

Automatización de cúpula astronómica a escala

Andrés F. Guerrero *

Juan C. Riaño

*Universidad Nacional de Colombia, Facultad de Ciencias
Departamento de Física
Electrónica Digital*

Febrero 2022

Abstract

Se construyó un modelo escala de una cúpula astronómica automatizada. La cúpula tiene capacidad de ser controlada manualmente desde un joystick o desde un computador, siendo necesario en este último introducir la coordenada azimutal desde el puerto serial para girarla o bien utilizar un comando para abrirla y cerrarla. Junto a la cúpula se construyó una estación meteorológica que da uso a los sensores disponibles en el kit de arduino. Las medidas tomadas son: temperatura, humedad, velocidad del viento y luminosidad. Estas son características de suma importancia en un observatorio astronómico dado que caracterizan la noche de observación. Las medidas son usadas para conocer las condiciones óptimas en las que se realiza la observación, en caso que no se cumplan las condiciones mínimas, el observatorio activa una señal de aviso.

1 Descripción del montaje

Para el montaje se utilizaron varios dos motores, uno paso a paso y otro motorreductor. El primero de estos elementos está encargado de la rotación de la cúpula, de tal forma que la cúpula no obstruye la observación del cielo. El segundo motor se encarga de la cubierta de la cúpula, dispuesta para la protección de la lente telescópica de factores como la lluvia, fuertes vientos o material particulado.

La maqueta de cúpula está hecha de cartón industrial recubierto de fomi y unido mediante cinta. Esta es soportada mediante metal y tornillos, que dan libertad de movimiento a la compuerta y aseguran la estructura de la cúpula.

Para la realización del circuito se dio uso a la placa arduino Uno así como a diversos sensores que se explican cuidadosamente en la descripción del circuito, y que están dedicados a notificar de las circunstancias meteorológicas del instante de observación. Estos sensores fueron conectados al arduino mediante una protoboard y el cableado necesario para el correcto funcionamiento.

Uno de los elementos importantes de la estación meteorológica es el anemómetro, construido con un spinner y sensores de luminosidad que registran su velocidad de giro al ser impulsado por el viento.

La imagen total del montaje es la figura 10.

2 Descripción del circuito

Se desglosará cada una de las partes dado que se usaron diversos componentes que se deben explicar en detalle.

2.1 Motor principal

El funcionamiento mecánico de la cúpula (movimiento azimutal) se realizó mediante un motor paso a paso nema 17, que sostiene el peso de la cúpula en su totalidad.

Este sistema es el más eficiente dado que se disminuye al máximo la fricción, dejando un movimiento que depende únicamente del motor. Este motor tiene la capacidad de ejercer un torque de aproximadamente 2.2 kg/m y funciona a 12-35 V con

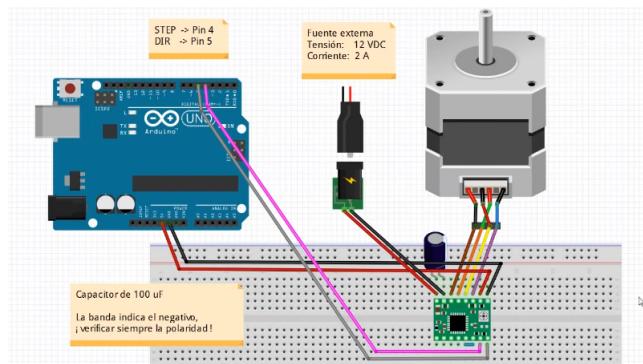


Figure 1. Imagen descriptiva del circuito para el motor principal.

una corriente máxima de 1.2 A. Para poder controlar el motor desde arduino es necesario un controlador A4988 "Pololu".

Para su uso se conecta el circuito al controlador A4988 y se da la alimentación de 5V al integrado directamente del arduino. El valor de la diferencia de potencial de referencia suministrada por el controlador al motor se debe regular, pues este valor de voltaje es diferente según el motor que se usa y depende de la corriente máxima a la que opera el motor. Este valor es obtenido mediante la ecuación:

$$V_{ref} = I_{max} * 8 * R_s \quad (1)$$

El controlador trae integrada una resistencia $R_s = 0.1\Omega$, y, como se dijo anteriormente, el motor usado trabaja con una corriente máxima de 1.2 A. Con estos valores se ajustó el voltaje de referencia en $V_{ref} = 0.8V$.

2.1.1 Conexión a Arduino

Los pines del arduino VMOT Y GND se alimentan con una fuente de 12 V con un condensador de $100\mu F$ requeridos por el fabricante del controlador por el caso en que el motor requiera que se almacene energía y posteriormente sea liberada en pequeñas cantidades.

