# Configuración Inicial de los motores Dynamixel.

Para poder garantizar el correcto funcionamiento de los motores, es necesario generar una configuración inicial para definir parámetros base que identifican a cada motor. Esto es primordial, ya que esta configuración es requerida al utilizar cualquier plataforma de desarrollo que requiera el uso de estos servomotores.

Como primer punto, es necesario especificar la conexión de todos los componentes del motor. Como se puede apreciar en la Fig. 1, la conexión consiste en conectar el U2D2 a la computadora vía USB, y los Dynamixel al puerto correspondiente, en nuestro caso, utilizamos los motores AX-12a que consisten en 3 pines, ya que trabajan con comunicación TTL; a la par, hay que conectar la fuente de alimentación.

Imagen que contiene Escala de tiempo

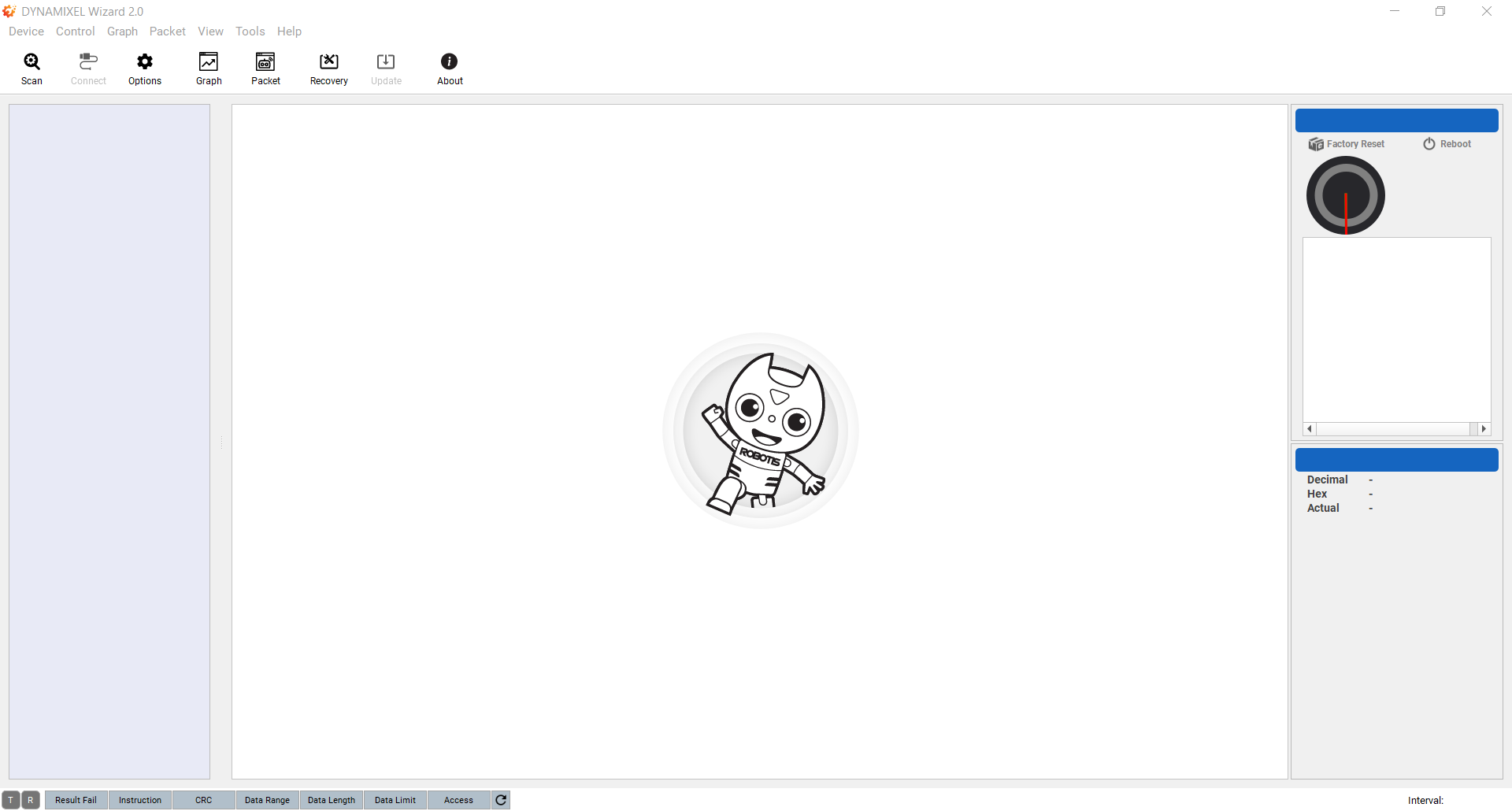
Descripción generada automáticamente

*Fig. 1 Diagrama de Conexión para motores Dynamixel.*

