Operation/Assembly Manual

# Assembly and Disassembly Instructions

The following list of steps breaks down the Assembly Instructions for the NASA Robotic explorer Rover. Each section is broken up into separate sub-assemblies, with listed steps, to manufacture the Physical Rover Model.

Note: Rover is not meant to be Disassembled, if need to disassemble, follow sub assembly steps listed below in reverse

## Body

1. Use nuts and bolts attach the aluminum panels to the frame of the rover. Repeat for each side.
2. Attach the top using hinges and nuts and bolts.
3. Feed the wiring through the holes in the panels to the Arduino.

Screw Drive Sub assembly

1. Attach the motor to the to the motor mount using the 5-screw included with the motor mount.
2. Attach the motor mount to the channel using ¼ 20 screws.
3. Extend the shaft on the motor by using a coupler and shaft extension.
4. The screw is attached to the extended shaft and fastened down using a set screw.
5. The channel is then screwed onto a wooded plate that also attaches to the linear actuators hinge using the ¼ 20 screws. Repeat for all 4 screws.

Linear Actuator Sub assembly

1. The 8 in. linear actuators are attached to the wooden plate and screw drive sub assembly through a hinge using nuts and bolts.
2. It is also attached to the body of the rover with another hinge and fastened with nuts and bolts.
3. The 4 in. Linear actuator is attached to the body of the 8 in. Actuator using zip ties.
4. The 4 in. Actuators are placed in the holes in the body and attached to the bottom of the rover using zip ties.

Arduino Sub assembly

1. The circuit shown below details the connections that must be made to supply power to all the components. 10-12 gauge wire should be used for all these connections. These are only the power connections and Arduino logic connections are shown in the next three steps.

A picture containing computer

Description automatically generated

1. The circuit diagram for relay module logic connections is shown below. Arduino male-male, male-female, and female-female wires can be used for these connections.

A circuit board

Description automatically generated

1. The circuit diagram for the motor driver logic connections is shown below. Arduino male-male, male-female, and female-female wire connectors can be used for these connections.

A circuit board

Description automatically generated

1. The circuit diagram for the Bluetooth module logic connections is shown below. Arduino male-male, male-female, and female-female wires can be used for these connections.

A circuit board

Description automatically generated

# Operation

This section covers the steps to operate the rover. The rover operates on a Bluetooth Receiver that allows the rover to move forward and back, turn left and right, and lower and extend the arm mechanisms.

Steps to Operate-

1. Upload Arduino code: Copy the code from **Appendix A** to an Arduino sketch on the Arduino programming software. Upload this code to the Arduino board using the upload cable.
2. On a phone, download the ‘ArduinoBlue’ app from the App Store, Open the app, connect to the Bluetooth chip (HM10 4.0 BT Module), and create a series of buttons identical to the ones below. Each button should be assigned a numerical value as detailed here: leftTurn = 0, rightTurn = 1, brake = 2, go = 3, legPosHigh = 4, legPosLow = 5.

A screenshot of a cell phone

Description automatically generated

1. Once buttons have been created in the ArduinoBlue app, and the phone is connected to the Bluetooth module, buttons can be pressed to begin operation of the rover. The function of each button is in its name, and that function will perform until another button is pressed. This means that to turn right and stop, first the ‘rightTurn’ button must be pressed followed by the ‘brake’ button being pressed.

# Maintenance

This section covers the maintenance instructions for each sub-component assembly of the rover. The steps required to clean, remove and fix the rover components are listed below.

Body

1. Make sure frame is clean, welds on corners are in place, and all brackets are secure
2. If need, remove all components from frame, following steps from assembly section
3. Repair any damages to weldments, brackets, panels, etc.
4. Mount all components back onto frame following steps from assembly section

Screw Drive

1. Remove set screw from Screw Drive shaft
2. Pull off the 3D printed screw drive and check for any visual damages
3. Replace screw drive with new one if needed (follow steps in screw drive sub assembly)

Linear Actuators

1. Check actuators often, looking for any visual damages as well as wiring showing
2. If needed, remove the linear actuators from the frame following steps 1-4 of the assembly section
3. Test linear actuators using multimeter or Arduino setup
4. Replace actuator with new one

Arduino

1. Check connections and all ports on Arduino to make sure they are secure
2. Use compressed air to clean electrical connections regularly. This will prevent dust or other buildup from causing increased heat in the circuit while operational.

