FINAL PROJECT

STAR WARS BB-8

KINEMATICS AND DYNAMICS FINAL PROJECT

INSTUCTED BY AARJAN O

SECTION 2

GROUP MEMBER

PRIYANKA JOSHI
66011646
ROBOTICS AND AI ENGINEERING

INTRODUCTION

RAI

BB-8, THE BELOVED DROID FROM STAR WARS, CAPTURED HEARTS WITH EXPRESSIVE MOVEMENTS AND LOYAL COMPANIONSHIP. INSPIRED BY ITS DESIGN, THIS PROJECT AIMS TO BUILD A SIMPLIFIED VERSION, BRINGING ITS CHARM AND FUNCTIONALITY TO LIFE. THE FOCUS IS ON REPLICATING BB-8'S ROLLING MOTION, HEAD STABILIZATION, AND BASIC INTERACTIVE FEATURES, CREATING AN ENGAGING AND RESPONSIVE ROBOTIC COMPANION.

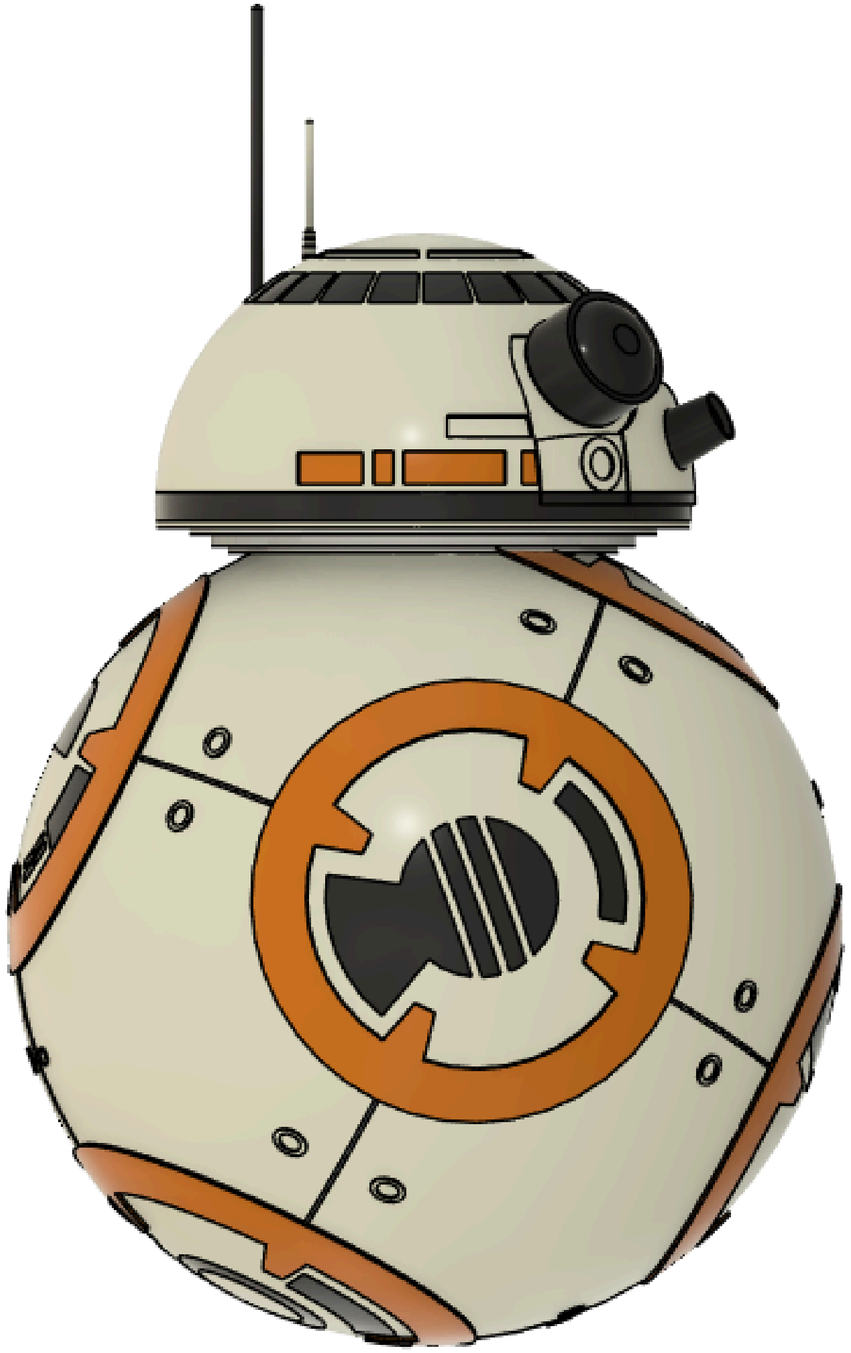
THE DESIGN

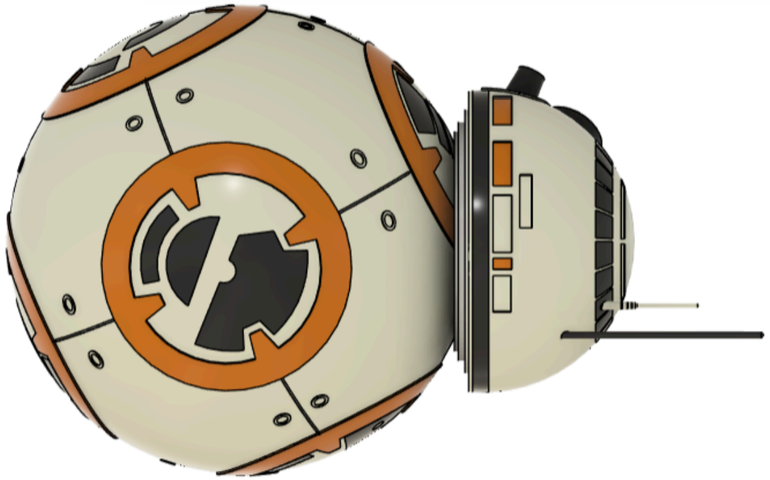
N THIS PROJECT, BB-8’S BODY IS 3D-PRINTED, ENSURING PRECISE SHAPE AND DETAIL REPLICATION. A GYROSCOPIC SYSTEM MAINTAINS BALANCE, ENABLING SMOOTH ROLLING MOTION. MAGNETS OR MECHANICAL LINKAGES STABILIZE THE HEAD, ALLOWING INDEPENDENT MOVEMENT AND EXPRESSION. THIS APPROACH SHOWCASES THE USE OF 3D PRINTING AND ROBOTICS IN RECREATING THE ICONIC DROID.

