Universidad Nacional Autónoma de México
Facultad de Ingeniería

Proyecto: Eslabón adaptable

Integrantes:
- Carmona Valdés Julio César
- Maldonado Martínez Miguel
- Viruell Asai Ana Regina

Materia: Robótica

Profesor: Erik Peña Medina

Grupo: 1

Semestre: 2024-1

CONTENIDO

OBJETIVO

Diseñar e implementar un eslabón adaptable el cual pueda modificar su longitud del cual se pueda sensar las aceleración angular y la corriente que consumen sus actuadores.

METAS

- Implementar un control de posición para el motor Dynamixel AX – 18A utilizando el Dynamixel Arduino Shield.

  En el diseño final del proyecto se cambiará el motor a Dynamixel XL430 – W250 – T.

- Implementar un control de posición para la junta prismática del eslabón empleando un potenciómetro lineal.
- Implementar un interfase de sensado de la aceleración angular del robot y la corriente consumida por sus actuadores.
- Diseñar los elementos de la estructura mecánica del eslabón adaptable para implementarlos en un entorno de simulación en Mathlab y/o ROS 2 Humble.
- Construir físico un prototipo del eslabón adaptable.
- Realizar pruebas de funcionamiento del eslabón.

CONTEXTO

En este reporte se mencionan los avances realizados del proyecto, relacionados con el objetivo y metas de este. A grandes rasgos, se divide en el uso del *Dynamixel Wizard*, el desarrollo de un control por posición con *Dynamixel Arduino Shield*, y un control de velocidad adicional con potenciómetro rotativo. Respecto al prototipo físico, se determinó el "frame" o bracket comercial compatible con el motor que mejor se adecue al movimiento deseado del proyecto. En base a ello, se modificó el *frame* para añadir el primer eslabón que será movido por el motor *Dynamixel* para finalmente fabricarlo con manufactura aditiva.

Cabe mencionar que en las pruebas con los motores se utilizaron ambos motores, AX – 18A y XL430 – W250 – T, con el fin de comprender en mayor escala el funcionamiento de estos. Asimismo, los procedimientos explicados pueden replicarse para ambos, pues estos cuentan con características similares, con algunos ligeros cambios que serán explicados en cada sección.

Si bien existen diferentes maneras de controlar y manejar los motores Dynamixel, los procedimientos mostrados en este reporte fueron realizados en función de las herramientas y componentes disponibles.

Así, para comenzar, se presentan los enlaces de las fichas técnicas de los componentes principales:

Documentación o Manual de Dynamixel AX – 18 A (referencia [1]):
https://emanual.robotis.com/docs/en/dxl/ax/ax-18a/

Documentación o Manual de Dynamixel XL430 – W250 – T (referencia [2]):
https://emanual.robotis.com/docs/en/dxl/x/xl430-w250/

Documentación o Manual de Dynamixel Shield (referencia [3]):
https://emanual.robotis.com/docs/en/parts/interface/dynamixel_shield/

DYNAMIXEL WIZARD

Inicialmente, en el proyecto se utilizaría un motor Dynamixel AX – 18A, sin embargo, para el diseño final del eslabón adaptable se optó por cambiar a un motor Dynamixel XL430 – W250 – T. Aun así, al ser del mismo fabricante con funcionamientos similares, las pruebas de control realizadas fueron hechas con ambos motores. El primer acercamiento o avance del proyecto se enfocó en comprender el funcionamiento de los motores mediante el Dynamixel *Wizard*. A su vez, se realiza una tabla comparativa de los parámetros o variables mostradas en *Wizard* para ambos productos.

El *Dynamixel Wizard* es una herramienta proporcionada por Robotis (fabricante de los motores Dynamixel) que ayuda a manejar dichos motores. Dentro de sus características se encuentra que la compatibilidad con los motores de la marca de serie AX, MX, X, PRO, y Robot Hand. Además, permite conexiones de múltiples motores, herramientas de gráficas y análisis, y auto diagnósticos.

A continuación, se explica el procedimiento para utilizar el Dynamixel Wizard, desde los componentes necesarios, conexión de estos, e interfaz del Wizard.

DESCARGA DEL SOFTWARE

Para utilizarlo es necesario descargar el software desde (referencia [4]):

https://emanual.robotis.com/docs/en/software/dynamixel/dynamixel_wizard2/

COMPONENTES NECESARIOS

Existen diferentes maneras de conectar el (los) motores al Dynamixel Wizard dependiendo de los componentes que estén disponibles. En este caso se utilizaron los siguientes:

1. Un convertidor U2D2 Dynamixel: Permitirá la comunicación del servo motor a una computadora PC mediante una conexión USB. El conector incluye un puerto TTL, UART y RS – 485. Para el proyecto se utilizará conexión TTL con el motor.

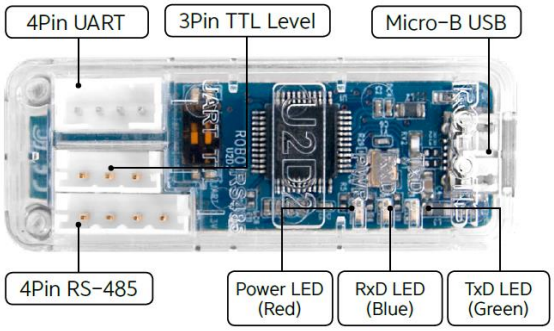

Figura 1 [5]. U2D2

2. U2D2 Power Hub Board (PHB): Este componente permite proporcionar energía estable al motor.

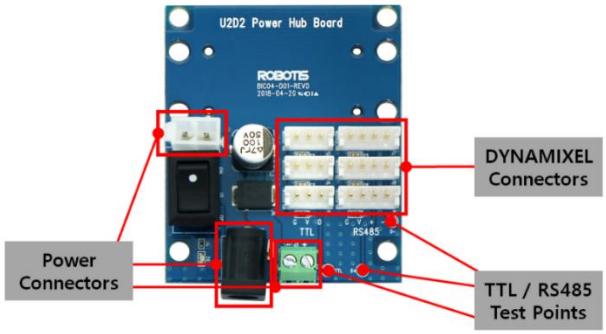

Figura 2. [6] U2D2 Power Hub Board (U2D2 PHB)

Se recomienda ensamblarlo con el U2D2 tanto mecánica como electrónicamente, como se muestra en las siguientes imágenes.

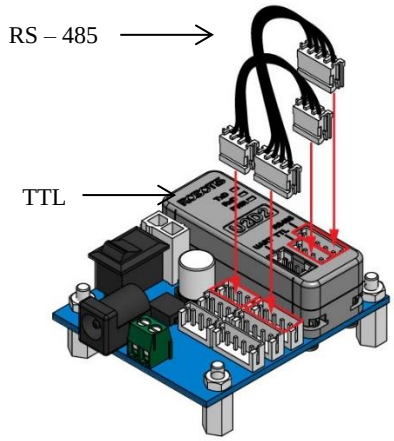

Figura 4. [6] Ensamble mecánico de U2D2 al Power Hub Board

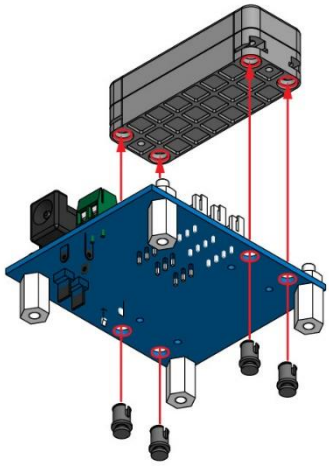

Figura 3. [6] Ensamble mecánico de U2D2 al Power Hub Board

Para más información sobre los conectores, véase referencia [6].

Nótese de la Figura 4 que en este caso únicamente es necesaria la conexión TTL.

3. Adaptador SMPS 12V 5A AC: Adaptador que se conectará al U2D2 PHB para proporcionar la energía desde una fuente de corriente alterna.

Figura 5. [7] Adaptador SMPS

4. Conectores: 2 conectores 3P para las conexiones TTL del U2D2 al U2D2 PHB y del U2D2 PHB al motor. Además de un cable micro USB para la conexión del U2D2 a la computadora PC.

CONEXIÓN

El procedimiento de conexión es igual para los dos motores de interés, AX – 18A y XL430 – W250 – T, ya que ambos cuentan con un cable TTL 3P.

Para comenzar la conexión, se debe asegurar que el U2D2 esté ensamblado al U2D2 PHB con el conector de TTL, como se muestra en la Figura 4.