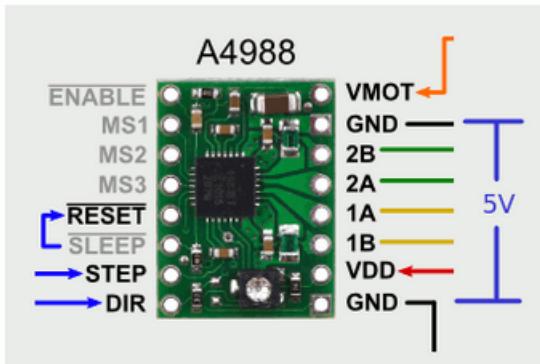


Figure 2. Esquema del controlador A4988

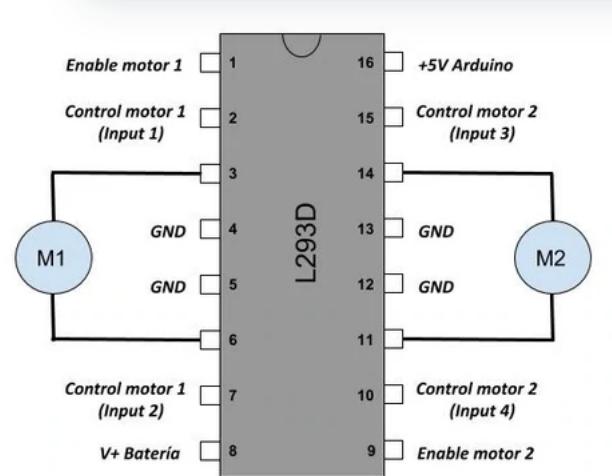


Figure 4. Datasheet del puente H integrado en el L293D

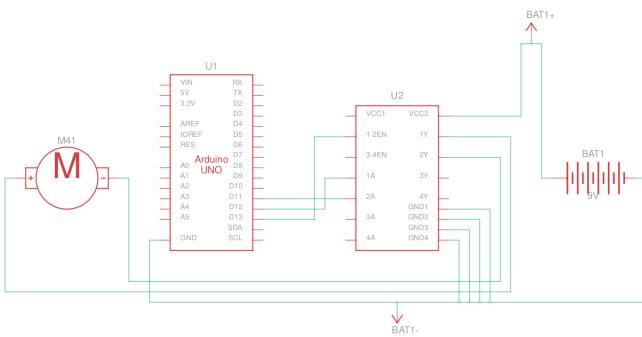


Figure 3. Esquema del circuito para el motor de la compuerta de la cúpula

Los pines 1A, 1B, 2A y 2B son conectados a las salidas respectivas del motor paso a paso nema 17; estos corresponden a las bobinas del motor y se encargan del movimiento ajustado.

Los pines VDD y GND son conectados a 5 V que son suministrados directamente por el arduino. El pin DIR del controlador es conectado al pin 5 de arduino y es el encargado de determinar la dirección de giro del motor, es decir, con un nivel HIGH gira en un sentido, con un nivel LOW gira en el sentido contrario. El pin STEP del controlador es conectado al pin 4 de arduino y este será el encargado de hacer que el motor gire.

Finalmente, el pin RESET del controlador es conectado al pin 9 de arduino y es el encargado de apagar el controlador cuando no se requiera movimiento del motor. Esto se hace para que el motor no sufra, es decir no se sobrecaliente y pueda tener una vida útil más prolongada.

2.2 Motor de la compuerta

Para la automatización de la compuerta de la cúpula se usó un motorreductor conectado a 12 V. Este elemento posee un torque máximo de 2 kg/cm. Para realizar el control del motor con arduino es necesario un controlador, que en este caso es un puente H incorporado en el integrado L293D.

El controlador L293D es un controlador para dos motores sencillos o uno solo de paso, en este caso, dado que solo se usa el integrado para manejar un motor, no se hace uso del módulo completo.

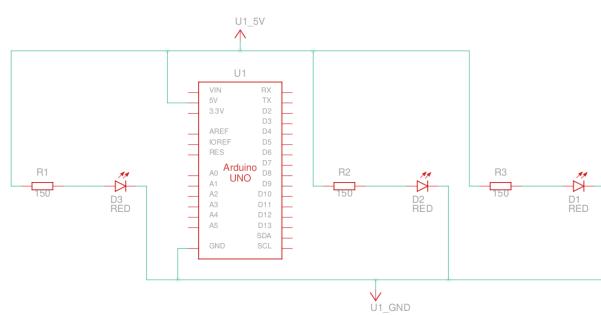


Figure 5. Esquema del circuito de iluminación del domo

2.2.1 Conexión a Arduino

El pin 1 del controlador, que maneja la velocidad del motor, se conectó al pin 11 del arduino, mientras que el pin 2 del puente H se conectó al pin 12 del arduino.