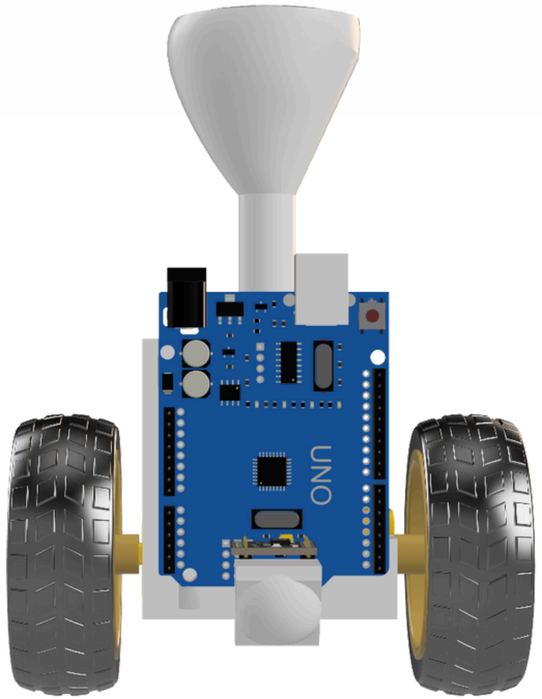
KMITL Robotics and AI

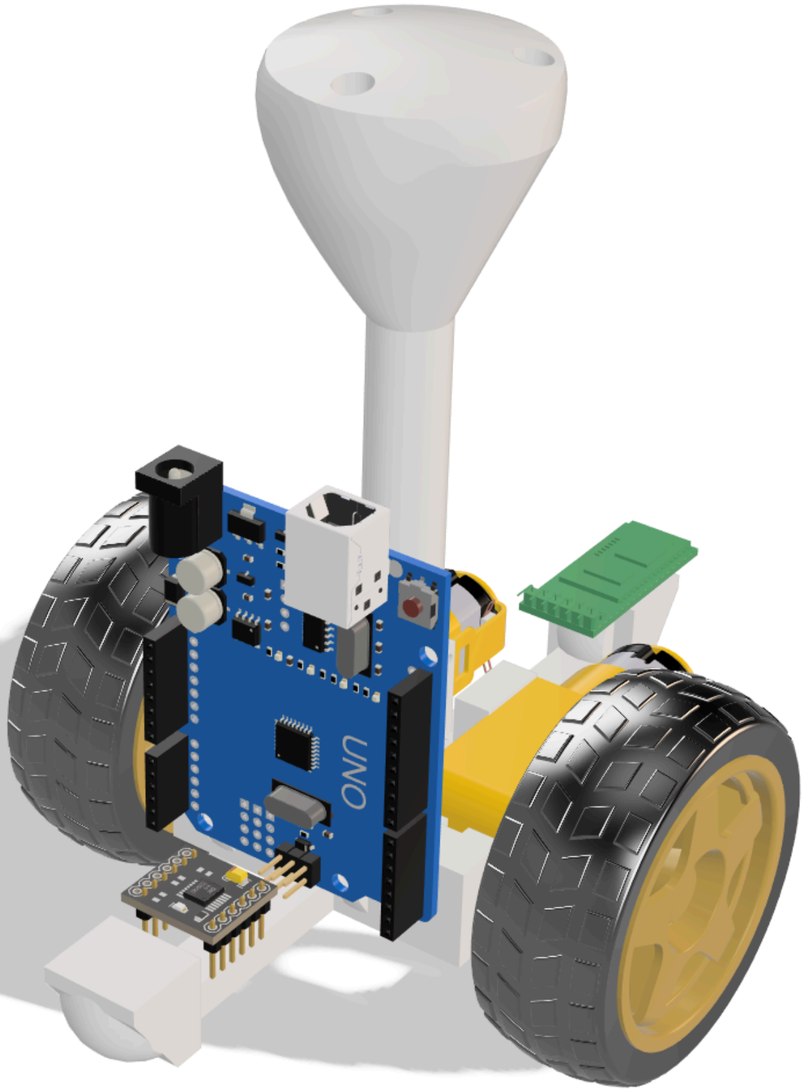
RAI

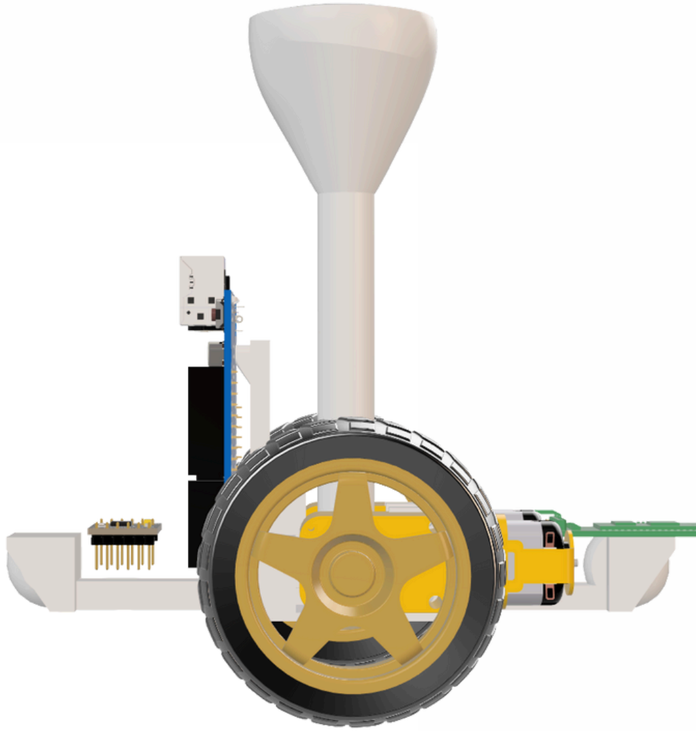
BB-8'S EXTERNAL DESIGN

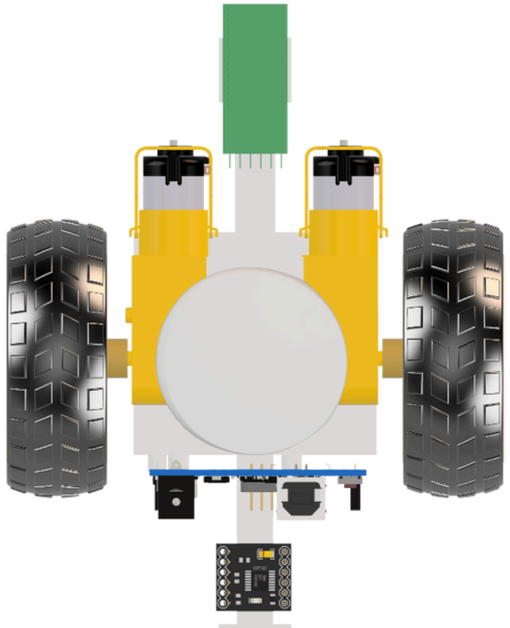