1. Conectar un extremo del cable TTL al U2D2 PHB.

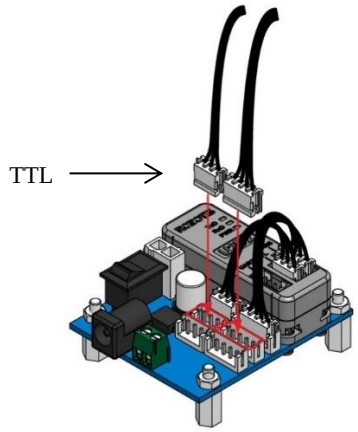

Figura 6. [6] Conexión TTL para comunicación serial con motor

2. Conectar el otro extremo del cable TTL al motor.

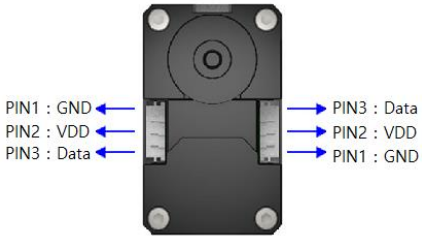

Figura 8. [2] Esquema motor con 3 pines

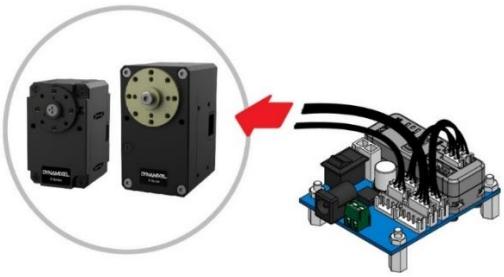

Figura 7. [6] Conexión de U2D2 PHB con motor

3. Conectar el adaptador SMPS al U2D2 PHB.

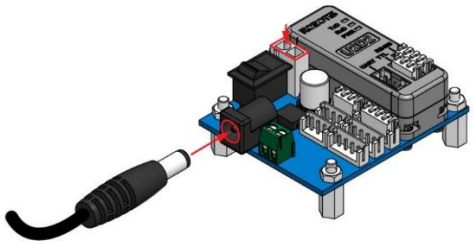

Figura 9. [6] Conexión del adaptador SMPS

4. Al conectar el enchufe del adaptador y encender con el interruptor localizado en el U2D2 PHB, se deberá encender un led rojo parpadeante del motor. Una vez que se corroboró lo último mencionado, volver a colocar el interruptor en la posición apagada.

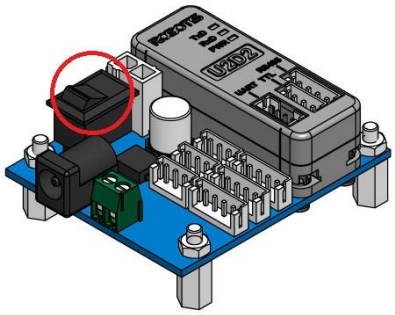

Figura 10. [6] Switch de encendido y apagado.

5. Conectar el cable micro USB al U2D2 y a la computadora.

El esquema completo de las conexiones mencionadas se muestra a continuación.

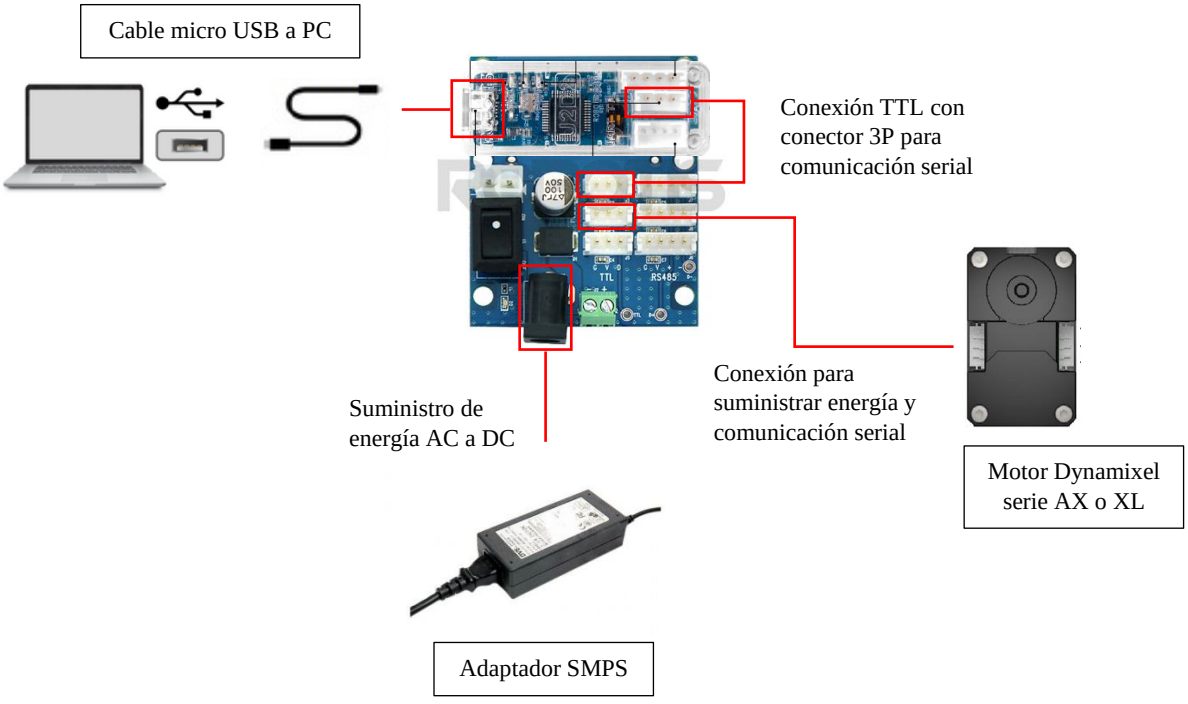

Figura 11. Esquema de conexión completo.

DYNAMIXEL WIZARD

Para comenzar a utilizar el *DXL Wizard*, el software deberá estar instalado y los componentes conectados como se explicó anteriormente (con el motor apagado).

1. Abrir el Dynamixel Wizard 2.0.

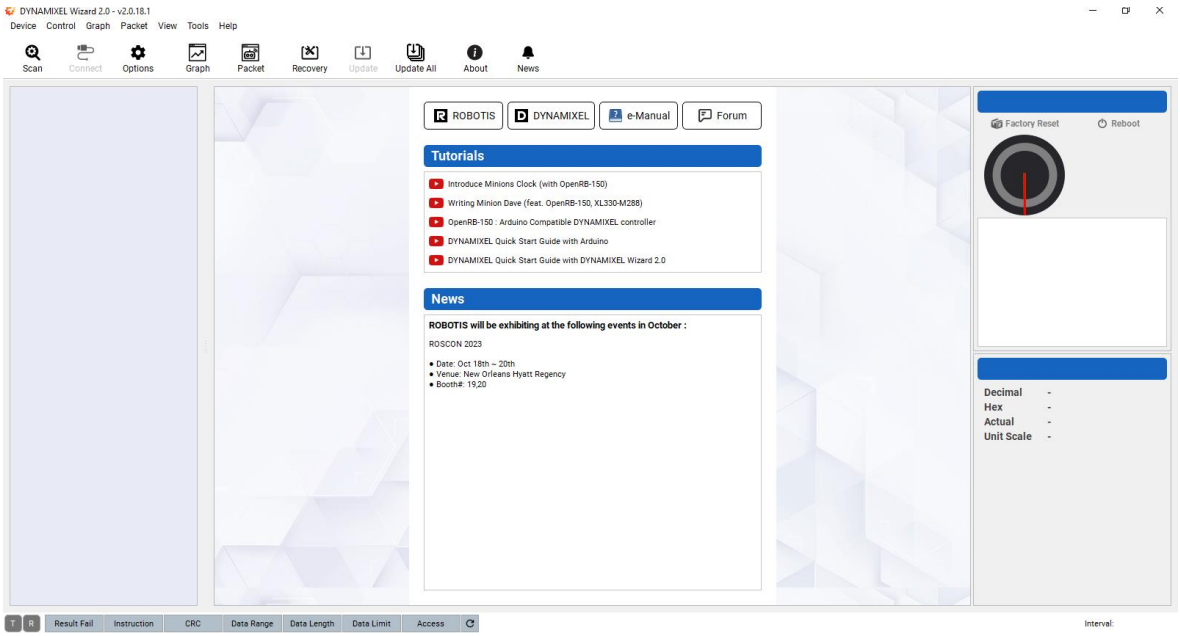

Figura 12. Interfaz de Dynamixel Wizard

2. Seleccionar en el menú superior las opción de Tools < Options, o utilizar el comando F4.
3. En la pantalla emergente, en la sección de “Scan”, seleccionar el puerto de la computadora donde se encuentra conectado el U2D2.