# Troubleshooting

The following section lists common issues with the rover component, with recommendations for fixing

Screw Drive -

1. If screw drive is not gripping the ground/ properly rotation, remove the drive shaft from the motor housing and check for broken components on the thread spine. If need, replace 3D printed part with new part. Follow steps 1-5 of the Screw Drive Assembly section of the manual.
2. Not Spinning when motor is turning, check to make sure connection between the set pin and motor itself is secure. If not, replace component with new screw drive.

Linear Actuators -

1. If linear actuators stop responding, remove linear actuator from rover, following steps 1-4 of the assembly section in reverse. Connect actuators to Arduino test circuit and see if actuator is operational. If not, replace the entire part.

Arduino –

1. If motors are not performing the desired commands (ex: motors are not rotating correct direction), modify the HIGH or LOW values in the digitalWrite commands of the movement functions and reupload code until the motor spins the desired direction when a button is pressed. DISCONNECT ANY BATTERIES PRIOR TO REUPLOAD.
2. If electrical components (motor driver, relay module, or Bluetooth module) are not functional, double check that connections are secure and look for burnt out parts. Substitute out any burnt fuses or components in the circuit. Try to use the brake function after every other function and allow the rover small moments of rest in between actions to prevent further circuit failure.

Appendix A

//Include Libraries

#include <SoftwareSerial.h>

#include <ArduinoBlue.h>

//Initialize Bluetooth Module

const int BLUETOOTH\_TX = 1;

const int BLUETOOTH\_RX = 0;

int button = -1;

SoftwareSerial bluetooth(BLUETOOTH\_TX, BLUETOOTH\_RX);

ArduinoBlue phone(bluetooth);

//Initialize Motors

int motor1 = 13;

int m11 = 12;

int m12 = 11;

int motor2 = 8;

int m21 = 9;

int m22 = 10;

int motor3 = 7;

int m11 = 6;

int m12 = 5;

int motor4 = 2;

int m11 = 3;

int m12 = 4;

//Initialize Vertical Actuator Motors

int yAct11 = 22;

int yAct12 = 24;

int yAct21 = 26;

int yAct22 = 28;

int yAct31 = 30;

int yAct32 = 32;

int yAct41 = 34;

int yAct42 = 36;

//Initialize Horizontal Actuator Motors

int xAct11 = 38;

int xAct12 = 40;

int xAct21 = 42;

int xAct22 = 44;

int xAct31 = 46;

int xAct32 = 48;

int xAct41 = 50;

int xAct42 = 52;

//setup code

void setup() {

//Begin signal input/output

Serial.begin(9600);

bluetooth.begin(9600);

delay(100);

//Set pinMode to OUTPUT

pinMode(motor1, OUTPUT);

pinMode(m11, OUTPUT);

pinMode(m12, OUTPUT);

pinMode(motor2, OUTPUT);

pinMode(m21, OUTPUT);

pinMode(m22, OUTPUT);

pinMode(motor3, OUTPUT);

pinMode(m31, OUTPUT);

pinMode(m32, OUTPUT);

pinMode(motor4, OUTPUT);

pinMode(m41, OUTPUT);

pinMode(m42, OUTPUT);