Al tener verificado la comunicación de los motores, el siguiente paso es configurarlos. Para eso, es necesario descargar un software de configuración diseñado para estos motores. El software utilizado se llama Dynamixel Wizard 2.0, Fig. 2, el cual tiene la ventaja de ser multiplataforma (Windows, Linux, Mac) y permite realizar la configuración necesaria.

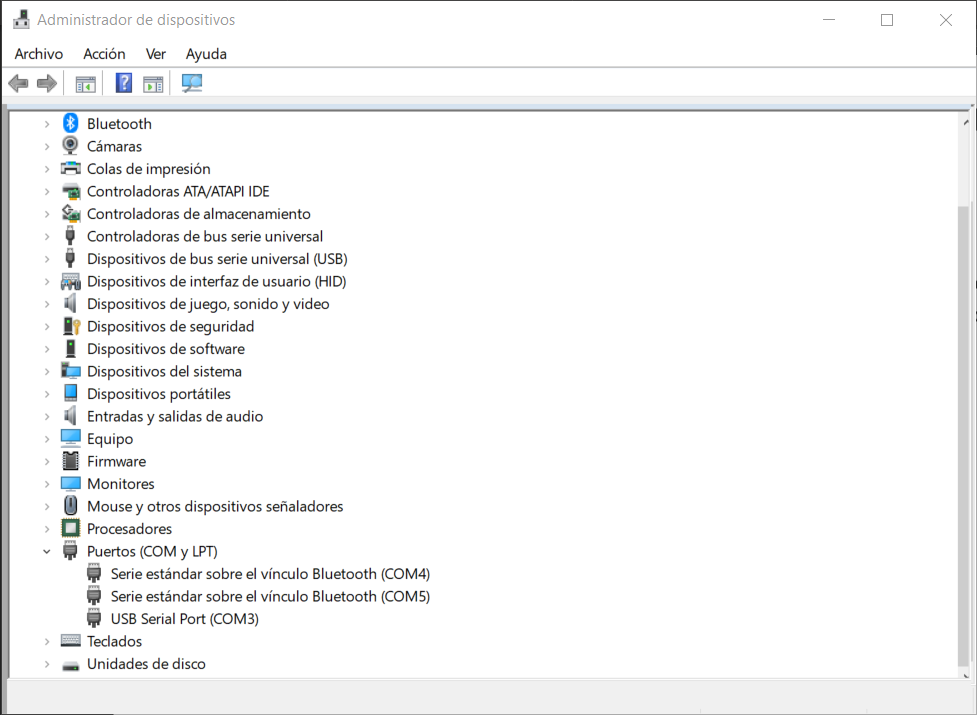
Para descargar este software, utilizamos el siguiente enlace y seleccionamos la versión correspondiente a nuestro sistema operativo.

[DYNAMIXEL Wizard 2.0 (robotis.com)](https://emanual.robotis.com/docs/en/software/dynamixel/dynamixel_wizard2/)



*Fig. 2 Ventana Principal de Dynamixel Wizard 2.0.*

Para iniciar la configuración, primero hay que conocer el puerto al que está asignado el U2D2. En Windows, podemos ingresar al **Administrador de Dispositivos** para encontrar el puerto COM al que está vinculado el U2D2, el cual podemos encontrar dentro de la sección **Puertos (COM y LPT),** como se muestra en la Fig. 3.