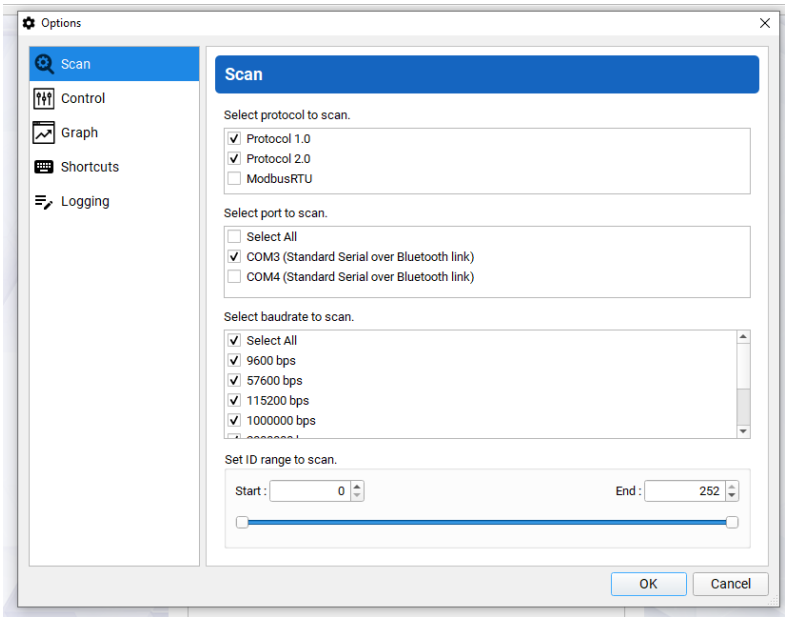

Figura 13. Pantalla de Scan en Dynamixel Wizard

En esta pantalla se podrán elegir algunos parámetros del motor conectado, siendo estos el protocolo, el ID, y los baudios (*baudrate*). En estos valores se recomienda seleccionar todas las opciones con el fin que el programa escanee todas las posibilidades. En caso de que se conozcan estos datos del motor o motores se puede seleccionar los valores específicos.

Al terminar de seleccionar lo necesario, dar clic en “OK”.

4. De regreso a la pantalla de inicio del Wizard, encender el motor con el interruptor del U2D2 PHB y seleccionar la opción de “Scan” en la esquina superior izquierda. Esto iniciará un escaneo, donde, en una nueva pantalla emergente se enlistarán los motores detectados.

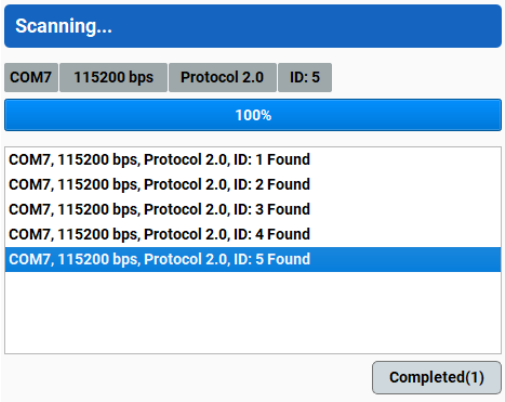

Figura 14. [4] Escaneo del DXL Wizard

Al terminar, el motor o motores detectados aparecerán en la parte izquierda de la interfaz.

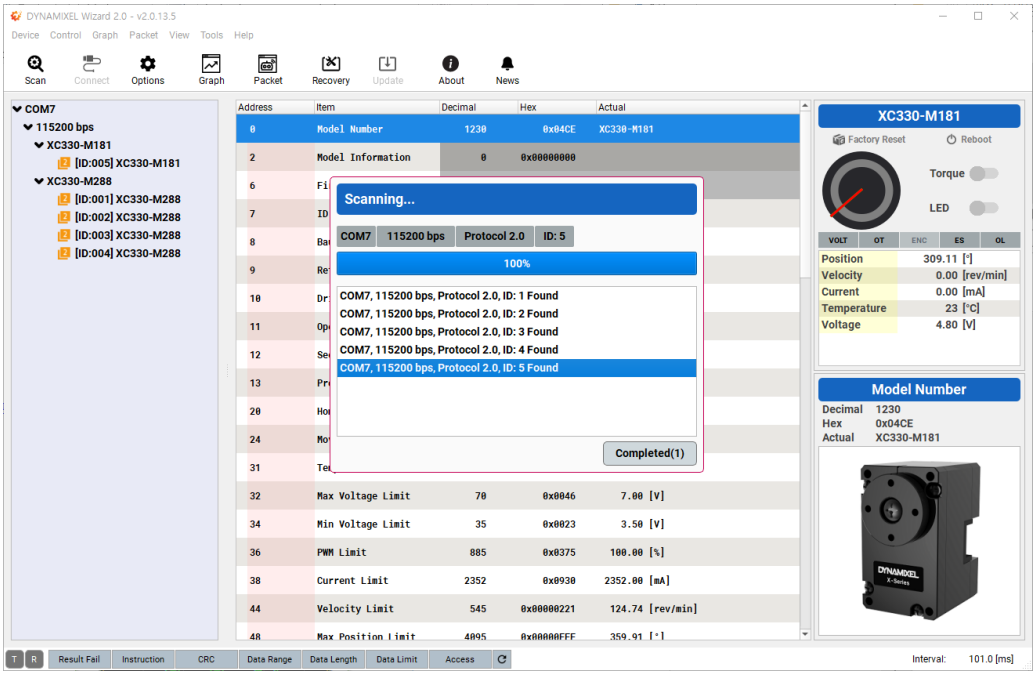

Figura 15 [4]. Pantalla complete del DXL Wizard después del escaneo

Al terminar el escaneo, cerrar la pantalla emergente.

En la pantalla inicial del Wizard, en la parte central aparecerá una lista extensa de todos los parámetros del motor, esta se conoce como “tabla de control” (*control table*).

En esta punto será posible experimentar moviendo el motor, ya sea en función del valor objetivo de posición o velocidad (*Goal Position* o *Goal Velocity*). Para ello, en la parte derecha superior de la interfaz, se debe activar el par (torque) y definir un valor objetivo de posición o velocidad.

Figura 16. [4] Activar el par o “torque”

Posteriormente, en la tabla de control, seleccionar “Goal Position” o “Goal Velocity”. Al hacer eso, en la parte derecha de la interfaz se podrá modificar el valor objetivo.