El pin 7 del controlador, que va dirigido al pin 13 del arduino, se encarga del control del motor mediante salidas LOW y HIGH según la dirección en la que se quiera mover el motor, similar al dispositivo anterior.

Las salidas 4, 5, 12 y 13 del integrado son todas conectadas a tierra. Las salidas 3 y 6 son conectadas al motor sin importar su polaridad inicial. El pin 8 del controlador es alimentado con una fuente de 12 V, la misma que alimenta al motor principal. Finalmente, el pin 16 del controlador es alimentado con 5 V que vienen directamente del arduino.

2.3 Iluminación del domo

La iluminación del domo es sencilla pues consiste en tres leds rojos conectados en paralelo con un resistencia de 150 Ω cada uno.

2.4 Control Manual (Joystick)

Para controlar el movimiento tanto del motor principal como del motor de la cúpula se usó un joystick. En este joystick el

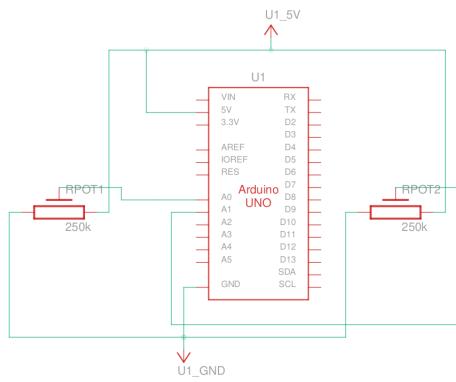


Figure 6. Esquema del circuito usado para conectar el Joystick

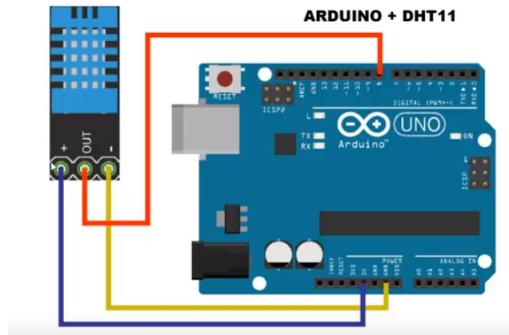


Figure 7. Esquema del circuito usado para el sensor de temperatura

eje *x* controla compuerta de la cúpula, es decir, la apertura de la misma. Cuando se sitúa el joystick hacia la derecha la compuerta se cierra y cuando se sitúa hacia la izquierda se abre.

Por otro lado, el eje *y* se encarga del movimiento rotacional de la cúpula. Cuando el joystick se baja de su posición natural, la cúpula gira en sentido de las manecillas del reloj, cuando el joystick se sube respecto a su posición inicial, la cúpula gira en sentido anti-horario.

Cuando el joystick es dejado en su posición de reposo, la cúpula y su compuerta permanecerán inmóviles en el estado actual hasta que reciban alguna señal.

2.4.1 Conexión a arduino

Para modelar el joystick se usaron dos potenciómetros, uno para el eje *x* y el otro para el eje *y*. El sensor del kit arduino trae 5 pines de conexión. Uno de ellos va a 5 voltios alimentados directamente del arduino, y otro va a tierra también conectada del arduino, mientras que las dos salidas analógicas fueron conectadas a A0 y A1 respectivamente.

2.5 Temperatura y humedad

Para la medida de temperatura y humedad se dio uso al sensor DHT-22 así como a las librerías necesarias para su uso. Este sensor provee una salida digital altamente precisa, pero limitada a un tiempo mínimo de dos segundos entre medidas.

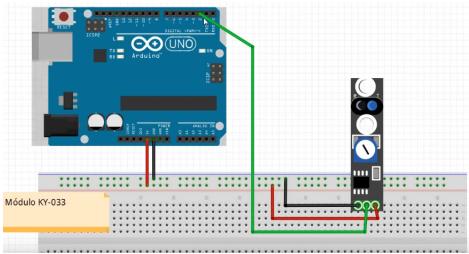


Figure 8. Esquema del circuito usado para el sensor de seguimiento usado para medir la velocidad del viento.

2.5.1 Conexión a arduino

El sensor de temperatura y humedad es alimentado con los 5 V del arduino, mientras que su salida es digital es conectada al pin 2 de arduino.