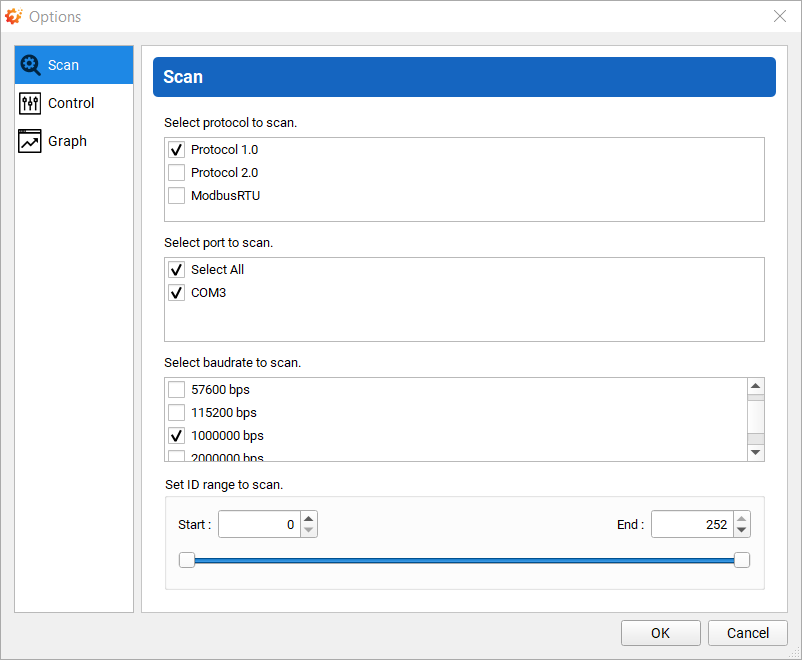


*Fig. 3 Ventana del Administrador de Dispositivos indicado el Puerto del U2D2.*

Conociendo el puerto asignado, que en nuestro caso es el “COM3”, regresamos al programa y seleccionamos ***Options/Scan*** Dentro de esta ventana, podemos escoger el Protocolo de Trabajo de los Motores, así como la Velocidad de Transmision de Datos (Baudrate) y el Identificador (ID) que deseamos buscar. Cabe resaltar que cuando se adquieren motores nuevos, o de los cuales no se conoce su configuración, es recomendable seleccionar todos los valores que proporciona esta ventana, así mismo, se recomienda hacer este procedimiento con un motor a la vez para prevenir algún fallo que se pueda generar a los motores al momento de asignar su configuración.

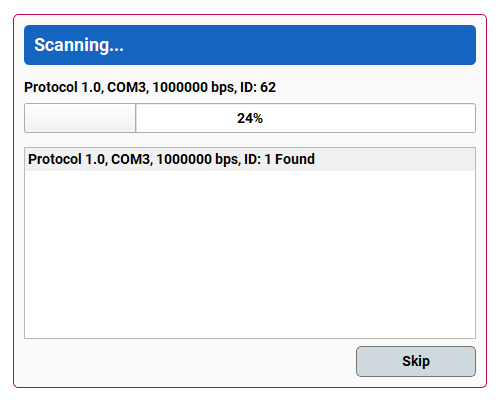
Para nuestro caso en particular, la Fig. 4 muestra la selección dentro de la ventana de opciones. La configuración recomendada es la siguiente:

* Puerto COM3
* Protocolo 1.0
* Baudrate 1000000 bps



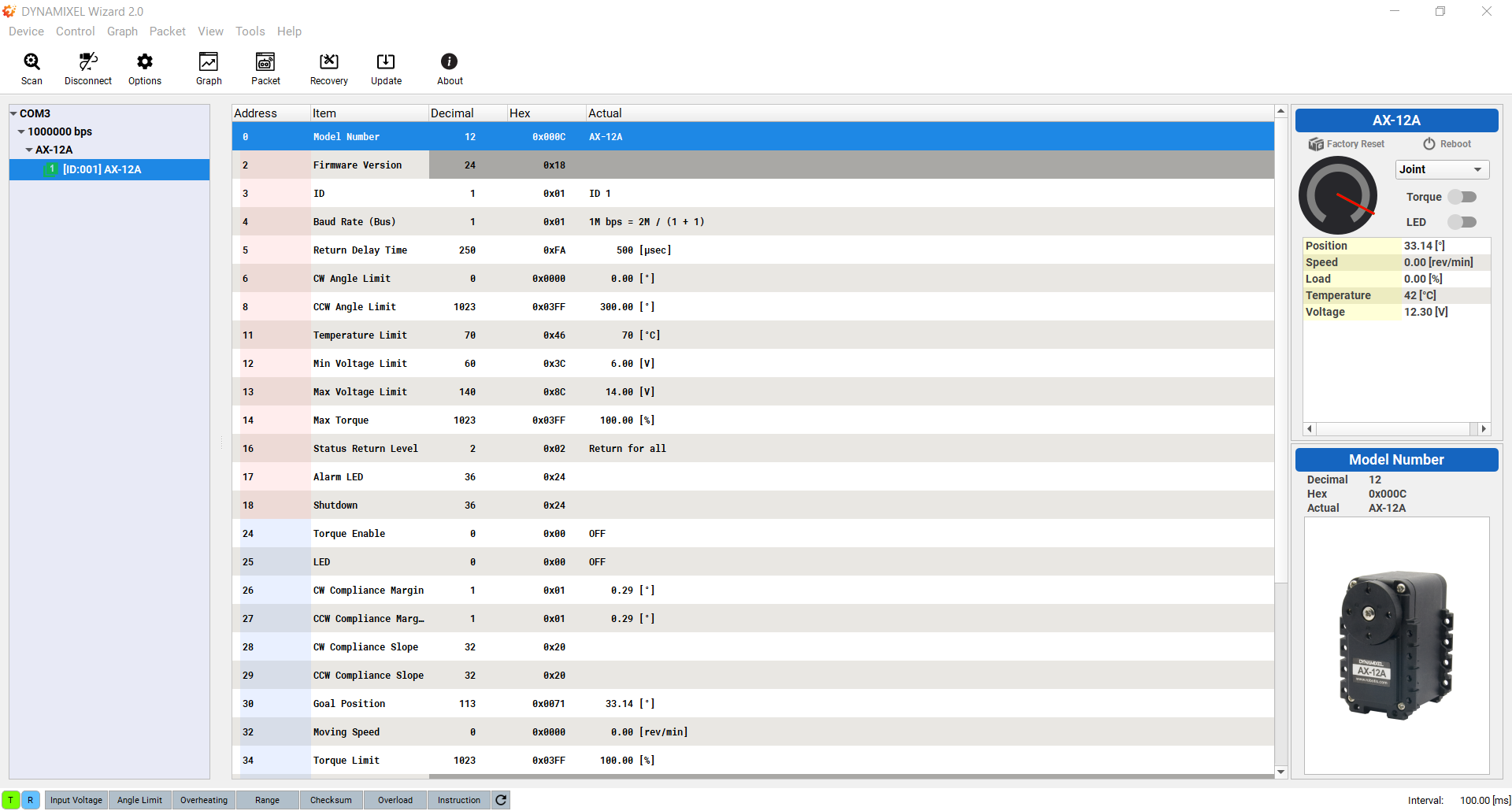
*Fig. 4 Ventana de Opciones de Dynamixel Wizard 2.0.*

Una vez seleccionado nuestros valores, damos clic en ***OK*** y procedemos a realizar la búsqueda de los motores conectados. Para comenzar la búsqueda, se da clic en ***Scan*** y una ventana aparecerá en donde se mostrará el progreso de la búsqueda. Una vez localizado el motor, mostrara en la ventana toda la información referente del motor conectado, como se muestra en la Fig. 5.