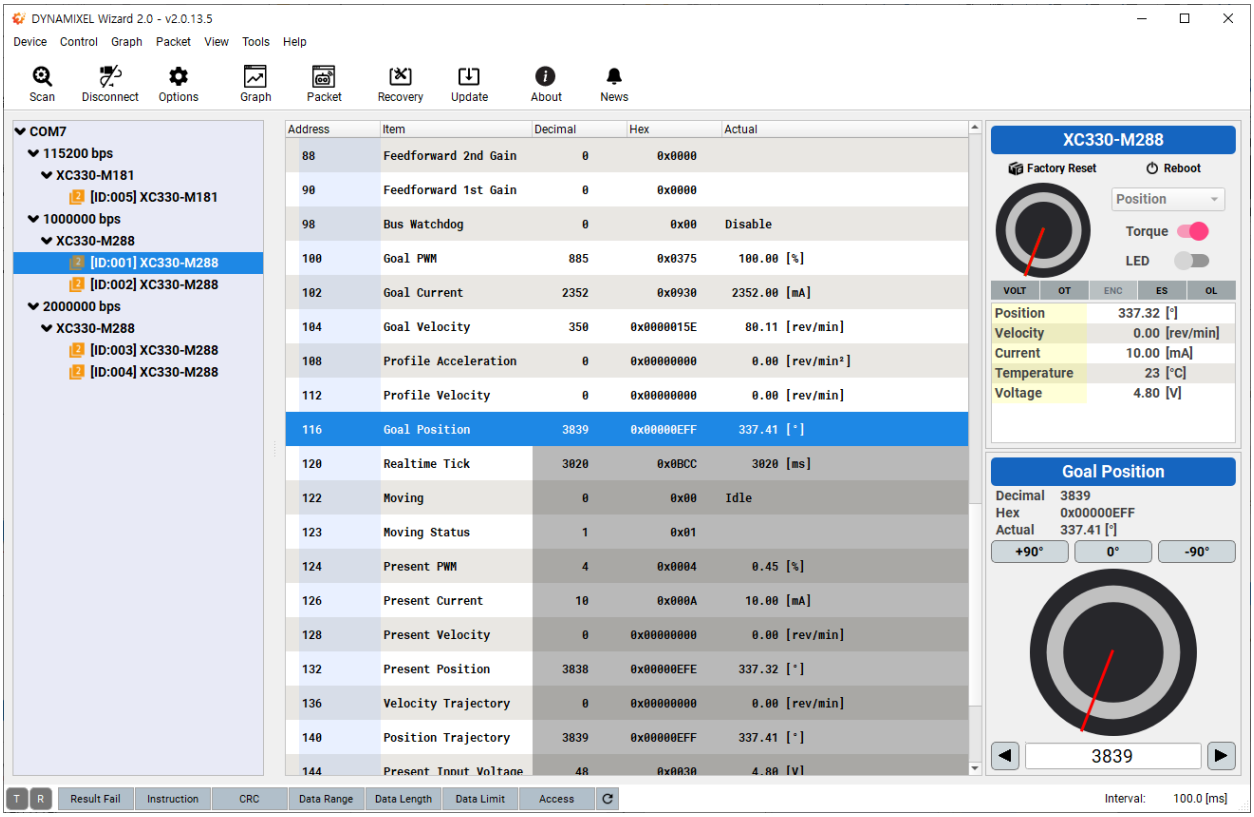

Figura 17. [4] Goal Position

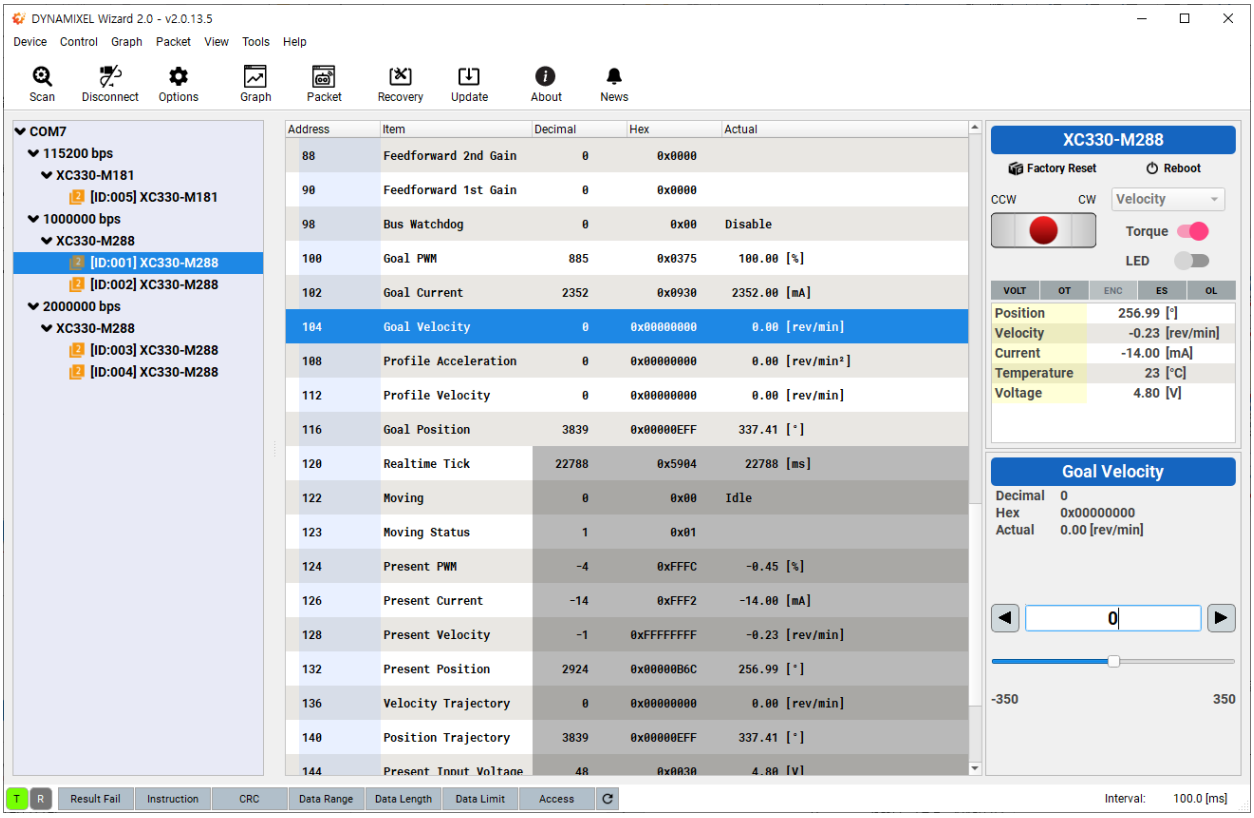

Figura 18. [4] Goal velocity

De manera manual también se podrá mover el motor, posicionando el mouse y arrastrando la aguja de la rueda, o el círculo rojo, como se observa en la siguiente figura.

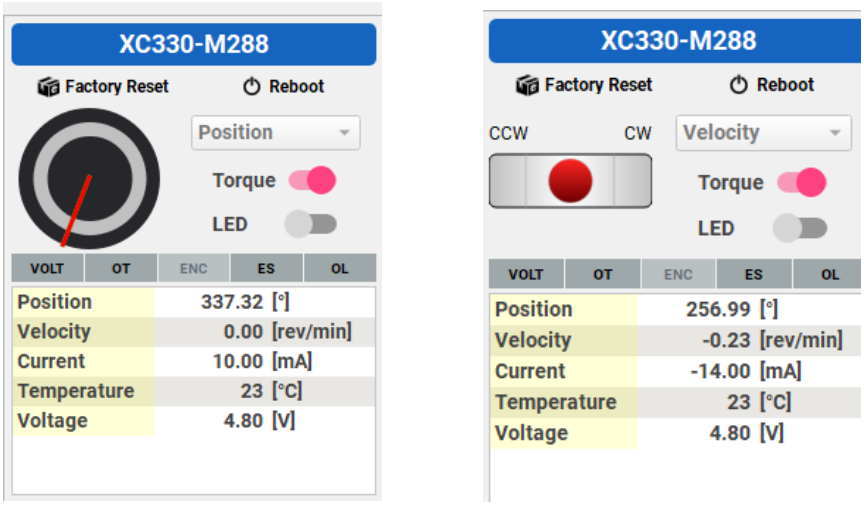

Figura 19. [4] Movimiento manual del motor mediante el DXL Wizard

Para desconectar el motor, en el menú horizontal en la parte superior de la pantalla, seleccionar “Disconnect” y apagar el motor con el interruptor del U2D2.

El procedimiento anterior se puede utilizar para n número de motores.

Ya que en el proyecto se propusieron dos motores, se realizó una tabla comparativa de los parámetros de la tabla de control disponibles y exclusivos para cada motor. En otras palabras, en la siguiente tabla se muestran los parámetros que se encuentran únicamente en un motor. De esta manera se podrán analizar las diferencias entre ambos, enfatizando las ventajas y desventajas.

Tabla 1. Comparación de las tablas de control de los motores

Motor XL430-W250		**Motor AX - 18A**	
ID:0		ID:1	
Adress	Item	Adress	Item
2	Model Information	2	Firmware Version
6	Firmware Version		
10	Drive Mode		
11	Operation Mode		
12	Secondary Shadow ID		
13	Protocol Type		
20	Homing Offset		
24	Moving Threshold		
36	PMW Limit		
44	Velocity Limit		
48	Max Position Limit		
52	Min Position Limit		
60	Startup Configuration		
69	Registered Instruction		
70	Malware Error Status		
76	Velocity I Gain		
78	Velocitu P Gain		
80	Position D Gain		
82	Postion I Gain		
84	Postion P Gain		
		17	Alarm Led
		26	Cw Compliance Margin
		27	Cw Compliance Margin
		28	Cw Compliance Load
		29	Ccw Compliace Slow
		32	Moving Speed
		34	Torque Limit
88	Feedforeard 2nd Gain		
90	Feedforeward 1st Gain		
98	Bud Watchdog		
100	Goal Pwm		
104	Global Velocity		
108	Profile Acceleration		
112	Profile Velocity		
120	Realtime Tick		
123	Moving Status		
124	Present Pwm		
136	Velocity Trajectory		
140	Position Trajectory		
		44	Registered
		47	Lock
		48	Punch
147	Backup Ready		

Como se observa de la Tabla 1, el motor XL430 – W250 – T posee un mayor número de parámetros de interés, que podrían ayudar al objetivo del proyecto. Por lo cual, será este motor el que se implementará en las etapas finales del mismo.

Para más información sobre el significado de los parámetros véanse las fichas técnicas, referencia [1] y [2].

DYNAMIXEL ARDUINO SHIELD

El Dynamixel Shield es un controlador que permite programar el motor a partir de Arduino IDE. Por lo anterior, el DXL Shield se ensambla sobre un Arduino UNO gracias a que posee la misma posición de pines. Adicionalmente, para utilizarse se deberá descargar las librerías correspondientes en Arduino.