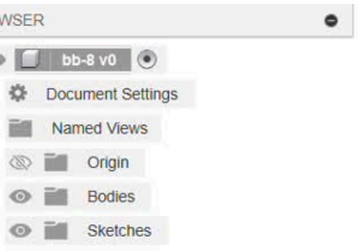


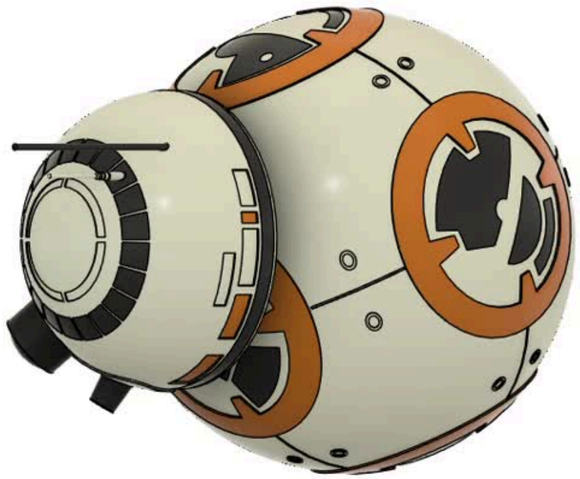




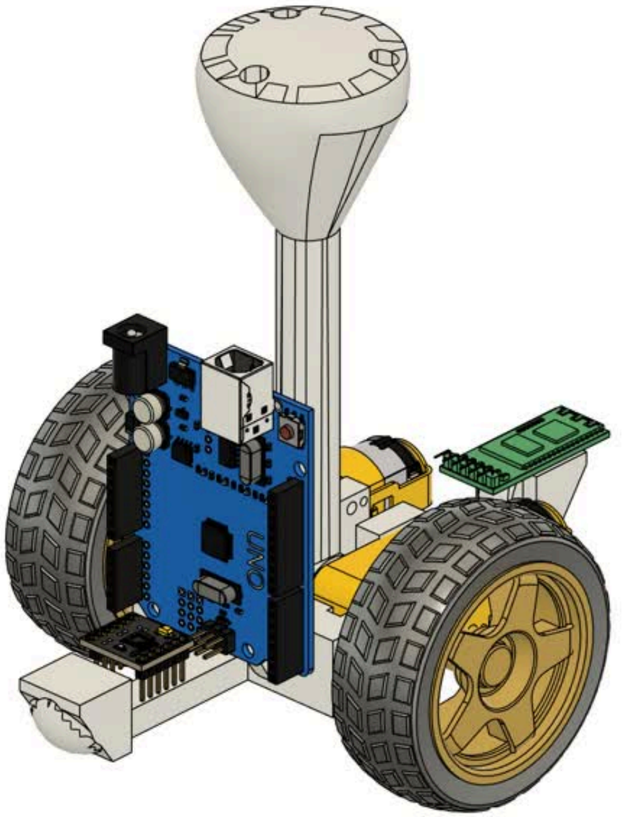
YOUTUBE LINK

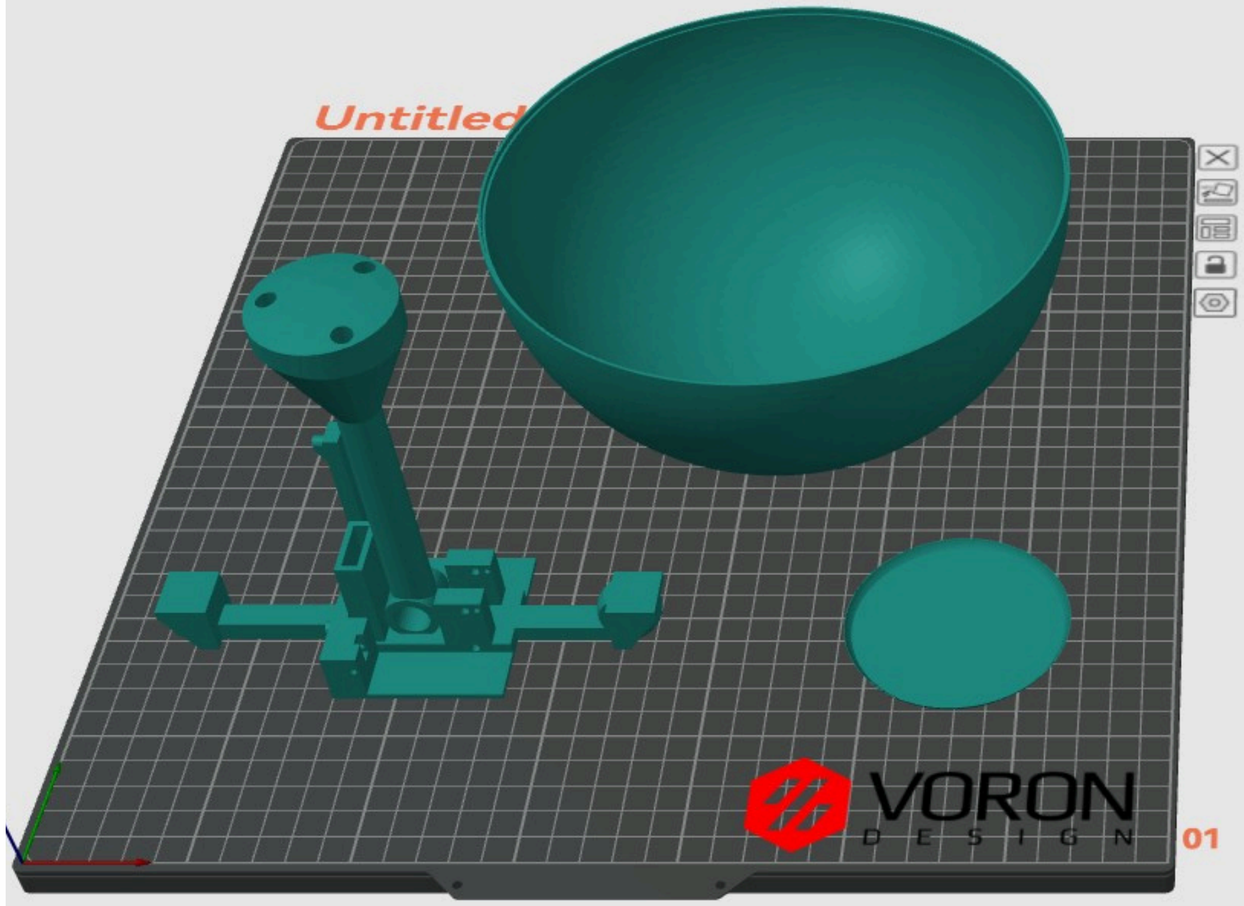
BB-8'S INTERNAL MECHANISM

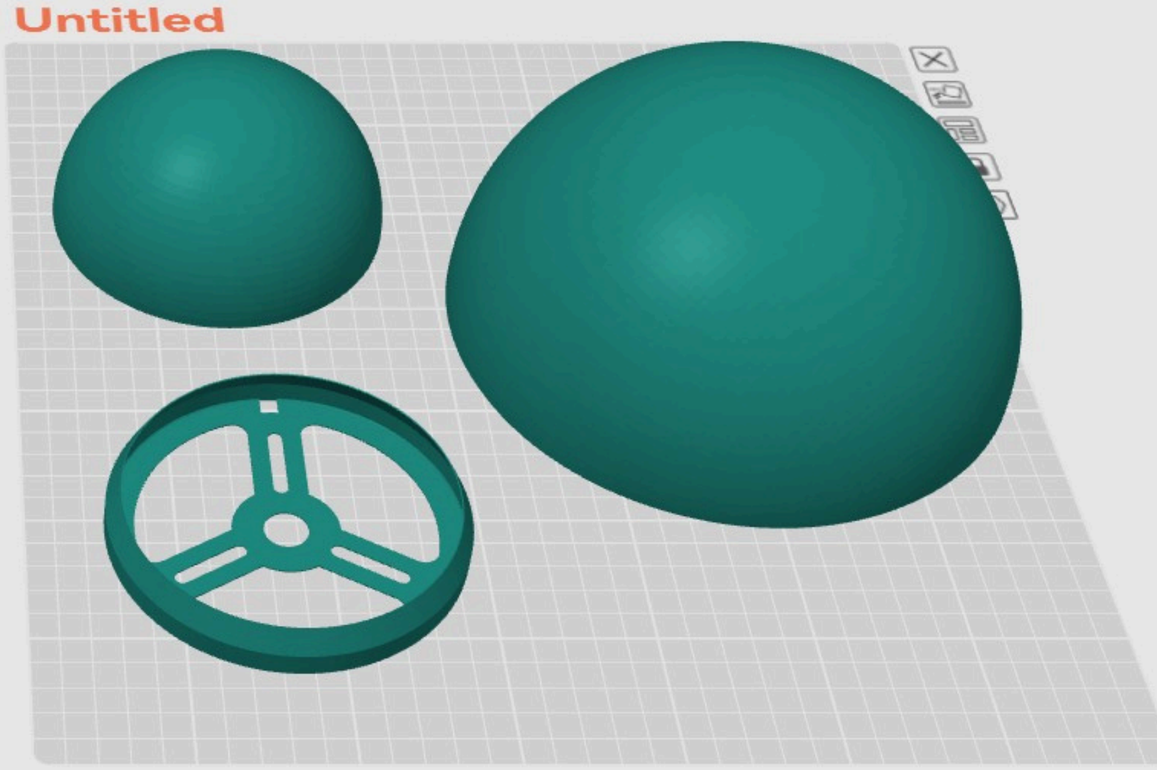

