Reporte de tarea 2.

Dinámica de Robots



“Movimiento de un robot”

Salguero Hernández Juan Pablo

\_\_\_\_\_\_\_

**Objetivo:**

Programar motor a pasos y hacer que gire lo indicado por el usuario mediante un nodo publicador de Ros..

**Materiales:**

* Computadora
* Sistema operativo Ubuntu 16.04
* Motor a pasos.
* Ros kinetic previamente instalado.
* Placa de programación Arduino
* Placa de programación Cypress CY8CKIT-42xx

Lo primero a realizar es cargar el código en la placa de Arduino.

**#include <ros.h>**

**#include <std\_msgs/Int16.h>**

**ros::NodeHandle nh;**

**int pot;**

**int retardo=5; // Tiempo de retardo en milisegundos (Velocidad del Motor)**

**int dato\_rx; // valor recibido en grados**

**int numero\_pasos = 0; // Valor en grados donde se encuentra el motor**

**String leeCadena; // Almacena la cadena de datos recibida**

**void potencia( const std\_msgs::Int16& cifra){**

**::pot = cifra.data;**

**}**

**ros::Subscriber<std\_msgs::Int16> sub("cifra", &potencia );**

**std\_msgs::Int16 res;**

**void setup() {**

**nh.initNode();**

**nh.subscribe(sub);**

**pinMode(11, OUTPUT); // Pin 11 conectar a IN4**

**pinMode(10, OUTPUT); // Pin 10 conectar a IN3**

**pinMode(9, OUTPUT); // Pin 9 conectar a IN2**

**pinMode(8, OUTPUT); // Pin 8 conectar a IN1**

**}**

**void loop() {**

**res.data = ::pot;**

**if (res.data>0){**

**res.data = (res.data \* 1.4222222222); // Ajuste de 512 vueltas a los 360 grados**

**}**

**while (res.data>numero\_pasos){ // Girohacia la izquierda en grados**

**paso\_izq();**

**numero\_pasos = numero\_pasos + 1;**

**}**

**while (res.data<numero\_pasos){ // Giro hacia la derecha en grados**

**paso\_der();**

**numero\_pasos = numero\_pasos -1;**

**}**

**leeCadena = ""; // Inicializamos la cadena de caracteres recibidos**

**apagado(); // Apagado del Motor para que no se caliente**

**nh.spinOnce();**

**delay(1000);**

**} ///////////////////// Fin del Loop ///////////////////////////**

**void paso\_der(){ // Pasos a la derecha**

**digitalWrite(11, HIGH);**

**digitalWrite(10, LOW);**

**digitalWrite(9, LOW);**

**digitalWrite(8, HIGH);**

**delay(retardo);**

**digitalWrite(11, LOW);**

**digitalWrite(10, HIGH);**

**digitalWrite(9, LOW);**

**digitalWrite(8, HIGH);**

**delay(retardo);**

**digitalWrite(11, LOW);**

**digitalWrite(10, HIGH);**

**digitalWrite(9, HIGH);**

**digitalWrite(8, LOW);**

**delay(retardo);**

**digitalWrite(11, HIGH);**

**digitalWrite(10, LOW);**

**digitalWrite(9, HIGH);**

**digitalWrite(8, LOW);**

**delay(retardo);**

**}**

**void paso\_izq() { // Pasos a la izquierda**

**digitalWrite(11, LOW);**

**digitalWrite(10, HIGH);**

**digitalWrite(9, HIGH);**

**digitalWrite(8, LOW);**

**delay(retardo);**

**digitalWrite(11, LOW);**

**digitalWrite(10, HIGH);**

**digitalWrite(9, LOW);**

**digitalWrite(8, HIGH);**

**delay(retardo);**

**digitalWrite(11, HIGH);**

**digitalWrite(10, LOW);**

**digitalWrite(9, LOW);**

**digitalWrite(8, HIGH);**

**delay(retardo);**

**digitalWrite(11, HIGH);**

**digitalWrite(10, LOW);**

**digitalWrite(9, HIGH);**

**digitalWrite(8, LOW);**

**delay(retardo);**

**}**

**void apagado() { // Apagado del Motor**

**digitalWrite(11, LOW);**

**digitalWrite(10, LOW);**

**digitalWrite(9, LOW);**

**digitalWrite(8, LOW);**

**}**

Y el código utilizado en la tarjeta cypress es el siguiente:

#include <ros.h>

#include <std\_msgs/Int16.h>

#include "StepperControl.h"

extern ros::NodeHandle nh;

namespace StepperControl {

const uint8\_t msec\_per\_step = 10;

int16\_t steps\_remaining = 0;

int16\_t abc = 0;

uint32\_t next\_time;

void messageCb( const std\_msgs::Int16& step\_msg){

steps\_remaining = step\_msg.data;

steps\_remaining = steps\_remaining\*1.4222222222;

}

ros::Subscriber<std\_msgs::Int16> sub("stepper\_move", &messageCb );

void setup()

{

nh.subscribe(sub);

next\_time = SysTimer::millis();

}

void paso\_der(){ // Pasos a la derecha

b2b\_Write(1);

b1b\_Write(0);

b2a\_Write(0);

b1a\_Write(1);

CyDelay(msec\_per\_step);

b2b\_Write(0);

b1b\_Write(0);

b2a\_Write(1);

b1a\_Write(1);

CyDelay(msec\_per\_step);

b2b\_Write(0);

b1b\_Write(1);

b2a\_Write(1);

b1a\_Write(0);

CyDelay(msec\_per\_step);

b2b\_Write(1);

b1b\_Write(1);

b2a\_Write(0);

b1a\_Write(0);

CyDelay(msec\_per\_step);

}

void paso\_izq() { // Pasos a la izquierda

b2b\_Write(0);

b1b\_Write(1);

b2a\_Write(1);

b1a\_Write(0);

CyDelay(msec\_per\_step);

b2b\_Write(0);

b1b\_Write(0);

b2a\_Write(1);

b1a\_Write(1);

CyDelay(msec\_per\_step);

b2b\_Write(1);

b1b\_Write(0);

b2a\_Write(0);

b1a\_Write(1);

CyDelay(msec\_per\_step);

b2b\_Write(1);

b1b\_Write(1);

b2a\_Write(0);

b1a\_Write(0);

CyDelay(msec\_per\_step);

}

void apagado() { // Apagado del Motor

b2b\_Write(0);

b1b\_Write(0);

b2a\_Write(0);

b1a\_Write(0);

}

void loop()

{

if ((int32\_t)(SysTimer::millis()-next\_time)>0) {

next\_time += msec\_per\_step;

if (steps\_remaining > abc) {

paso\_der();

abc++;

} else if (steps\_remaining < abc) {

paso\_izq();

abc--;