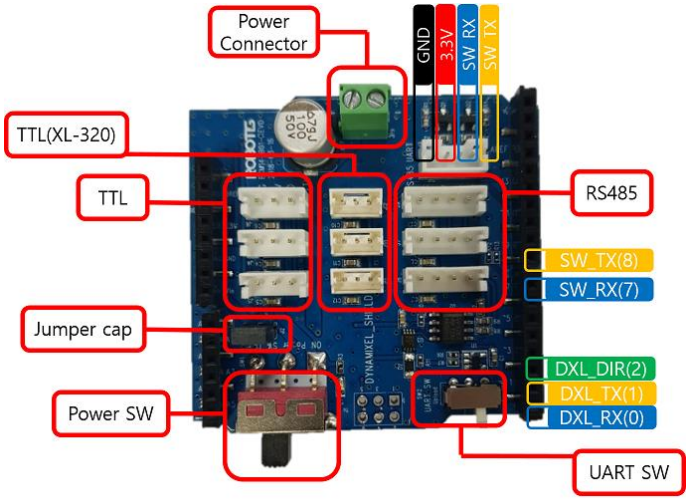

Figura 20 [3]. Esquema de Dynamixel Shield

Es importante notar que el Shield cuenta con un switch de encendido y apagado (Power SW), así como un interruptor para cargar y correr los programas (UART SW), véase la Figura 20.

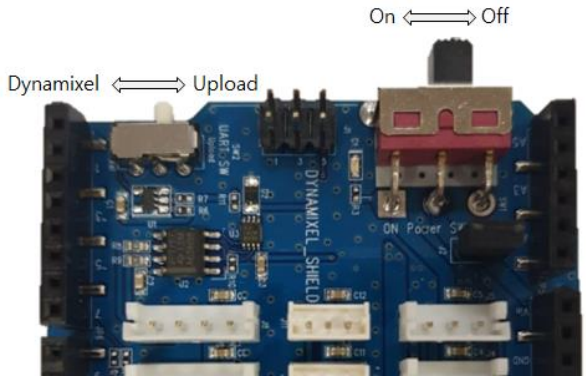

Figura 21 [3]. Posición de los interruptores del DXL Shield

CONEXIONES

El procedimiento de conexión es el mismo para los motores de interés, AX – 18A y XL430 – W250 – T.

Al igual que el Dynamixel Wizard, es necesario conectores de 3 pines para la comunicación TTL al motor, y el adaptador SMPS. De manera adicional, es necesario un cable USB para Arduino, y un cable UART para usar el Monitor Serial de Arduino con el DXL Shield, y observar los valores en tiempo real.

1. Conectar el DXL Shield sobre el Arduino UNO, asegurando que los pines de cada uno concuerden (pin 0 del Shield con pin 0 del Arduino, etc.), y tengan continuidad.

Figura 22 [8]. Arduino con DXL Shield

2. Conectar la comunicación TTL del DXL Shield al motor Dynamixel.
3. Conectar el UART al Shield y a la computadora.
4. Conectar el cable USB del Arduino a la computadora PC.
5. Conectar el potenciómetro rotativo en los pines del Shield, de GND (tierra), Vin (alimentación), y A5 para la señal (Output).
6. Conectar el adaptador SMPS al Arduino UNO, y al enchufe. Al encender el suministro de energía con el interruptor del Shield, deberá encenderse un LED del motor. Cuando se verifique lo anterior, volver a apagar el motor.

Las conexiones completas se muestran en el siguiente esquema.

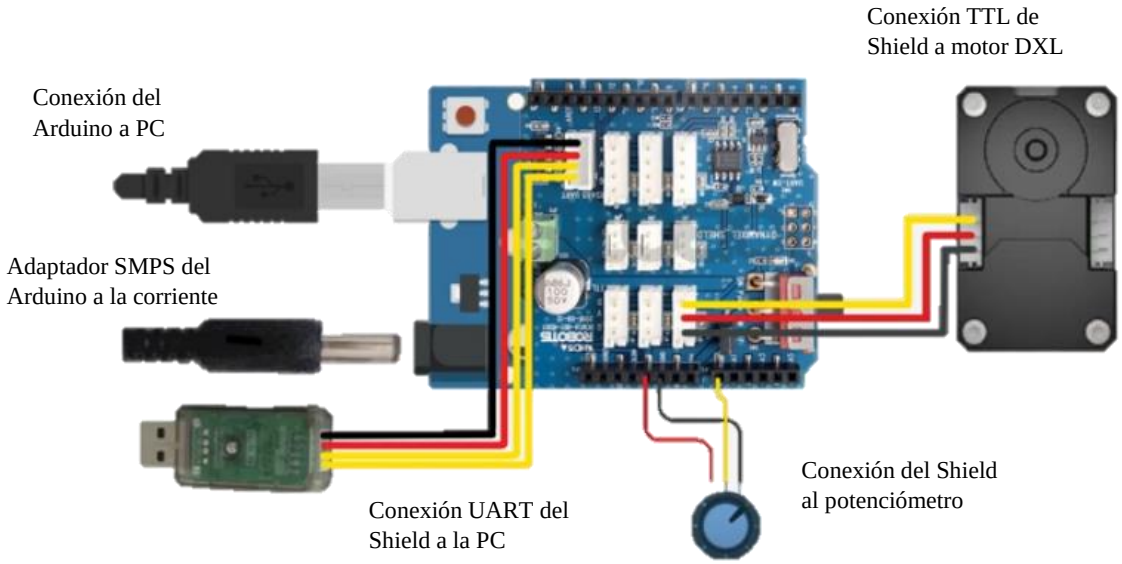

Figura 23 [9]. Esquema de conexiones del Shield

DESCARGA DE LIBRERÍA

Para descargar las librerías necesarias, en el Arduino IDE, en el menú superior, ir a Tools < Manage Libraries. En la pantalla emergente, buscar e instalar “Dynamixel 2 Arduino” y “Dynamixel Shield”.

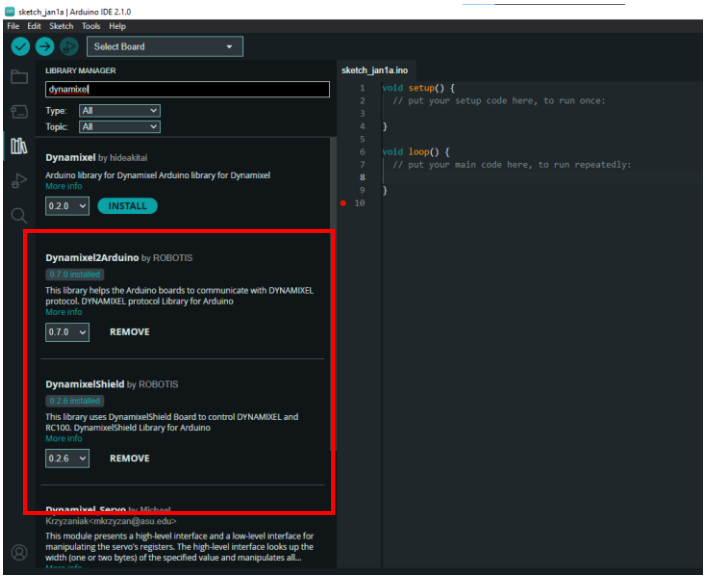

Figura 24. Descarga de librerías

En ellas se podrán encontrar todos los ejemplos predeterminados para los motores de la marca. Para acceder a ellos, se deberá entrar en el menú superior del IDE, File < Examples < Dynamixel Shield < Basic.

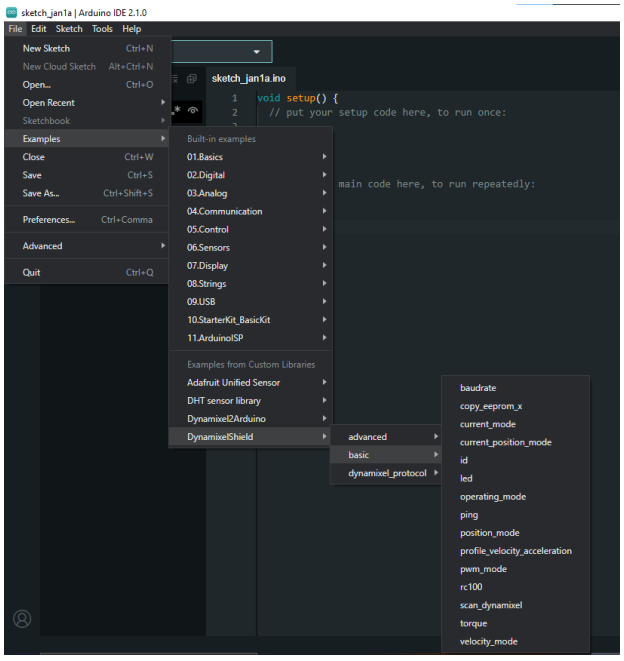

Figura 25. Ejemplos predeterminados

Para los códigos del proyecto se modificaron dichos ejemplos de control por velocidad y posición de las librerías (*Velocity Mode*, *Position Mode*).

CONTROL DE LOS MOTORES DXL

Si bien el objetivo del control para el proyecto es que al modificar la posición de un potenciómetro se modifique la posición del motor, por facilidad, se inició el desarrollo de los códigos por un control por velocidad.