Es importante notar que este sensor solo tiene una salida para su lectura pero que está leyendo dos variables distintas. Este detalle sera ajustado mediante librerías especializadas que se describen en la sección de código fuente.

2.6 Anemómetro

Para la construcción del anemómetro se hizo uso del sensor de seguimiento HW-511 TCRT5000 que consiste en una fuente y un emisor de infrarrojo.

Para poder medir la velocidad del viento se dispuso de un spinner tal que una de sus aspas es de color blanco, dicho color con el propósito de que refleje luz infrarroja. Las otras dos son de color negro para que reflejen tan poco como sea posible.

Se ubica el sensor cerca de las aspas, con lo que cada vez que pase la asta con el color blanco, empujada por el viento, el sensor de rastreo la detecta y envía un pulso al arduino. Con la diferencia de tiempo entre cada paso del brazo blanco del molino, así como las dimensiones del aspa, es posible encontrar la velocidad del viento.

2.6.1 Conexión a arduino

Dada la simpleza del sensor en cuanto conexión, basta con alimentarlo a los 5 V dados por el arduino, mientras su pin de salida va al pin 7 de la placa arduino.

2.7 Luminosidad

La medida de luminosidad, importante para la toma de fotografías, se mide mediante un sensor GL55.

2.7.1 Conexión a arduino

La fotoresistencia se implementó alimentando al sensor con los 5 V del arduino, llevando su salida al pin 3.

3 Código fuente

El código fuente inicia con las librerías que se deben importar.

Solo es necesario incluir DHT_U.h y su complemento DHT.H pues solamente el sensor de temperatura y humedad necesita una librería especializada. Este sensor necesita la librería debido a que la entrada en las medidas de temperatura y humedad es la misma, y la librería posee la herramienta para separar cada una.

3.1 Librerías y Pines

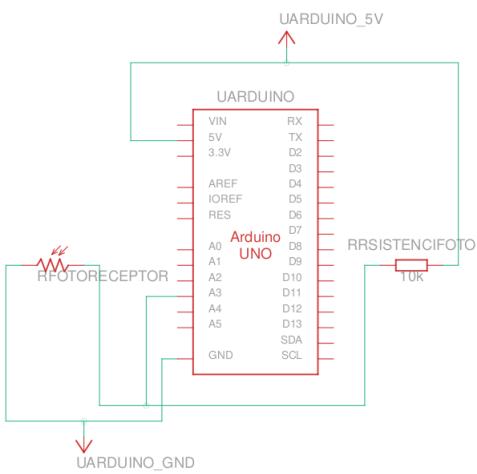


Figure 9. Esquema del circuito usado para la fotoresistencia

```
#include <DHT_U.h>
#include <DHT.h>
```

A continuación se definen todas los pines que serán usados tanto por los sensores como por los controladores de los motores. El propósito de cada pin ya fue especificado en la sección anterior:

```
#define STEP 4
#define DIR 5
#define RESET 9
#define LDR A3
#define TempHum 2
#define speedPin 1
#define dirPin1 12
#define dirPin2 13
#define Viento 7
#define xpin A0
#define ypin A1
```

3.2 Variables

Seguidamente se definen todas las constantes y variables necesarias para los cálculos a realizar.

El entero *LUMINOSIDAD* guarda el valor leído de la entrada A0, mientras que *ValorMapeado* guarda el valor de luminosidad a trabajar.

```
int LUMINOSIDAD;
int ValorMapeado;
```

Ahora se establecen los enteros de *TEMPERATURA* y *HUMEDAD* que almacenan los valores que da el sensor DHT-22 tras ser separados por su respectiva librería.

```
int TEMPERATURA;
int HUMEDAD;
```

Dada la forma en la que se calcula la velocidad del tiempo, se necesitan una mayor cantidad de variables. Aquí los enteros almacenan los valores leídos por el sensor de seguimiento, mientras que los demás almacenan vueltas de aspas, tiempo

recorrido, las especificaciones del aspa, los términos necesarios para el cálculo de la velocidad y su respectiva conversión.

```
int Vueltas;
int ANTERIOR = 1;
float tt;
bool b;
int vueltas;
float t1 = 0;
float t2;
float r = 4E-5;
float pi = 3.1415926535897932384626433832795;
float w;
float v;
float mlah = 2.777E7;
float ts;
```

Las variables de los ejes de la cúpula ya se han descrito anteriormente. Aquí simplemente se define la variable a almacenarlos, pero también se establece la velocidad del motor de apertura de la cúpula.

```
int speedMotor = 1200;
int xVal;
int yVal;
```