WSER
bb-8 v0
Document Settings
Named Views
Origin
Bodies
Sketches



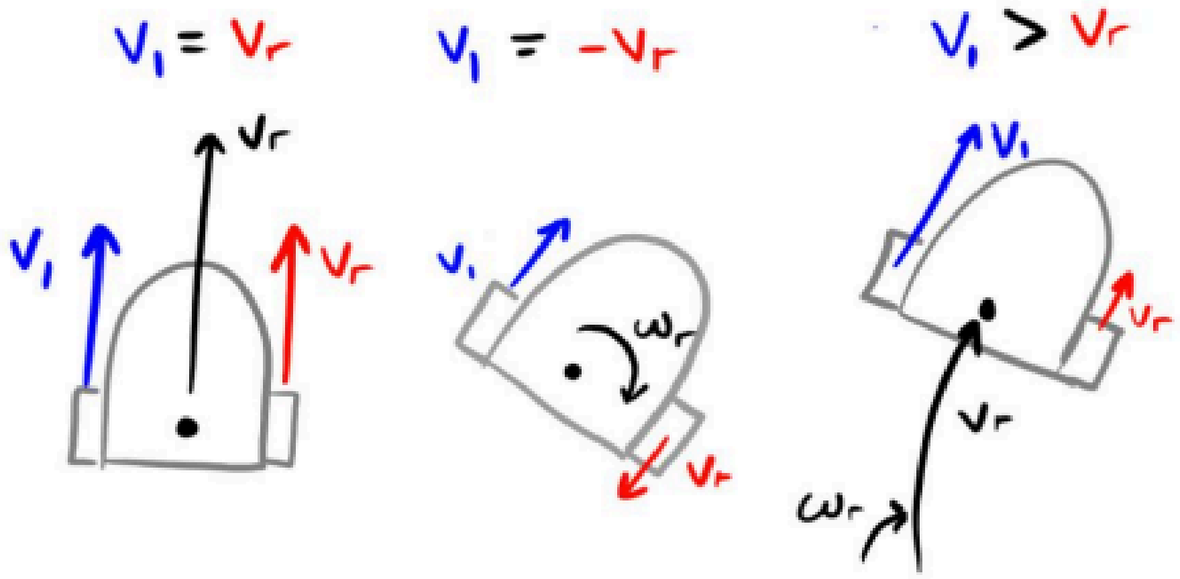

LEFT


UNO


Untitled
VORON
DESIGN
01


Untitled

THE MATHEMATICAL THEORY



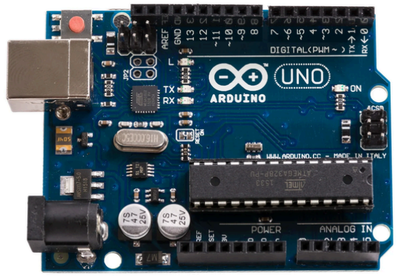
To compute linear and rotational velocities, the following equations are used:

$$v_{m/s} = \frac{v_r \ m/s \ + v_l \ m/s}{2}$$

and

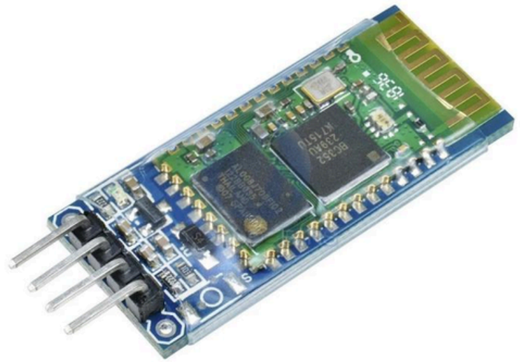
$$\omega_{rad/s} = \frac{v_r \ m/s \ - v_l \ m/s}{W}$$