El código de los ejemplos se divide en 4 secciones principales. En la primera sección se referencian las librerías y una serie de condicionales "if" donde se indica el tipo de Arduino que se está utilizando, y los puertos.

```
#include <DynamixelShield.h> // Referencia la libreria de Arduino

#if defined(ARDUINO_AVR_UNO) || defined(ARDUINO_AVR_MEGA2560)
  #include <SoftwareSerial.h>
  SoftwareSerial soft_serial(7, 8); // DYNAMIXELShield UART RX/TX
  #define DEBUG_SERIAL soft_serial
#elif defined(ARDUINO_SAM_DUE) || defined(ARDUINO_SAM_ZERO)
  #define DEBUG_SERIAL SerialUSB // Para poder obtener datos en tiempo real
#else
  #define DEBUG_SERIAL Serial
#endif
```

Figura 26. Ejemplo de velocidad, primera sección

En la segunda sección se indica el número de ID y de protocolo del motor en específico que se utilizará. Por ello, estos se deberá cambiar dependiendo si está utilizando el AX – 18A o XL430 – W250 – T.

```
// Descripcion del Dynamixel utilizado
const uint8_t DXL_ID = 1; //Numero de ID
const float DXL_PROTOCOL_VERSION = 1.0; //Protocolo

DynamixelShield dxl;

//This namespace is required to use Control table item names
using namespace ControlTableItem;
```

Figura 27. Ejemplo de velocidad para AX – 18 A, segunda sección

La siguiente sección del código corresponde al set up, en ella se deberá colocar el número de baudios del motor, el cual también cambiará dependiendo el motor.

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);

  // For Uno, Nano, Mini, and Mega, use UART port of DYNAMIXEL Shield to debug.
  DEBUG_SERIAL.begin(115200);

  // Set Port baudrate. This has to match with DYNAMIXEL baudrate.
  dxl.begin(1000000); //Verificar en la Tabla de Control del motor en el Wizard

  // Set Port Protocol Version. This has to match with DYNAMIXEL protocol version.
  dxl.setPortProtocolVersion(DXL_PROTOCOL_VERSION);

  // Get DYNAMIXEL information
  dxl.ping(DXL_ID);

  // Turn off torque when configuring items in EEPROM area
  dxl.torqueOff(DXL_ID);
  dxl.setOperatingMode(DXL_ID, OP_VELOCITY);
  dxl.torqueOn(DXL_ID);

}
```

Figura 28. Ejemplo de velocidad para AX – 18 A, tercera sección

El número del ID, y los baudios pueden consultarse en la tabla de control del Dynamixel Wizard. El protocolo se puede consultar en las fichas técnicas de los motores, siendo protocolo 1.0 para el AX – 18A, y protocolo 2.0 para el XL430 – W250 – T. A continuación, se presenta dicha tabla del motor AX – 18A, señalando los datos necesarios.

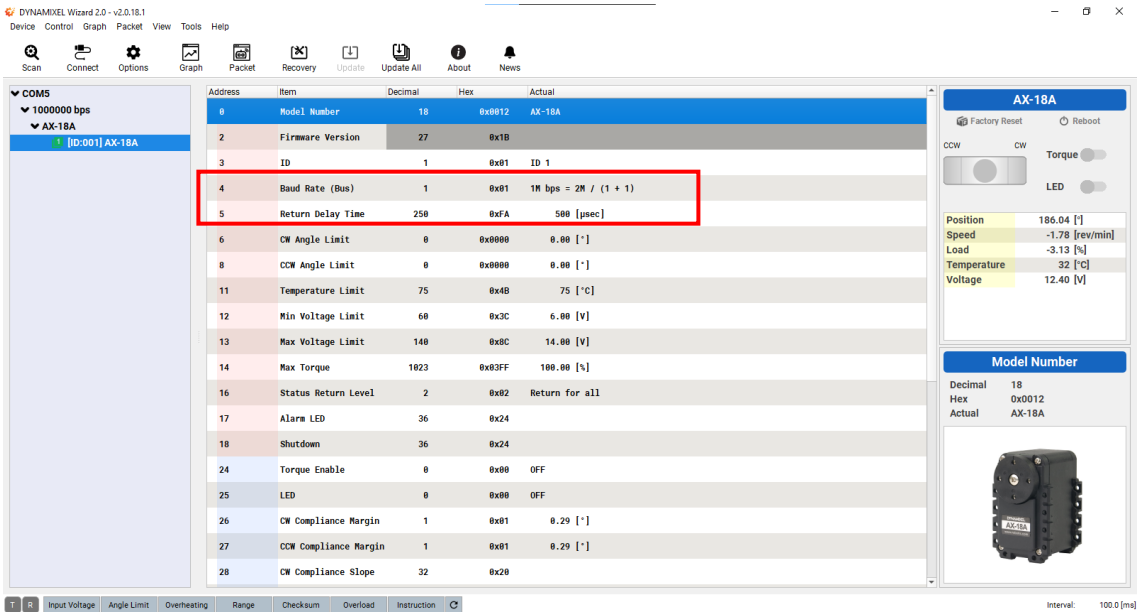

Figura 29. Tabla de control del motor AX – 18 A

La última sección consiste en el control del motor. En los ejemplos de velocidad, hay tres maneras en que se puede controlar el motor. Se puede definir un valor “Raw Unit” que representa la velocidad máxima mínima con un rango de – 255 a +255 para los motores de serie X, – 1023 a +1023 para todos los motores de la marca. Asimismo, en el control por velocidad se puede controlar por RPM o por porcentaje de velocidad; En el control por posición, se puede utilizar un valor “Raw Unit”, o en grados. Según convenga se pueden utilizar cualquiera de los métodos.

```
void loop() {
  // put your main code here, to run repeatedly:

  // Please refer to e-Manual(http://emanual.robotis.com) for available range of value.
  // Set Goal Velocity using RAW unit
  dxl.setGoalVelocity(DXL_ID, 200);
  delay(1000);
  // Print present velocity
  DEBUG_SERIAL.print("Present Velocity(raw) : ");
  DEBUG_SERIAL.println(dxl.getPresentVelocity(DXL_ID));
  delay(1000);

  // Set Goal Velocity using RPM
  dxl.setGoalVelocity(DXL_ID, 25.8, UNIT_RPM);
  delay(1000);
  DEBUG_SERIAL.print("Present Velocity(rpm) : ");
  DEBUG_SERIAL.println(dxl.getPresentVelocity(DXL_ID, UNIT_RPM));
  delay(1000);

  // Set Goal Velocity using percentage (-100.0 [%] ~ 100.0 [%])
  dxl.setGoalVelocity(DXL_ID, -10.2, UNIT_PERCENT);
  delay(1000);
  DEBUG_SERIAL.print("Present Velocity(ratio) : ");
  DEBUG_SERIAL.println(dxl.getPresentVelocity(DXL_ID, UNIT_PERCENT));
  delay(1000);
}
```

Figura 30. Ejemplo de velocidad, cuarta sección

Para correr el código (ya sea por velocidad o posición), se deben seguir el siguiente procedimiento:

1. Encender el motor con el switch del Shield (véase Figura 21).
2. Posicionar el switch UART en “Upload” (Figura 21).
3. Desde el IDE, seleccionar el tipo de Arduino usado, así como el puerto COM donde éste esté conectado.
4. Subir el programa desde el IDE.
5. Cuando este haya compilado y terminado de subirse, cambiar el switch UART a la posición “Dynamixel” (Figura 21).
6. Presionar el botón de Reset del Arduino.
7. Cambiar de puerto COM al puerto donde está conectado el UART. Lo anterior desde el IDE, en el menú Tools < Port. Si se mantiene el puerto COM a donde está conectado el Arduino después de subir y correr el programa, no se podrán observar los valores en tiempo real.
8. Abrir el Monitor Serial del IDE.

El procedimiento anterior es el mismo independientemente del motor siempre y cuando se cambie el número de ID, protocolo, y baudios.

CONTROL POR VELOCIDAD

Para el control por velocidad con un potenciómetro rotativo, se utilizó el ejemplo de Dynamixel Shield de Velocity Mode con el método de Raw Unit. Se añadió una variable denominada “value”, la cual será directamente dada por el voltaje del potenciómetro. En esa línea, deberá especificarse el pin analógico donde se colocó el pin Output del potenciómetro. En el caso de las pruebas, este fue el pin A5.

Además, en un nueva variable denominada “val” se realizó un mapeo del valor del potenciómetro (con un rango de 0 a 1023) a los Raw Unit del motor (en un rango de 0 a 255). Finalmente, se escribieron líneas para imprimir los valores en el Monitor Serial. Si se deseara utilizar los grados en lugar de los valores “Raw Unit”, se deberán cambiar los valores correspondientes, verificando los valores de RPM máximos posibles por el motor.

El código modificado completo para el motor AX – 18A se muestra a continuación.