Las variables requeridas para el uso de la cúpula remota se establecen aquí, en que algunas relevantes son *azimut*, que se utiliza como el ángulo al que se desea mover la cúpula, mientras que los demás establecen la cantidad a moverse con base en este dato y la posición anterior.

```
float paso;
int posicion = 0;
int azimut;
int dato;
int dif;
int mov;
int aux;
```

Por último se tienen variables generales del funcionamiento de la cúpula, relacionados con su apertura y movimiento, así como el float *velocidad* dedicado a la velocidad del viento.

```
void abrir();
void cerrar();
void moviz(int pasos);
void movder(int pasos);
float velocidad( float Valor);
```

3.3 Inicialización de pines

Esta sección del código inicializa y define cada uno de los pines como entradas o salidas según correspondan.

```
void setup()
{
    Serial.begin(9600);

    pinMode(STEP, OUTPUT);
    pinMode(DIR, OUTPUT);
    pinMode(RESET, OUTPUT);
    pinMode(speedPin,OUTPUT);
    pinMode(dirPin1,OUTPUT);
    pinMode(dirPin2,OUTPUT);

    pinMode(xpin, INPUT);
```

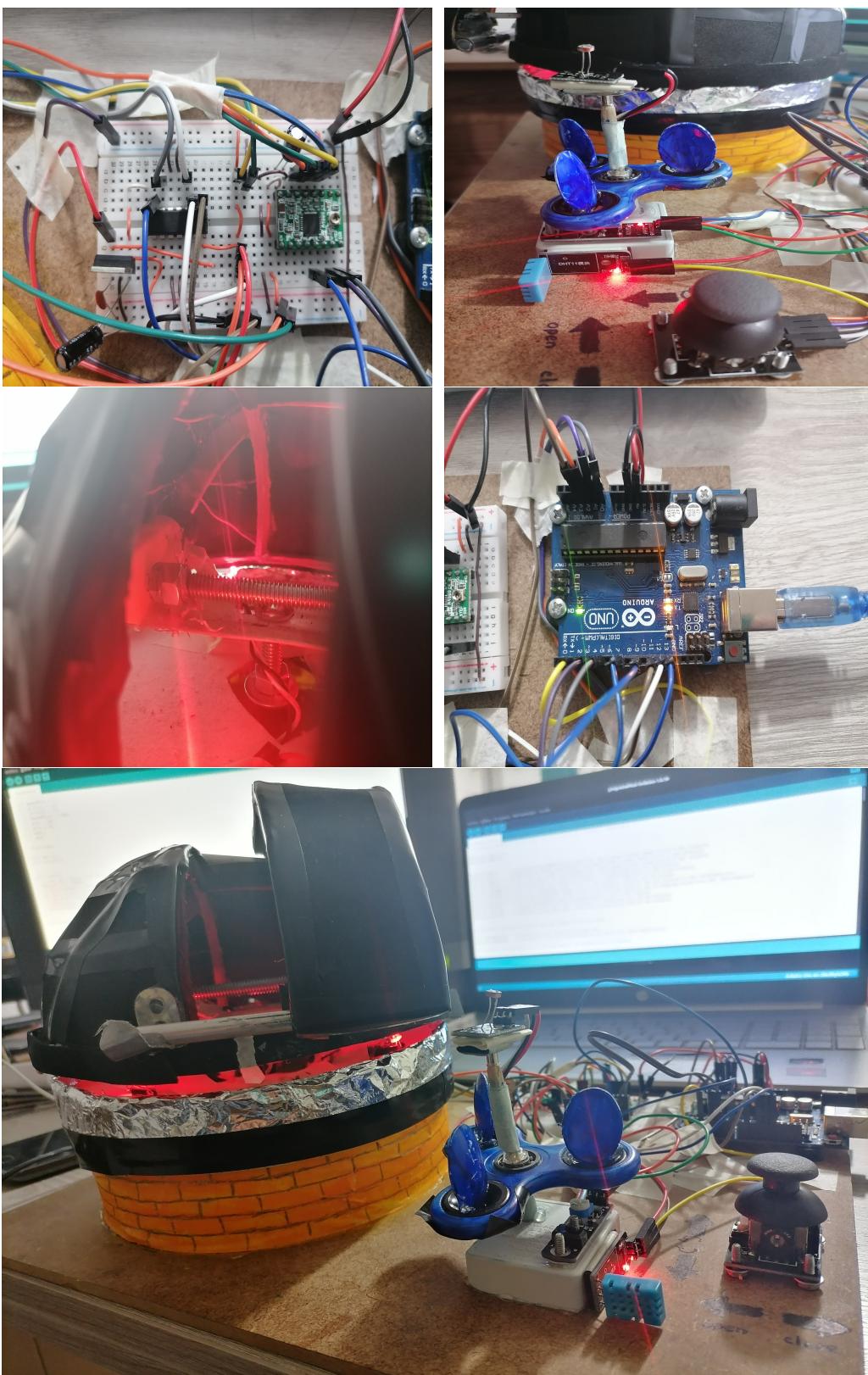


Figure 10. Fotos detalladas de cada parte del montaje. a) La imagen superior izquierda muestra un protoboard con los dos controladores de los motores de movimiento de la cúpula, así como las demás conexiones requeridas para los sensores. b) En la imagen superior derecha se observa el montaje de la estación meteorológica en el que se ve claramente el sensor de fotoresistivo, el sensor de temperatura y humedad, y el sensor de seguimiento para el anemómetro. c) La imagen central izquierda presenta los dos motores al interior de la cúpula y deja ver claramente los pernos que mueven todo el sistema. d) La imagen central derecha muestra las conexiones en el arduino. e) Foto del montaje en su totalidad.

```

pinMode(ypin, INPUT);
pinMode(Viento, INPUT);

dht.begin();

Serial.println("#Ubicacion,tiempo ,temperatura,
    ↪ humedad, viento, luminosidad");
}

```

3.4 Operaciones con las entradas de los sensores

En esta sección se escriben los datos de los sensores como salidas codificables que se imprimen.

```

DHT dht(TempHum, DHT11);
void loop()
{
    while(Serial.available() == 0){

        TEMPERATURA = dht.readTemperature();
        HUMEDAD = dht.readHumidity();
        tt = millis();
        ts = tt /1000;
        Vueltas = digitalRead(Viento);
        LUMINOSIDAD = analogRead(LDR);
        ValorMapeado = map(LUMINOSIDAD, 0, 1023, -10, 30);
        v = velocidad(Vueltas);
        Serial.print(posicion);
        Serial.print(" ");
        Serial.print(ts);