THE HARDWARE



ARDUINO UNO BOARD

MOTOR DRIVER DRV8633



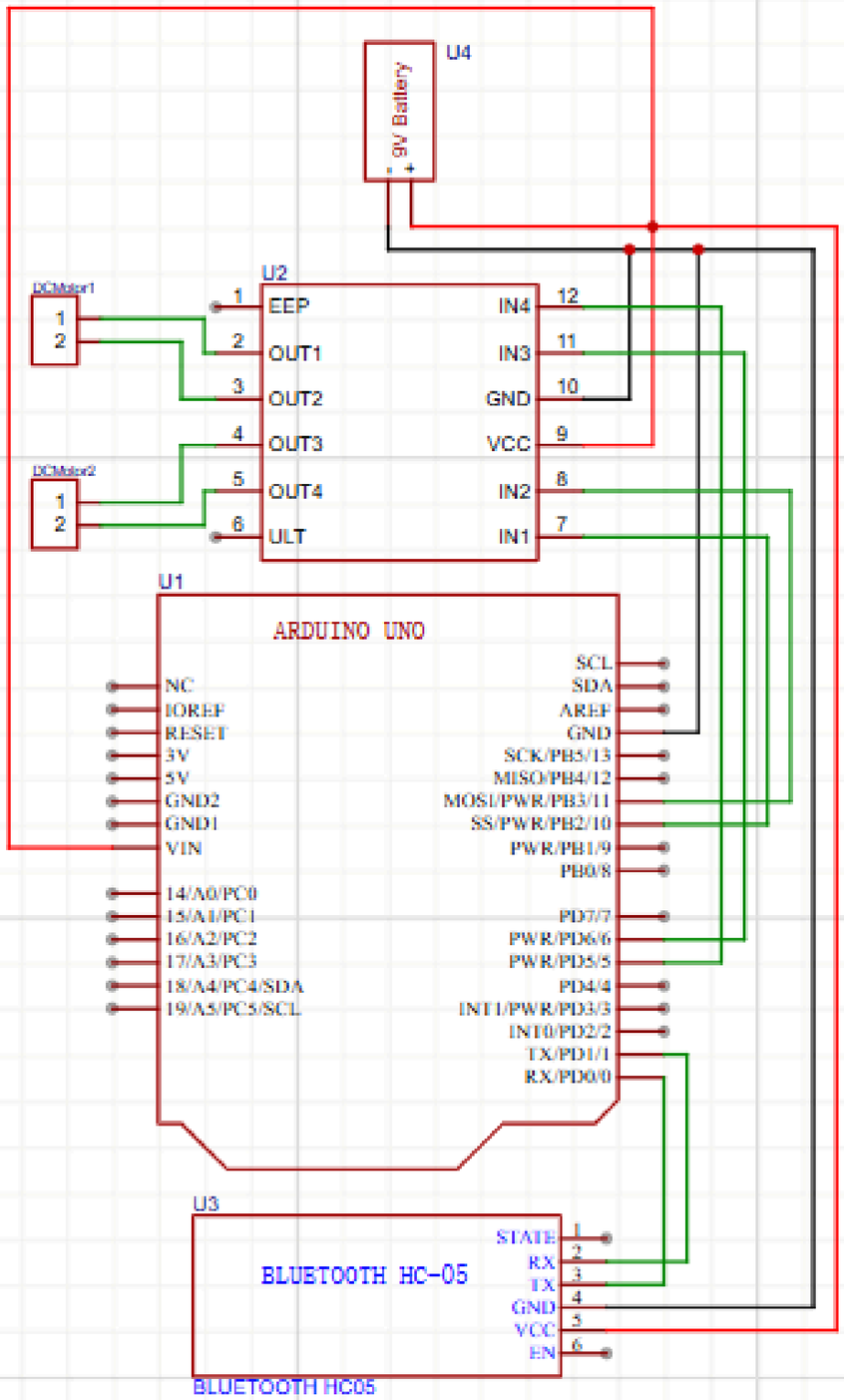
HC-06 BLUETOOTH MODEL

COMPONENETS

KMITL Robotics and AI

RAI

THE SCHEMATIC



Schematic	kinematics	Create at	2025-03-31

CODE AND SIMULATION

BLUETOOTH COMMUNICATION

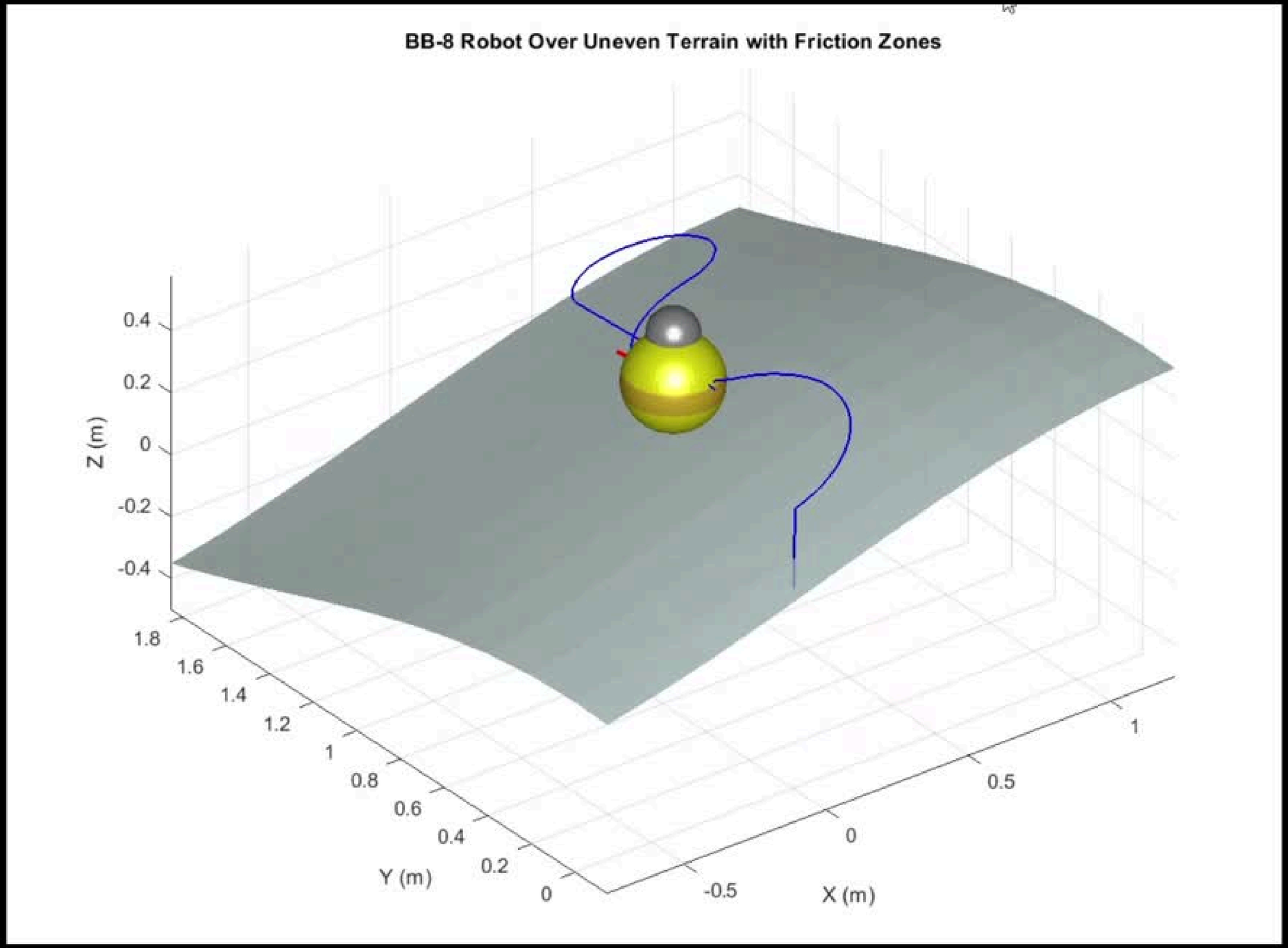
```
// Define the motor control pins
#define MOT_A1_PIN 10
#define MOT_A2_PIN 9
#define MOT_B1_PIN 6
#define MOT_B2_PIN 5

// Define Bluetooth module pins (HC-06)
#define BT_RX 2  // Connect to HC-06 TX
#define BT_TX 3  // Connect to HC-06 RX

#include <SoftwareSerial.h>
SoftwareSerial bluetooth(BT_RX, BT_TX); // RX, TX for HC-06
```

