SYMBIOSIS INSTITUTE OF TECHNOLOGY, PUNE
DEPARTMENT OF ROBOTICS AND AUTOMATION ENGINEERING

ACADEMIC YEAR: 202425 SEM: V

ASSIGNMENT NO: 4

NAME:Akshaj Chainani
PRN: 22070127008
BATCH: RA1

Aim:

To control an epuck robot via keyboard and obtain accelerometer and camera data. The data is printed to the console, with the camera data displayed in an RGB format.

Apparatus:

Hardware: epuck robot

Software: Webots (robot simulation software), OpenCV (for advanced image processing if needed)

Theory

Basic Definitions

Robot Control: Managing the robot's movements using commands (e.g., forward, backward, turning).

Keyboard Input: Commands are entered through the keyboard to control the robot.

Accelerometer: Measures the robot's acceleration along x, y, and z axes.

Camera: Captures visual data in RGB format for image analysis.

Concepts

Motion Control: Adjusting motor speeds to achieve desired robot movements based on keyboard inputs.

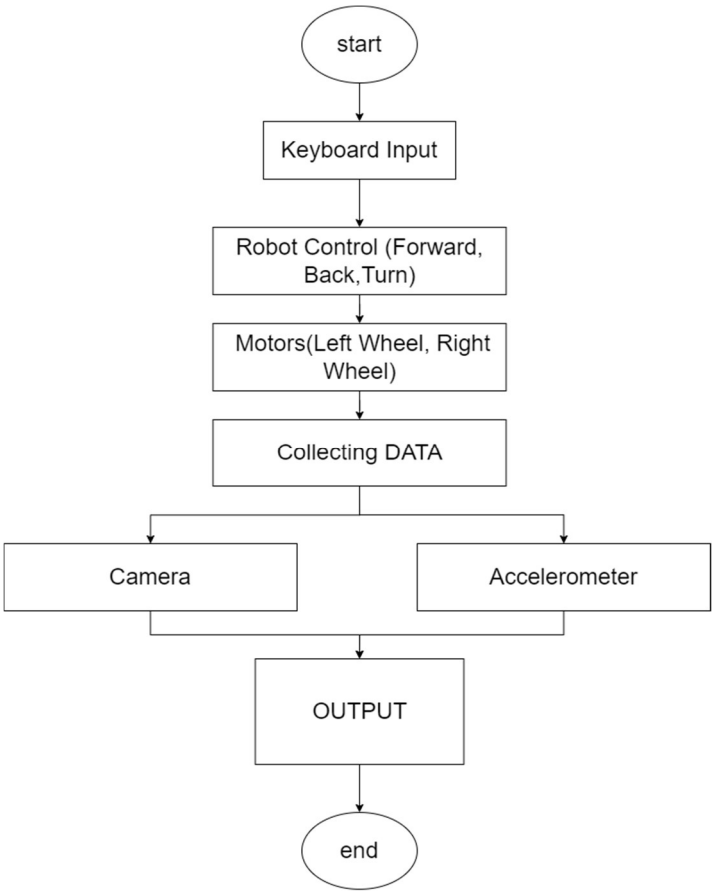
Image Processing: Analyzing and visualizing RGB image data captured by the camera.

Sensor Data Acquisition: Collecting and displaying data from the accelerometer and camera.

System Components

1. Keyboard Input: Sends movement commands to the robot.
2. Robot Control: Converts commands into motor actions.
3. Motors: Drive the robot's wheels based on control commands.
4. Accelerometer: Provides acceleration data, output to the console.
5. Camera: Captures RGB images, with data displayed in a textbased format.

Block Diagram:



Program Code

```c
include <webots/robot.h>
include <webots/motor.h>
include <webots/keyboard.h>
include <webots/accelerometer.h>
include <webots/camera.h>
include <stdio.h>

define TIME_STEP 64  // Simulation time step in milliseconds
define MAX_SPEED 6.28  // Maximum speed for epuck's motors

int main(int argc, char argv) {
  wb_robot_init();

  wb_keyboard_enable(TIME_STEP);
```