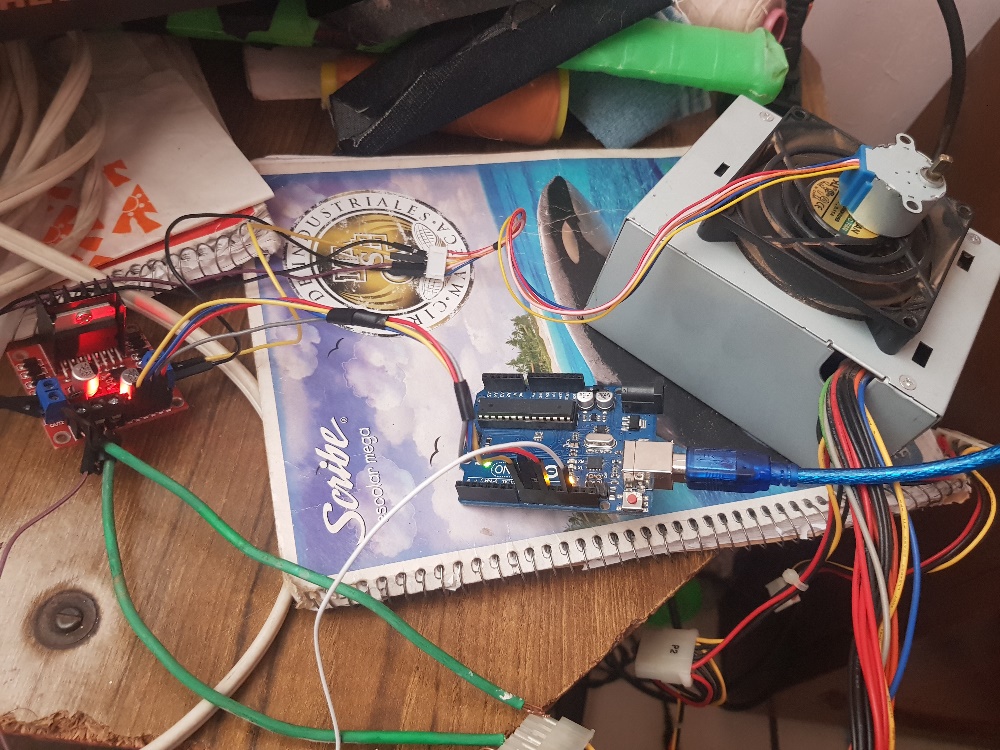
}

}

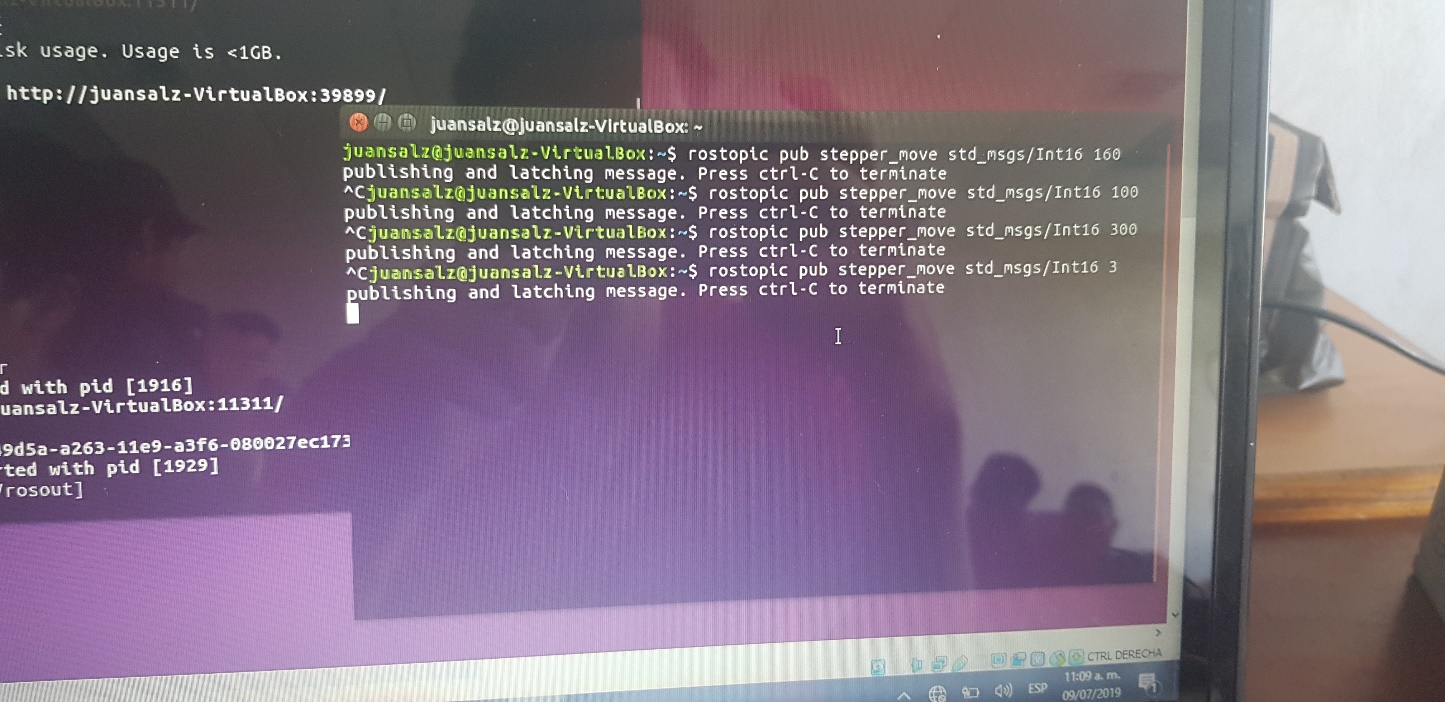
}

} // namespace StepperControl

El circuito consta de un motor a pasos conectado a un driver para su control, el driver utilizado es el L298 el cual tiene un soporte de hasta dos amperes y el motor se encuentra conectado a voltaje externo de una fuente de alimentación.



Para poder dar el dato del movimiento que queremos que realice nuestro robot es necesario tener el nodo maestro ejecutándose, una correcta conexión con el dispositivo y ejecutar la línea de código que se muestra en las siguientes imágenes:



Después de la palabra Int16 se coloca el numero que queremos enviar a nuestra tarjeta para que esta lo procese y ejecute logrando que le motor se mueva según el dato ingresado por el usuario.

**RESULTADO**

Como resultado se obtiene el control de un motor a pasos mediante el dato ingresado en el nodo publicador en ros, esto es de mucha utilidad para la programación de los motores para controlar el movimiento de un brazo robótico.