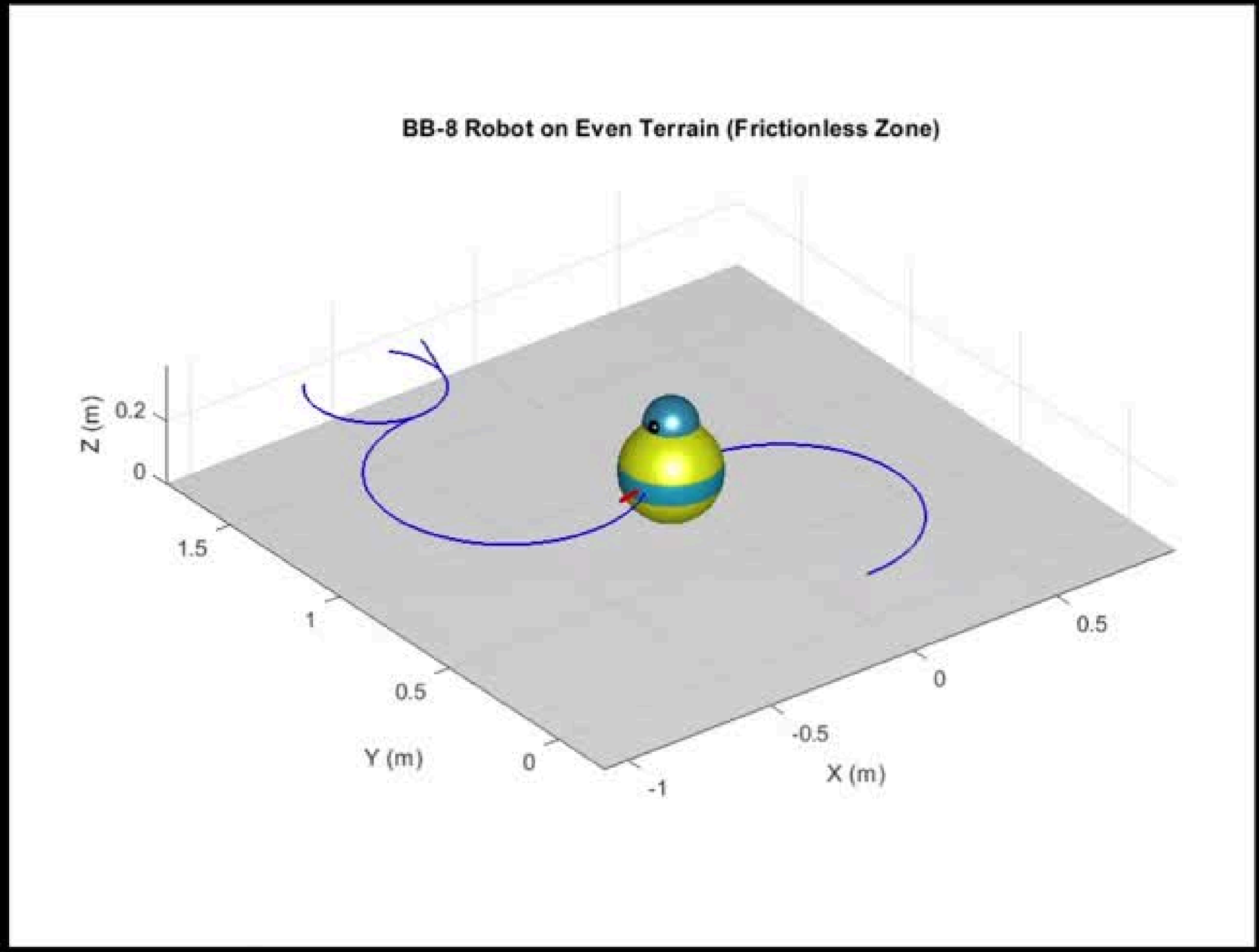
MOTOR CONTROLLING

```
void setMotor(int pwm_A, int pwm_B) {
  // Stop motors before changing direction
  analogWrite(MOT_A1_PIN, 0);
  analogWrite(MOT_A2_PIN, 0);
  analogWrite(MOT_B1_PIN, 0);
  analogWrite(MOT_B2_PIN, 0);

  delay(10); // Small delay to ensure stopping

  // Motor A
  if (pwm_A > 0) {
    analogWrite(MOT_A1_PIN, pwm_A);
    analogWrite(MOT_A2_PIN, 0);
  } else if (pwm_A < 0) {
    analogWrite(MOT_A1_PIN, 0);
    analogWrite(MOT_A2_PIN, -pwm_A);
  }

  // Motor B
  if (pwm_B > 0) {
    analogWrite(MOT_B1_PIN, pwm_B);
    analogWrite(MOT_B2_PIN, 0);
  } else if (pwm_B < 0) {
    analogWrite(MOT_B1_PIN, 0);
    analogWrite(MOT_B2_PIN, -pwm_B);
  }
}

void forward() {
  setMotor(speedVal, speedVal);
}

void backward() {
  setMotor(-speedVal, -speedVal);
}

// Modify turning functions to create a radius turn instead of abrupt stops
void turnLeft() {
  setMotor(turnSpeed, speedVal); // Left motor moves slower, right motor moves normally
}

void turnRight() {
  setMotor(speedVal, turnSpeed); // Right motor moves slower, left motor moves normally
}

void stopMotors() {
  setMotor(0, 0);
}
```