```
/*******************************************************************************
* Copyright 2016 ROBOTIS CO., LTD.
*
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
*
*     http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
*******************************************************************************/

#include <DynamixelShield.h> // Referencia la libreria de Arduino

#if defined(ARDUINO_AVR_UNO) || defined(ARDUINO_AVR_MEGA2560)
  #include <SoftwareSerial.h>
  SoftwareSerial soft_serial(7, 8); // DYNAMIXELShield UART RX/TX
  #define DEBUG_SERIAL soft_serial
#elif defined(ARDUINO_SAM_DUE) || defined(ARDUINO_SAM_ZERO)
  #define DEBUG_SERIAL SerialUSB // Para poder obtener datos en tiempo real
#else
  #define DEBUG_SERIAL Serial
#endif

// Descripción del Dynamixel utilizado
const uint8_t DXL_ID = 1; //Numero de ID
const float DXL_PROTOCOL_VERSION = 1.0; //Protocolo

DynamixelShield dxl;

//This namespace is required to use Control table item names
using namespace ControlTableItem;
```

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);

  // For Uno, Nano, Mini, and Mega, use UART port of DYNAMIXEL Shield to debug.
  DEBUG_SERIAL.begin(115200);

  // Set Port baudrate. This has to match with DYNAMIXEL baudrate.
  dxl.begin(1000000); //Verificar en la Tabla de Control del motor en el Wizard

  // Set Port Protocol Version. This has to match with DYNAMIXEL protocol
version.
  dxl.setPortProtocolVersion(DXL_PROTOCOL_VERSION);

  // Get DYNAMIXEL information
  dxl.ping(DXL_ID);

  // Turn off torque when configuring items in EEPROM area
  dxl.torqueOff(DXL_ID);
  dxl.setOperatingMode(DXL_ID, OP_VELOCITY);
  dxl.torqueOn(DXL_ID);

}

void loop() {
  // put your main code here, to run repeatedly:

  // Please refer to e-Manual(http://emanual.robotis.com) for available range of
value.
  // Set Goal Velocity using RAW unit
  int value=analogRead(A5); //Lectura del potenciómetro, se indica el pin donde
se conecto
  int val=map(value,0,1023,0,255); // Mapeo del voltaje a valores RAW Unit
  dxl.setGoalVelocity(DXL_ID,val); // El valor objetivo será determinado por el
potenciometro
  delay(1000);

  // Print present velocity
  DEBUG_SERIAL.print("Present Velocity(raw) : ");
  DEBUG_SERIAL.println(dxl.getPresentVelocity(DXL_ID));
  delay(1000);

}
```

Para correr el código, seguir el procedimiento descrito en la sección Control de los motores DXL.

CONTROL POR POSICIÓN

El control por posición, de manera similar, se partió del ejemplo predeterminado de Position Mode, añadiendo las variables “val” y “value” mencionadas en el control por velocidad. La diferencia es que en el mapeo de la variable “value”, se utilizó un rango de 0 a 512. Lo anterior fue con el fin de abarcar todo el rango de – 256 a 256 posibles del motor, en números positivos, y con ello, evitar valores negativas.

El código modificado completo para el motor AX – 18A se muestra a continuación.

```
/*******************************************************************************
  Copyright 2016 ROBOTIS CO., LTD.

  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.
*******************************************************************************/

#include <DynamixelShield.h> // Referencia la libreria de Arduino

#if defined(ARDUINO_AVR_UNO) || defined(ARDUINO_AVR_MEGA2560)
#include <SoftwareSerial.h>
SoftwareSerial soft_serial(7, 8); // DYNAMIXELShield UART RX/TX
#define DEBUG_SERIAL soft_serial
#elif defined(ARDUINO_SAM_DUE) || defined(ARDUINO_SAM_ZERO)
#define DEBUG_SERIAL SerialUSB  // Para poder obtener datos en tiempo real
#else
#define DEBUG_SERIAL Serial
#endif

// Descripción del Dynamixel utilizado
const uint8_t DXL_ID = 1; //Numero de ID
const float DXL_PROTOCOL_VERSION = 1.0; //Protocolo
DynamixelShield dxl;

//This namespace is required to use Control table item names
using namespace ControlTableItem;
```

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

  // For Uno, Nano, Mini, and Mega, use UART port of DYNAMIXEL Shield to debug.
  DEBUG_SERIAL.begin(115200);

  // Set Port baudrate. This has to match with DYNAMIXEL baudrate.
  dxl.begin(1000000); //Verificar en la Tabla de Control del motor en el Wizard

  // Set Port Protocol Version. This has to match with DYNAMIXEL protocol
version.
  dxl.setPortProtocolVersion(DXL_PROTOCOL_VERSION);

  // Get DYNAMIXEL information
  dxl.ping(DXL_ID);

  // Turn off torque when configuring items in EEPROM area
  dxl.torqueOff(DXL_ID);
  dxl.setOperatingMode(DXL_ID, OP_POSITION);
  dxl.torqueOn(DXL_ID);
}

void loop() {
  // put your main code here, to run repeatedly:

  // Please refer to e-
Manual(http://emanual.robotis.com/docs/en/parts/interface/dynamixel_shield/) for
available range of value.
  // Set Goal Position in RAW value
  int value = analogRead(A5); //Lectura del potenciómetro, se indica el pin donde
se conecto
  int val = map(value, 0, 1023, 0, 512); // Mapeo del voltaje a valores RAW Unit
  dxl.setGoalPosition(DXL_ID, value); // El valor objetivo será determinado por
el potenciometro
  delay(1000);