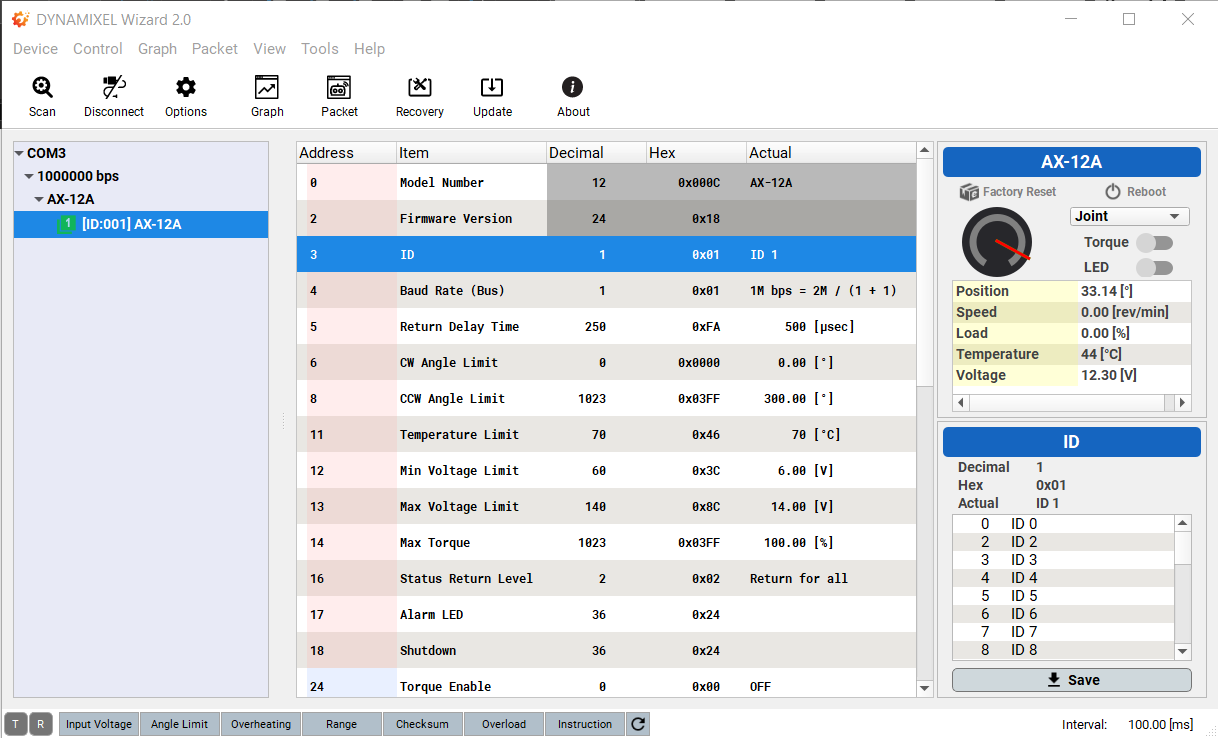
*Fig. 5 Ventana de Escaneo de Motores.*

Al haber localizado el motor, se habilitarán las opciones para modificar los registros del motor. Todos estos valores pueden ser utilizados para poder sacar información del motor o verificar su correcto funcionamiento, como lo muestra la Fig. 6. Dentro de esta ventana, nos enfocaremos en registros específicos, los cuales se explican a continuación:



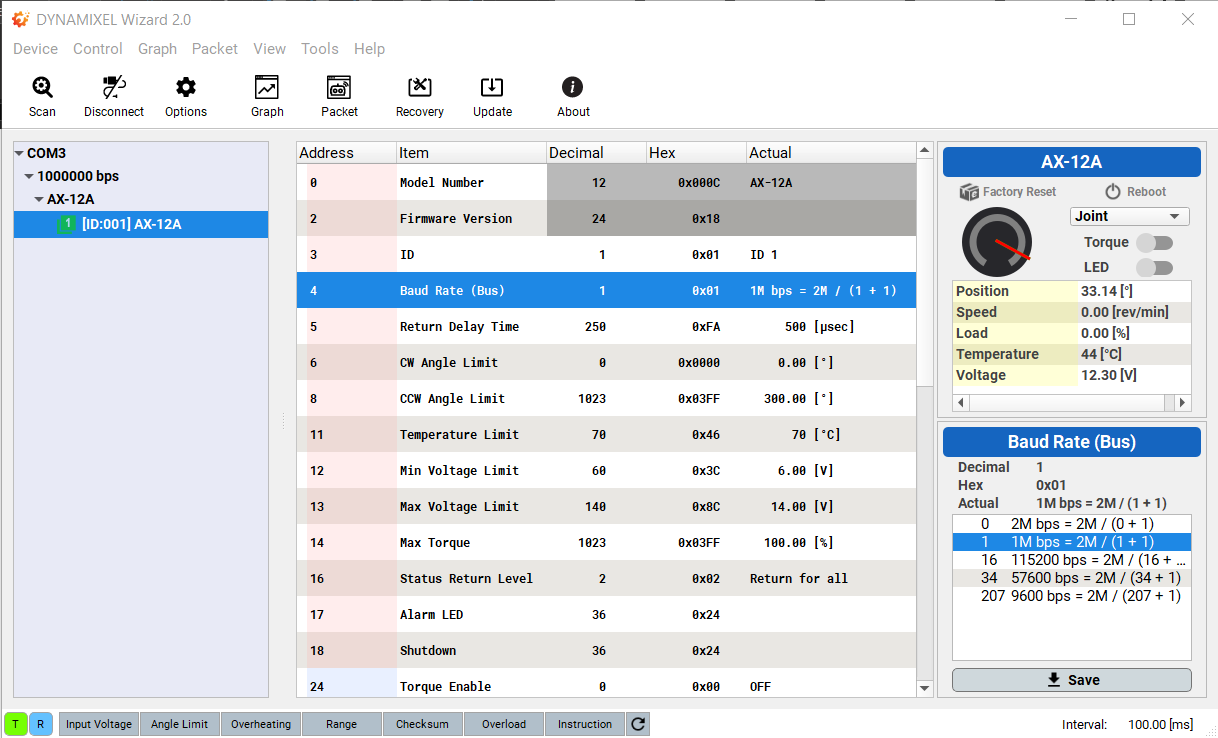
*Fig. 6 Ventana de Configuración de Registros.*