        Serial.print(" ");
        Serial.print(HUMEDAD);
        Serial.print(" ");
        Serial.print(TEMPERATURA);
        Serial.print(" ");
        Serial.print(v);
        Serial.print(" ");
        Serial.println(ValorMapeado);
    }
}

```

Aquí algunos comandos relevantes son *digitalRead* y *analogRead*, que reciben y entienden como señal digital o analógica, la entrada dada por el sensor respectivo.

3.5 Control remoto de la cúpula

En esta sección del código se leen los datos del usuario y se utilizan como datos de apertura de la cúpula.

```

dato = Serial.parseInt();
// Se ha elegido 500 como el dato del usuario
if(dato == 500){
    digitalWrite(RESET, HIGH);
    digitalWrite(DIR, LOW);

    Serial.println("abriendo ...");
    abrir();
    Serial.println("Cúpula abierta");

    delay(20);
    digitalWrite(RESET, LOW);
}

// Se ha elegido 600 como el dato del usuario
if(dato == 600){
    digitalWrite(RESET, HIGH);
    digitalWrite(DIR, LOW);
}

```

```

Serial.println("cerrando...");
cerrar();
Serial.println("Cúpula Cerrada");

digitalWrite(RESET, HIGH);
delay(20);
digitalWrite(RESET, LOW);
}

```

La rotación de la cúpula se decide mediante un ángulo dado por el usuario, que posteriormente se lleva a expresiones condicionales que lo traducen a pasos de motor.

```

if (dato > 0 and dato <= 360 ){

    dif = posicion - dato;
    Serial.print("El angulo a mover es: ");
    Serial.println(dif);

    if(dif < 0 and dif >= -180){

        mov = abs(dif);
        paso = 0.5555 * mov ;
        paso = int(paso);
        moviz(paso);
        posicion = dato;
    }

    if (dif > 0 and dif <= 180){

        mov = abs(dif) ;
        paso = 0.5555 * mov ;
        paso = int(paso);
        movder(paso);
        posicion = dato;
    }

    if (dif > 180 and dif < 360){

        mov = 360 - abs(dif);
        paso = 0.5555 * mov ;
        paso = int(paso);
        moviz(paso);
        posicion = dato;
    }

    if (dif < -180 and dif > -360){

        mov = 360 - abs(dif) ;
        paso = 0.5555 * mov ;
        paso = int(paso);
        movder(paso);
        posicion = dato;
    }
}

```

3.6 Movimiento manual de la cúpula

En esta sección se estable el código de control de la cúpula mediante el joystick, con la primera parte dedicada al almacenamiento de salidas analógicas en sus respectivas variables de *x* e *y* del joystick.

```

while (dato == 1000){ // El comando para activar
    ↪ el modo manual se define como 1000

    xVal = analogRead(xpin);
    yVal = analogRead(ypin);
    Serial.println(xVal);
}