  // Print present position in raw value
  DEBUG_SERIAL.print("Present Position(raw) : ");
  DEBUG_SERIAL.println(dxl.getPresentPosition(DXL_ID));
  delay(1000);
}
```

Igualmente, si se desea utilizar en lugar de “Raw Unit” se deberá cambiar los valores en el rango correspondiente según indica el manual.

COMPATIBILIDAD DE *FRAMES* CON EL MOTOR XL430-W250-T

Para el prototipo físico se partió de un *frame* comercial dada por la marca Robotis. Si bien las pruebas de control fueron hechas con ambos motores, la manufactura del prototipo físico se enfocará únicamente en el motor XL430 – W250 – T, al ser este el que se implementará el diseño final. Así, hay 6 diferentes partes (*frames* con rodamientos) compatibles con el motor XL430-W250-T (referencia [10]). En la ficha técnica del motor ([2]) se pueden observar los montajes con los diferentes *frames*.

1. **Modelo**: FR11-H101K Set

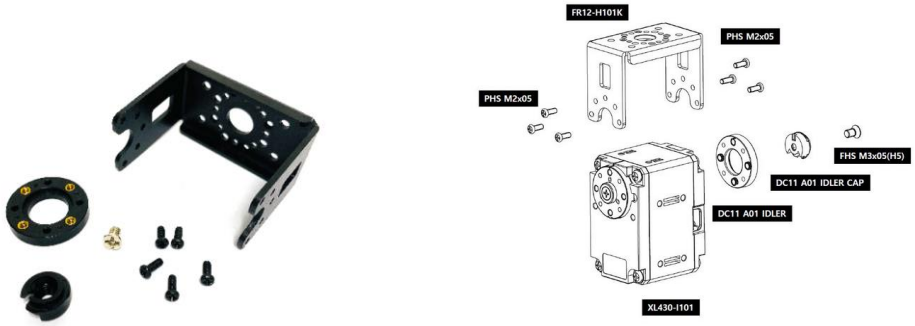

Figura 31 [11]. Pieza FR11-H101K Set y ensamble

Incluye:

- *Bracket* o *frame* (FR12-H101K)
- Rodamientos (HN11-I101, referencia [11.1]). Se conecta al lado opuesto al eje de salida del motor. Puede utilizarse para montar el *frame*.
- Tornillos (5 M2, 1 M3)

2. **Modelo**: FR12-H104K Set

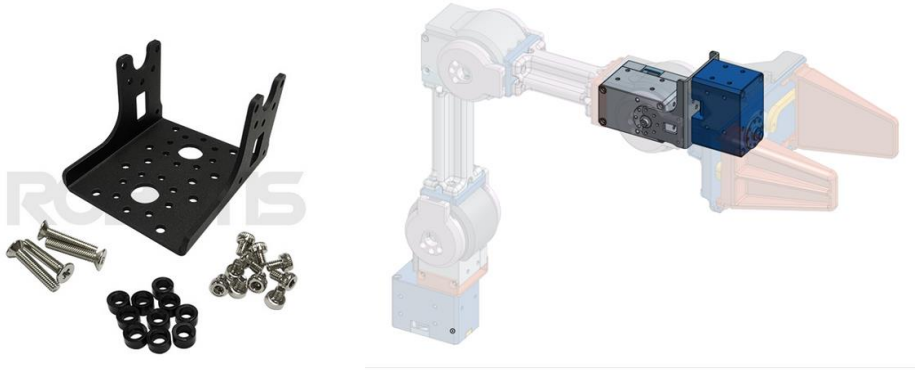

Figura 32 [12]. Pieza FR12-H104K Set y aplicación

Características:

- *Frame* extendido utilizado para *Open Manipulator* (colocar dos motores).
- Se puede extender en el eje de bisagra y utilizar en combinación con un eje de intersección
- Se debe comprar por aparte un juego de rodamiento: HN11-I101.

Incluye:

- *Bracket* o *frame*
- Tornillos y anillos.

3. **Modelo**: FR12-G101GM Set

Figura 33 [13]. Pieza FR12-G101GM Set

Características:

- *Gripper Frame*
- Se debe comprar por aparte un juego de rodamiento: HN11-I101.

Incluye:

- 2 *frames* (FR12-E170GM y FR12-E171GM)
- Tornillos y anillos.

4. **Modelo**: FR12-H103GM Set

Figura 34 [14]. Pieza FR12-H103GM Set

Características:

- Se debe comprar por aparte un juego de rodamiento: HN11-I101.

Incluye:

- *Bracket* o *frame*
- Tornillos y anillos.

5. **Modelo**: FR12-S101K Set

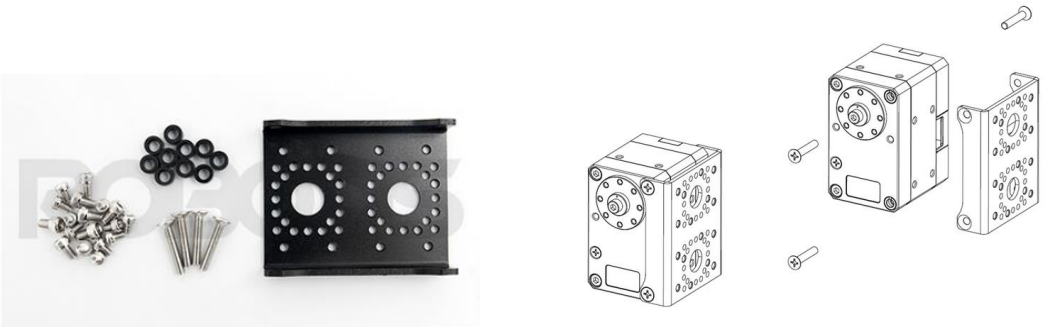

Figura 35 [15]. Pieza FR12-S101K Set y ensamble

Características:

- Se puede utilizar para montarlo en la parte lateral del actuador.

Incluye:

- *Bracket* o *frame*
- Tornillos y anillos.

Precio: $ 28.60

6. **Modelo**: FR12-S102K Set

Figura 36 [16]. Pieza FR12-S102K Set y ensamble

Características:

- Se puede utilizar para montarlo en la parte lateral del actuador.

Incluye:

- *Bracket* o *frame*
- Tornillos y anillos.

FRAME MODIFICADO

En el diseño del proyecto, el motor Dynamixel estará posicionado en la base, y unido a él se encontrará un eslabón. Esto con el fin que este tenga un movimiento rotativo. Por ello, de los *frames* investigados, aquel que se adecúa a esa disposición es el modelo 1, FR12-H101K.

Para optimizar el ensamble del diseño final, se modificó el CAD del *frame* (referencia [19]) y se añadió el eslabón sobre él. De esta manera, se omite el ensamble del eslabón con el *frame*. Asimismo, se aumentó el espesor de las paredes del eslabón, específicamente para disminuir el riesgo de fractura en la zona de unión con el motor. Lo anterior es debido a que el espesor determinado en el CAD del frame fabricado por Robotis es de metal, mientras que el modificado e impreso seria de PLA, un material significativamente mas frágil.

Figura 37. Frame modificado

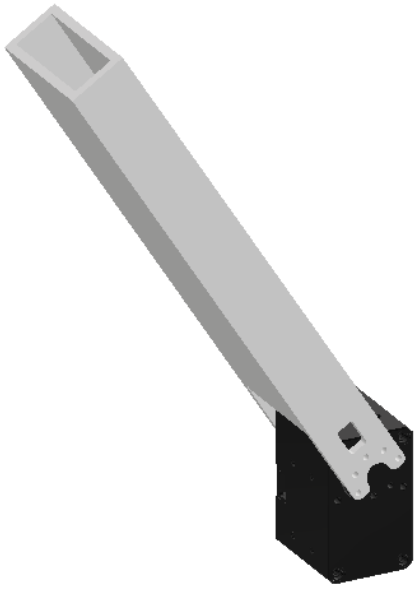

Figura 38. Ensamble de frame modificado

CONCLUSIONES

En los avances presentados en este reporte se pudo desarrollar el procedimiento para realizar el control por posición del motor DXL en función del movimiento de un potenciómetro. En desarrollos futuros, en lugar del potenciómetro rotativo, se deberá cambiar a un potenciómetro lineal con el fin que el eslabón sea adaptable. Respecto al prototipo físico, se deberá añadir el eslabón adaptable con el potenciómetro lineal para el proyecto final. Por lo anterior, se considera que a lo largo del semestre se pudo avanzar con las metas del proyecto, enfocándose en el objetivo principal.

Finalmente, en el repositorio de Github se incluye una carpeta con el CAD y archivo .stl del *frame* modificado, el presente reporte, y los códigos de velocidad y posición.

REFERENCIAS

[1]

Y. Name, "AX – 18A," ROBOTIS e-Manual. https://emanual.robotis.com/docs/en/dxl/ax/ax-18a/

[2]

Y. Name, "XL430 – W250 – T," ROBOTIS e-Manual. https://emanual.robotis.com/docs/en/dxl/x/xl430-w250/

[3]

Y. Name, "DYNAMIXEL Shield," ROBOTIS e-Manual. https://emanual.robotis.com/docs/en/parts/interface/dynamixel_shield/ (accessed Jan. 02, 2024).

[4]

Y. Name, "DYNAMIXEL Wizard 2.0," ROBOTIS e-Manual. https://emanual.robotis.com/docs/en/software/dynamixel/dynamixel_wizard2/

[5]

Y. Name, "U2D2," ROBOTIS e-Manual. https://emanual.robotis.com/docs/en/parts/interface/u2d2/

[6]

Y. Name, "U2D2 Power Hub," ROBOTIS e-Manual. https://emanual.robotis.com/docs/en/parts/interface/u2d2_power_hub/ (accessed Jan. 02, 2024).

[7]

Y. Name, "DYNAMIXEL Quick Start," ROBOTIS e-Manual. https://emanual.robotis.com/docs/en/dxl/dxl-quick-start-insert/#dynamixel-starter-set (accessed Jan. 02, 2024).

[8]

"DYNAMIXEL Shield – Digiware Store,", digiwarestore.com, 2024. https://external-content.duckduckgo.com/iu/?u=https%3A%2F%2Ftse4.mm.bing.net%2Fth%3Fid%3DOIP.yFYzf8QDY1kNEwLrmlvLcAHaHa%26pid%3DApi&f=1&ipt=964b4eb7ca7cfa0dd91cf64bd3df898742a3de74352192b3cadde6bc24583f55&ipo=images (accessed Jan. 02, 2024).

[9]

ROBOTIS Costumer Support, "How to Use DYNAMIXEL with Basic Potentiometer + Arduino Uno || Servo Control using Potentiometer," www.youtube.com, Dec. 22, 2020. https://www.youtube.com/watch?v=ZvZtG4ILW1A&list=PLEf1s0tzVSnTLiMYePCMOYx7TpE3wzw8S&index=3 (accessed Jan. 02, 2024).

[10]

"ROBOTIS Shop – Compatibility," ROBOTIS. https://en.robotis.com/service/compatibility_table.php?cate=dx (accessed Nov. 13, 2023).

[11]

“FR11-H101K Set,” ROBOTIS. https://www.robotis.us/fr11-h101k-set/ (accessed Nov. 13, 2023).

[11.1]

“HN11-I101 Set,” ROBOTIS. https://www.robotis.us/hn11-i101-set/

[12]
“FR12-H104K Set,” *ROBOTIS*. https://www.robotis.us/fr12-h104k-set/ (accessed Nov. 13, 2023).

[13]

“FR12-G101GM Set,” *ROBOTIS*. https://www.robotis.us/fr12-g101gm-set/(accessed Nov. 13, 2023).

[14]

“FR12-H103GM Set,” *ROBOTIS*. https://www.robotis.us/fr12-h103gm-set/ (accessed Nov. 13, 2023).

[15]

“FR12-S101K Set,” *ROBOTIS*. https://www.robotis.us/fr12-s101k-set/ (accessed Nov. 13, 2023).

[16]

“FR12-S102K Set,” *ROBOTIS*. https://www.robotis.us/fr12-s102k-set/ (accessed Nov. 13, 2023).

[17]

“ROBOTIS Shop – Download,” ROBOTIS.
https://en.robotis.com/service/downloadpage.php?ca_id=7030 (accessed Jan. 02, 2024).