot.
2. **Programming:** Use mBlock to program the robot to follow the line based on sensor input.

**Sample Code(mBlock)**

import event, time, cyberpi, mbot2, mbuild

base\_power = 0

kp = 0

left\_power = 0

right\_power = 0

@event.is\_press('a')

def is\_btn\_press():

global base\_power, kp, left\_power, right\_power mbot2.drive\_power(0, 0)

@event.is\_press('b')

def is\_btn\_press1():

global base\_power, kp, left\_power, right\_power

while True:

base\_power = 30 kp = 0.8 while True:

left\_power = (base\_power - kp \* mbuild.quad\_rgb\_sensor.get\_offset\_track(1)) right\_power = -1 \* ((base\_power + kp \* mbuild.quad\_rgb\_sensor.get\_offset\_track(1))) mbot2.drive\_power (left\_power, right\_power)

**Conclusions:**

The provided code showcases how to control the Makeblock robot using Python in the mBlock environment. By following the structured sequence of movement, students enhance their understanding of robotics, coding, and mechanical navigation. This task serves a practical introduction in programming and robotics for learners at various levels.

**These questions are designed to assess understanding of the tasks associated with the Makeblock robot**

**1.What is the primary objective of task 1 involving movement in the Makeblock robot?**

* The primary objective of the task 1 is to program the mBlock to execute a continuous circular movement, allowing students to grasp the principles of motor control and trajectory management.

**2.How does the methodology for moving the Makeblock robot in a rectangle pattern help students understand movement commands?**

* The methodology requires students to instruct the robot to navigate to the four corners of a rectangle, which helps understand basic movement commands and the concept of coordinate systems through practical application.

**3.In Task 3, what is the purpose of integrating QUAD RGB sensors with the Makeblock robot?**

* The purpose of integrating QUAD RGB sensors is to enable the robot to detect and respond to different colours in its environment, teaching students about sensor integration and colour recognition.

**4.What does Task 4 involving ultrasonic sensors enable the makeblock robot to do, and why is this important?**

* Task 4 enables the robot to measure the distance to nearby objects and avoid collisions, enhancing students’ understanding of distance measurement and safe navigation in robotics, which is crucial for real-world applications.

**5.Descibe the significance of implementing line-following capabilities in Task 5 and what students learn from this task**

* Implementing line-following capabilities is significant as it teaches students about feedback control and sensor response, helping them understand how robots can autonomously follow paths marked by lines, which is a fundamental skill in robotics and automation.