```

Aquí se programa para que al mover el joystick en cada una de las direcciones posibles haya una rotación horaria o anti-horaria.

```

if(xVal > 800){
    digitalWrite(RESET, HIGH);
    delay(1);
    digitalWrite(DIR, HIGH);
    digitalWrite(STEP, HIGH);
    delay(15);
    digitalWrite(STEP, LOW);
    delay(15);

}

if(xVal < 200){
    digitalWrite(RESET, HIGH);
    delay(1);
    digitalWrite(DIR, LOW);
    digitalWrite(STEP, HIGH);
    delay(15);
    digitalWrite(STEP, LOW);
    delay(15);

}

```

Si el joystick está en reposo, se debe apagar el controlador del motor.

```

if(xVal <=800 and xVal >=200){
    digitalWrite(RESET, LOW);
    //digitalWrite(STEP, LOW);
}

```

Para la apertura y cerradura del joystick tenemos esta parte, en que es importante que cuando el joystick está en reposo, el motor se apaga.

```

if(yVal > 800){
    digitalWrite(dirPin1, 1);
    digitalWrite(dirPin2,0);
    analogWrite(speedPin, speedMotor);
}

if(yVal < 200){
    digitalWrite(dirPin1, 0);
    digitalWrite(dirPin2,1);
    analogWrite(speedPin, speedMotor);
}

if(yVal <= 800 and yVal >= 200){
    digitalWrite(dirPin1, 0);
    digitalWrite(dirPin2,0);
    analogWrite(speedPin, speedMotor);
}

```

3.7 Funciones relevantes

La siguiente función es aquella dedicada a activar el motor de apertura o cerradura de la cúpula durante 9.5 segundos.

```

void abrir(){
    digitalWrite(dirPin1, 1);
    digitalWrite(dirPin2,0);
    analogWrite(speedPin, speedMotor);
    delay(9500);
    digitalWrite(dirPin1, 0);
    digitalWrite(dirPin2,0);
    analogWrite(speedPin, speedMotor);
}

void cerrar(){
    digitalWrite(dirPin1, 0);
    digitalWrite(dirPin2,1);
}

```

```

analogWrite(speedPin, speedMotor);
delay(9500);
digitalWrite(dirPin1, 0);
digitalWrite(dirPin2,0);
analogWrite(speedPin, speedMotor);
}

```

En el caso en que la humedad alcanza un límite de tolerancia resulta conveniente que la cúpula se cierre fortuitamente en vez de simplemente dar un aviso. El siguiente código se encarga de ello

```

if (HUMEDAD > 90 and aux_h == false){
    Serial.print("La humedad ha superado el límite de
        ↪ tolerancia");
    cerrar();
    aux_h = true;
}

if (HUMEDAD <= 90){
    aux_h = false;
}

```

Esta función recibe como parámetro el número de pasos que se desea mover la cúpula y activa el motor que gira la cúpula hasta el ángulo pedido.

```

void moviz(int pasos){
    digitalWrite(RESET, HIGH);
    digitalWrite(DIR, HIGH);
    for(int i = 0; i < pasos; i++){
        digitalWrite(STEP, HIGH);
        delay(10);
        digitalWrite(STEP, LOW);
        delay(10);
    }
    digitalWrite(RESET, LOW);
}

```

Es de notar que el dato de rotación lo recibe como grados, pero tiene en cuenta que 200 pasos equivalen a una rotación completa de la cúpula (360 grados).

La función velocidad recibe el valor del sensor de seguimiento y devuelve la velocidad del viento calculando la velocidad angular de las aspas, dejándola en kilómetros por hora.

```

float velocidad( float VALOR){
    if (VALOR == 0 and b == false){
        vueltas = vueltas + 1;
        b = true;

        t2 = tt - t1;
        t1 = tt;

        w = 2*pi / t2;
        w = w * mlah;
        v = r*w ;
    }
    if (VALOR == 1){
        b = false;
    }
    return v;
}

```

4 Dificultades encontradas

Una de las principales dificultades encontradas tiene que ver con el motor paso a paso con el que se realizó la rotación

de la cúpula, pues anteriormente se utilizaba un motor lineal activado durante un tiempo dado asociado a cierto valor de rotación. Este funcionamiento resultó inexacto y complicado de manejar debido a la fricción de la cúpula con la superficie, por lo que se hizo el cambio a un motor paso a paso con el que ya fue posible realizar la manipulación de la cúpula.