//Set Actuator pins to Outputs

pinMode(yAct11,OUTPUT);

pinMode(yAct12,OUTPUT);

pinMode(yAct21,OUTPUT);

pinMode(yAct22,OUTPUT);

pinMode(yAct31,OUTPUT);

pinMode(yAct32,OUTPUT);

pinMode(yAct41,OUTPUT);

pinMode(yAct42,OUTPUT);

pinMode(xAct11,OUTPUT);

pinMode(xAct12,OUTPUT);

pinMode(xAct21,OUTPUT);

pinMode(xAct22,OUTPUT);

pinMode(xAct31,OUTPUT);

pinMode(xAct32,OUTPUT);

pinMode(xAct41,OUTPUT);

pinMode(xAct42,OUTPUT);

//Set motor speed

analogWrite(motor1,100);

analogWrite(motor2,100);

analogWrite(motor3,100);

analogWrite(motor4,100);

analogWrite(motor5,100);

analogWrite(motor6,100);

}

void loop() {

// put your main code here, to run repeatedly:

button = phone.getButton()

String str = phone.getText();

if (button != -1) {

translateBT();

button = -1;

}

}

void translateBT(){

switch(button)//check results for input commands

{//start case

case 0: leftTurn(); break;

case 1: rightTurn(); break;

case 2: brake(); break;

case 3: go(); break;

case 4: legPosHigh(); break;

case 5: legPosLow(); break;

}//end case

}

void rightTurn(){

digitalWrite(m11, HIGH);

digitalWrite(m12, LOW);

digitalWrite(m21, LOW);

digitalWrite(m22, HIGH);

digitalWrite(m31, LOW);

digitalWrite(m32, LOW);

digitalWrite(m41, LOW);

digitalWrite(m42, LOW);

}

void leftTurn() {

digitalWrite(m11, LOW);

digitalWrite(m12, HIGH);

digitalWrite(m21, HIGH);

digitalWrite(m22, LOW);

digitalWrite(m31, LOW);

digitalWrite(m32, LOW);

digitalWrite(m41, LOW);

digitalWrite(m42, LOW);

}

void brake(){

digitalWrite(m11,LOW);

digitalWrite(m12,LOW);

digitalWrite(m21, LOW);

digitalWrite(m22, LOW);

digitalWrite(m31, LOW);

digitalWrite(m32, LOW);

digitalWrite(m41, LOW);

digitalWrite(m42, LOW);

}

void go(){

digitalWrite(m11, HIGH);

digitalWrite(m12, LOW);

digitalWrite(m21, HIGH);

digitalWrite(m22, LOW);

digitalWrite(m31, HIGH);

digitalWrite(m32, LOW);

digitalWrite(m41, HIGH);

digitalWrite(m42, LOW);

}

void legPosHigh(){

digitalWrite(yAct11,HIGH);

digitalWrite(yAct12,LOW);

digitalWrite(yAct21,HIGH);

digitalWrite(yAct22,LOW);

digitalWrite(yAct31,HIGH);

digitalWrite(yAct32,LOW);

digitalWrite(yAct41,HIGH);

digitalWrite(yAct42,LOW);

digitalWrite(xAct11,LOW);

digitalWrite(xAct12,HIGH);

digitalWrite(xAct21,LOW);

digitalWrite(xAct22,HIGH);

digitalWrite(xAct31,LOW);

digitalWrite(xAct32,HIGH);

digitalWrite(xAct41,LOW);

digitalWrite(xAct42,HIGH);

delay(3000);

}

void legPosLow(){

digitalWrite(yAct11,LOW);

digitalWrite(yAct12,HIGH);

digitalWrite(yAct21,LOW);

digitalWrite(yAct22,HIGH);

digitalWrite(yAct31,LOW);

digitalWrite(yAct32,HIGH);

digitalWrite(yAct41,LOW);

digitalWrite(yAct42,HIGH);

digitalWrite(xAct11,HIGH);

digitalWrite(xAct12,LOW);

digitalWrite(xAct21,HIGH);

digitalWrite(xAct22,LOW);

digitalWrite(xAct31,HIGH);

digitalWrite(xAct32,LOW);

digitalWrite(xAct41,HIGH);

digitalWrite(xAct42,LOW);

delay(3000);

}