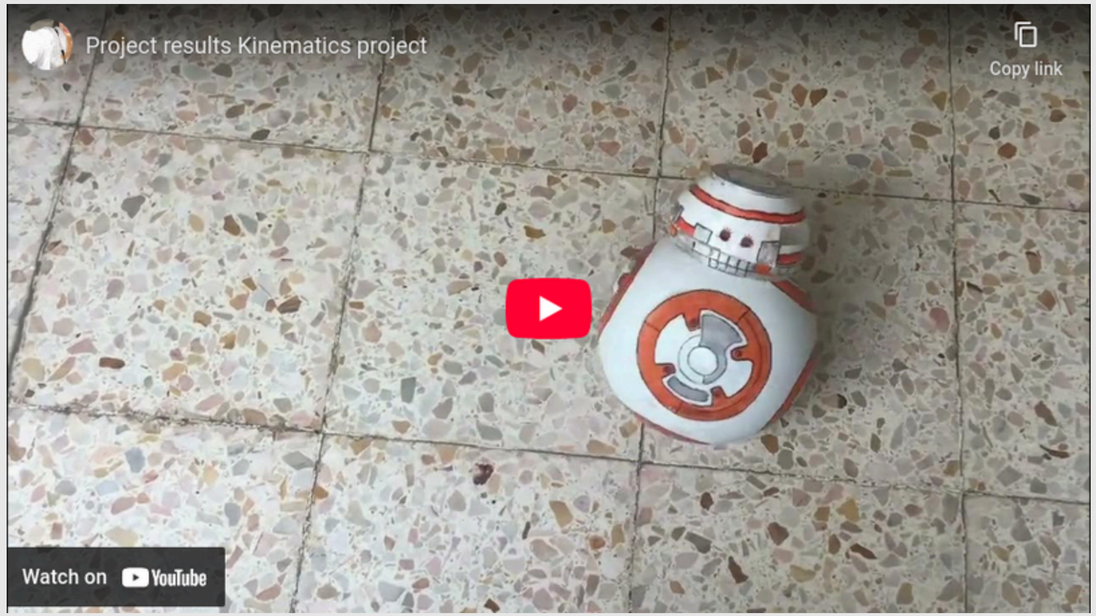
```
% Simulation time
dt = 0.05;
simTime = 30;
t = 0:dt:simTime;
N = length(t);

% Initial pose
x = zeros(1, N);
y = zeros(1, N);
theta = zeros(1, N);

% Wheel angular velocities
omega_R = zeros(1, N);
omega_L = zeros(1, N);

% Create motion pattern
for i = 1:N
    if i < N/4
        omega_R(i) = 5; omega_L(i) = 3;
    elseif i < N/2
        omega_R(i) = 3; omega_L(i) = 5;
    elseif i < 3*N/4
        omega_R(i) = 6; omega_L(i) = 2;
    else
        omega_R(i) = 4; omega_L(i) = 4;
    end
end
```

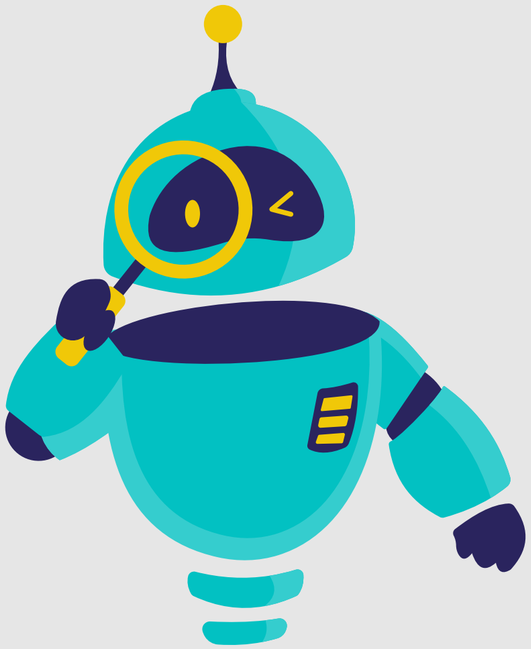
```
% Simulate motion
for i = 1:N-1
    v_R = r * omega_R(i);
    v_L = r * omega_L(i);

    v = (v_R + v_L) / 2;
    omega = (v_R - v_L) / L;

    x(i+1) = x(i) + v * cos(theta(i)) * dt;
    y(i+1) = y(i) + v * sin(theta(i)) * dt;
    theta(i+1) = theta(i) + omega * dt;
end
```


BB-8 Robot Over Uneven Terrain with Friction Zones
Z (m)
0.4
0.2
0
-0.2
-0.4
Y (m)
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
X (m)
-0.5
0
0.5
1


BB-8 Robot on Even Terrain (Frictionless Zone)
Z (m)
0.2
0
1.5
1
0.5
Y (m)
0
-1
-0.5
0
0.5
X (m)

RESULT





CLICK IN THE PICTURE

THANK YOU