Otra problemática que se tuvo tiene que ver con la placa arduino que, por la forma en que ejecuta los programas, tiene la imposibilidad de tomar medidas mientras se cierra la compuerta. Si bien esto no afecta al funcionamiento óptimo de la cúpula, no resulta ideal.

Finalmente, se tiene que el montaje del sensor de viento, dado el peso del spinner, no termina dar medidas precisas de la velocidad del viento. Sobre esta precisión de las medidas se puede mencionar otro problema con la toma de datos de los sensores. Por ejemplo, el sensor de temperatura y humedad solo puede tomar medidas cada 2 segundos, que si bien no afectan importantemente a la cúpula, pueden ser necesaria continuidad para mediciones concretas de la bóveda celeste.

En el caso del sensor del viento se tiene la posibilidad de que, por la forma en que el sensor de fotorresistencia toma medidas, se ignoren ciertos pasos de las aspas por contaminación lumínica. Si bien es un caso límite que no resulta probable, también es posible que las aspas vayan lo suficientemente rápido como para que el sensor no mida un paso del aspa blanca y calcule equivocadamente el valor de la velocidad.

Algunos problemas técnicos tienen que ver con la cantidad de cableado, puesto que debido a su abundancia se produjeron múltiples cortos que condujeron al daño de elementos relevantes, como el motor. Además, en varias ocasiones, debido a la rotación de la cúpula, los cables de la compuerta se enredaron y condujeron a problemas adicionales.

5 Sugerencias

Dado que el motor paso a paso funcionó correctamente para la maqueta, una cúpula astronómica real podría valerse de un motor similar, pero con una fuerza de torque apropiada al tamaño de la cúpula.

Para la problemática de ejecución del programa se sugiere utilizar múltiples placas arduino o un sistema similar más complejo, así como sensores que puedan obtener medidas más decisivas y sofisticadas.

Una solución para las medidas del viento es la utilización de un anemómetro especializado, que no posea ninguno de los problemas descritos y que dé inmediatamente el valor de la velocidad del viento.

Para solucionar problemas sobre el montaje se sugiere trabajar con objetos que involucren directamente cada uno de los sensores y que se aprovechen de la miniaturización existente de los instrumentos.

En cuanto la rotación y el cableado de la compuerta, puede utilizarse activación mediante señales y una alimentación autónoma al resto del circuito, de tal modo que el cableado no sufra al aplicar una rotación.

6 Referencias

Arduino.cc. (2015). *Arduino*. [online] Disponible en: <http://arduino.cc/en>

Aosoong Electronics Co., Ltd. (n.d.). *Digital-output relative humidity & temperature sensor/module*. [online]

Disponible en: <https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf>

Arduino Uno. (2017). *Arduino Uno Datasheet* [online] Disponible en: <https://www.farnell.com/datasheets/1682209.pdf>

Arduino Modules. (2021). *KY-022 INFRARED RECEIVER MODULE* [online] Disponible en: <https://arduinomodules.info/ky-022-infrared-receiver-module/>

Astroshop.es. (2022). *Construcción de observatorios* [online] Disponible en: https://www.astroshop.es/acerca-de-nosotros/construccion-de-observatorios/i_1191

Cambatronics Online. (2018). *DHT22 : Usando arduino para leer temperatura y humedad*. [online] Disponible en: <https://www.youtube.com/watch?v=55C9Jwd1LDQ>

Cardona, J & Blanco, O. (2012). *Electrónica Digital y su Aplicación a la Instrumentación: guía de laboratorio*. Facultad de Ciencias. Universidad Nacional de Colombia.

Components 101. (2021) *Joystick Module*. [online] Disponible en: <https://components101.com/modules/joystick-module>

Components 101. (2019) *NEMA 17 Stepper Motor*. [online] Disponible en: <https://components101.com/motors/nema17-stepper-motor>

D& R Tutoriales. (2020). *Módulo Joystick con Arduino*. [online] Disponible en: <https://www.youtube.com/watch?v=wQm0j0xcRHU>

eeeBox. (2019). *Line Tracking Sensor HW-511*. [online] Disponible en: <https://eeebboxbd.com/product/line-tracking-sensor-hw-511/>

freeCodecamp.org. (2021). *Arduino Course for Beginners - Open-Source Electronics Platform*. Disponible en: https://www.youtube.com/watch?v=zJ-LqeX_fLU

Materiales de Laboratorio. (n.d.). *Anemómetro*. [online] Disponible en: <https://materialeslaboratorio.com/anemometro/>

Mano, M. M. (1982). *Lógica digital y diseño de computadores*. Prentice Hall.