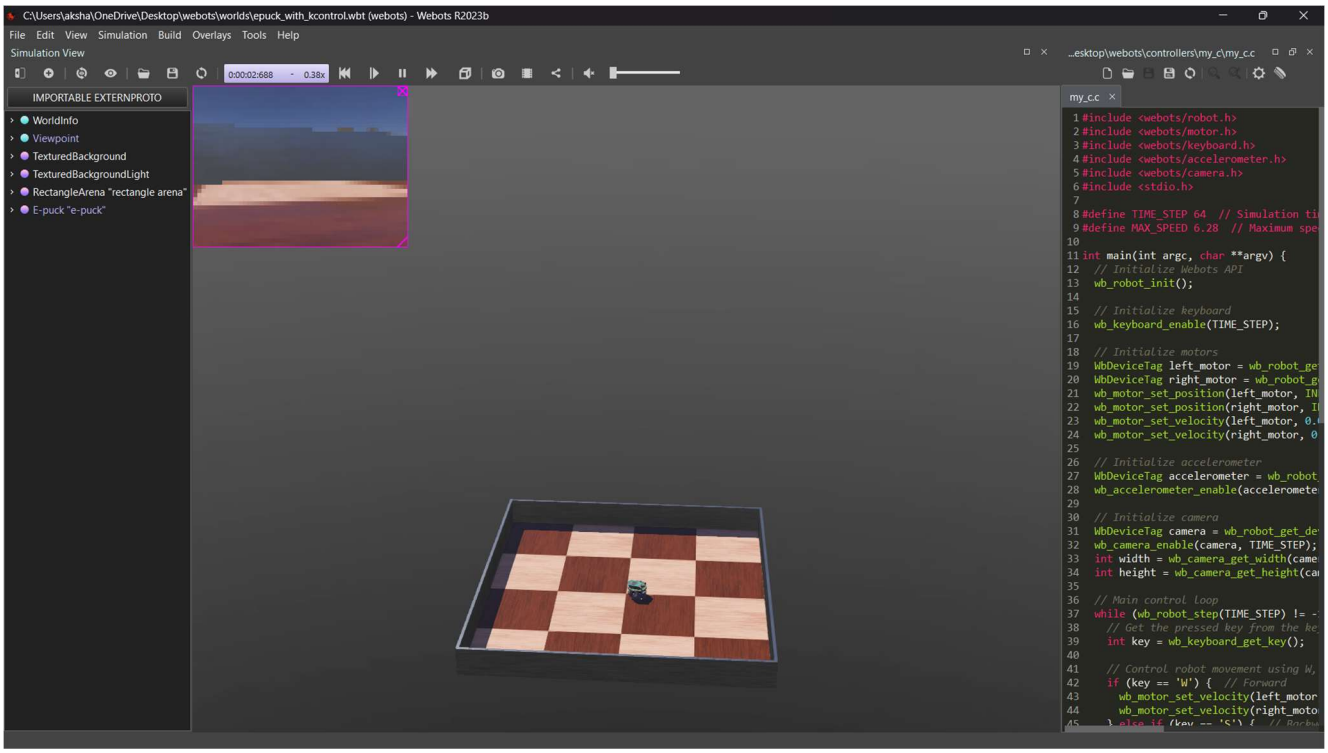
```
WbDeviceTag left_motor = wb_robot_get_device("left wheel motor");
WbDeviceTag right_motor = wb_robot_get_device("right wheel motor");
wb_motor_set_position(left_motor, INFINITY);
wb_motor_set_position(right_motor, INFINITY);
wb_motor_set_velocity(left_motor, 0.0);
wb_motor_set_velocity(right_motor, 0.0);

WbDeviceTag accelerometer = wb_robot_get_device("accelerometer");
wb_accelerometer_enable(accelerometer, TIME_STEP);

WbDeviceTag camera = wb_robot_get_device("camera");
wb_camera_enable(camera, TIME_STEP);

while (wb_robot_step(TIME_STEP) != 1) {
  int key = wb_keyboard_get_key();

  if (key == 'W') {  // F
    wb_motor_set_velocity(left_motor, MAX_SPEED);
    wb_motor_set_velocity(right_motor, MAX_SPEED);
  } else if (key == 'S') {  // B
    wb_motor_set_velocity(left_motor, MAX_SPEED);
    wb_motor_set_velocity(right_motor, MAX_SPEED);
  } else if (key == 'A') {  // L
    wb_motor_set_velocity(left_motor, MAX_SPEED);
    wb_motor_set_velocity(right_motor, MAX_SPEED);
  } else if (key == 'D') {  // R
    wb_motor_set_velocity(left_motor, MAX_SPEED);
    wb_motor_set_velocity(right_motor, MAX_SPEED);
  } else {
    wb_motor_set_velocity(left_motor, 0.0);
    wb_motor_set_velocity(right_motor, 0.0);
  }

  const double acc_values = wb_accelerometer_get_values(accelerometer);
  printf("Accelerometer data: x=%f, y=%f, z=%f\n", acc_values[0], acc_values[1],
       acc_values[2]);

  const unsigned char image = wb_camera_get_image(camera);
  if (image) {
    printf("Camera image captured.\n");
  }
 }

 wb_robot_cleanup();

 return 0;
}```
```

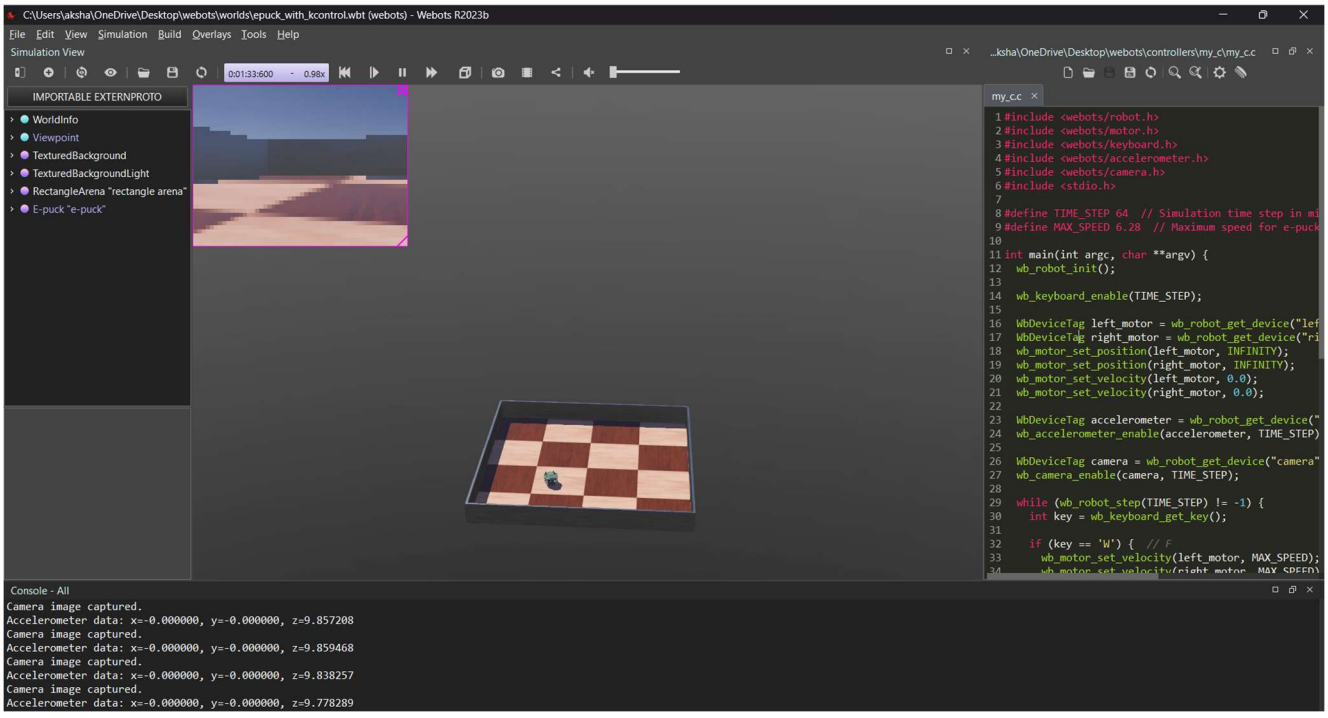
WBT file:

```
VRML_SIM R2023b utf8

EXTERNPROTO
    "https://raw.githubusercontent.com/cyberbotics/webots/R2023b/projects/objects/backg
    rounds/protos/TexturedBackground.proto"
EXTERNPROTO
    "https://raw.githubusercontent.com/cyberbotics/webots/R2023b/projects/objects/backg
    rounds/protos/TexturedBackgroundLight.proto"
EXTERNPROTO
    "https://raw.githubusercontent.com/cyberbotics/webots/R2023b/projects/objects/floors
    /protos/RectangleArena.proto"
EXTERNPROTO
    "https://raw.githubusercontent.com/cyberbotics/webots/R2023b/projects/robots/gctroni
    c/epuck/protos/Epuck.proto"

WorldInfo {
}
Viewpoint {
  orientation 0.19455261928687348 0.18139131084927665 0.9639743101748085
      1.6773454265779086
  position 0.28545433771922846 4.071496209182291 2.44512037232444
}
TexturedBackground {
}
TexturedBackgroundLight {
}
RectangleArena {
}
Epuck {
  hidden position_0_0 262.4992779207983
  hidden position_0_1 189.93403053593255
  hidden linearVelocity_0 1.6399050366757036e15 4.4940407558975555e15
      8.526512829121203e16
  hidden angularVelocity_0 1.8363892038771716e13 6.822679138188623e14
      2.4435111877874352e14
  hidden rotation_1 0 1 0 1.3945049807443257
  hidden linearVelocity_1 1.5707930374654517e15 6.475482717559927e15
      5.110721562009854e16
  hidden angularVelocity_1 2.7970268141456163e13 1.1802739347619813e13
      2.443836539563662e14
  hidden rotation_2 0 1 0 1.4384713205449555
  hidden linearVelocity_2 1.0855528104854933e15 7.292014879253209e16
      1.7193531606395667e15
  hidden angularVelocity_2 2.978521470146129e13 1.1719086728670828e13
```

```
        2.4778795190580043e14
    translation 0.020922074871331044 0.012784018963247207 6.396199578768622e05
    rotation 1.1088596607676432e09 1.8731646256556124e10 1 0.33538101926767894
    controller "my_c"
}
```

Snapshots





Learning Outcome:

Understanding how to control a robot using keyboard inputs in Webots.
Reading and interpreting accelerometer data.
Capturing and displaying camera data (in RGB format) on the console.
Integrating hardware sensors and control systems in a simulated environment.

This summary provides a structured overview of the project, including theory, code, and expected outcomes.