* Address 3 – ID: Permite cambiar el identificador de cada motor. Esto es necesario ya que, para su correcto funcionamiento, cada motor debe de tener un ID propio, Fig. 7.



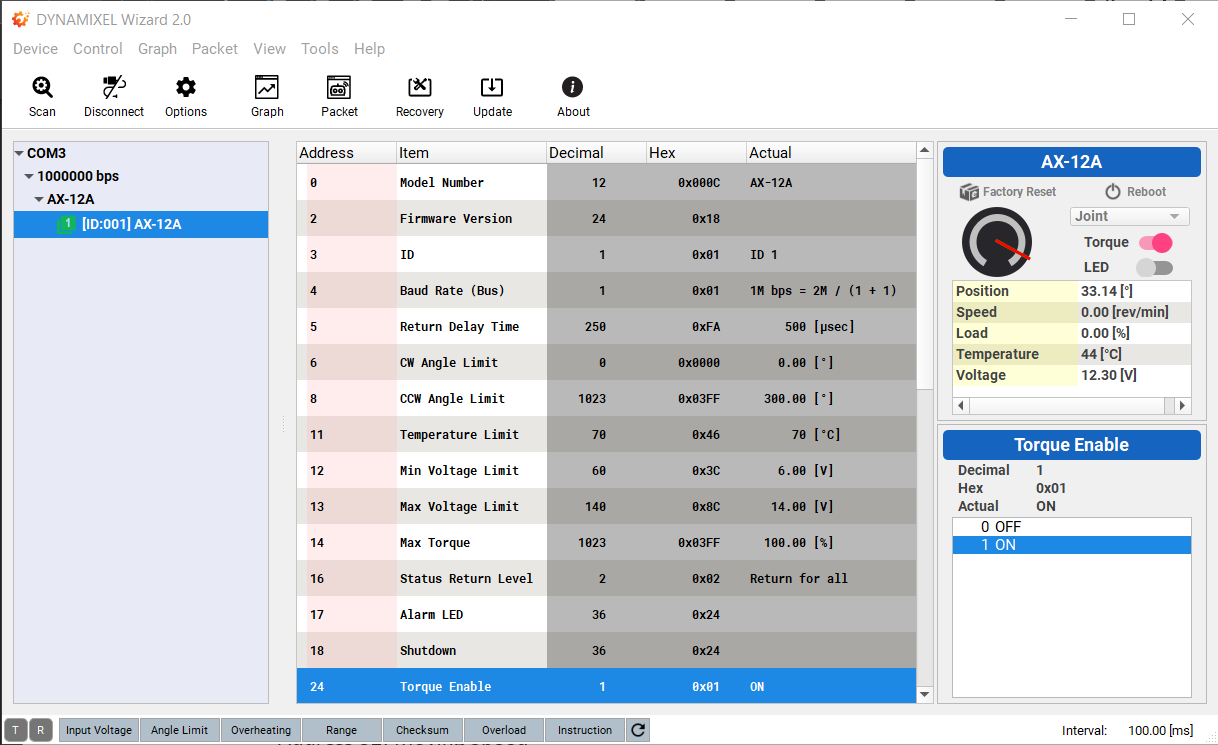
*Fig. 7 Registro de la Dirección 3.*

* Address 4: Baudrate: Es la velocidad de envio de datos. Para evitar la pérdida de datos, es recomendable trabajar con la velocidad de 1000000 (1M), Fig. 8



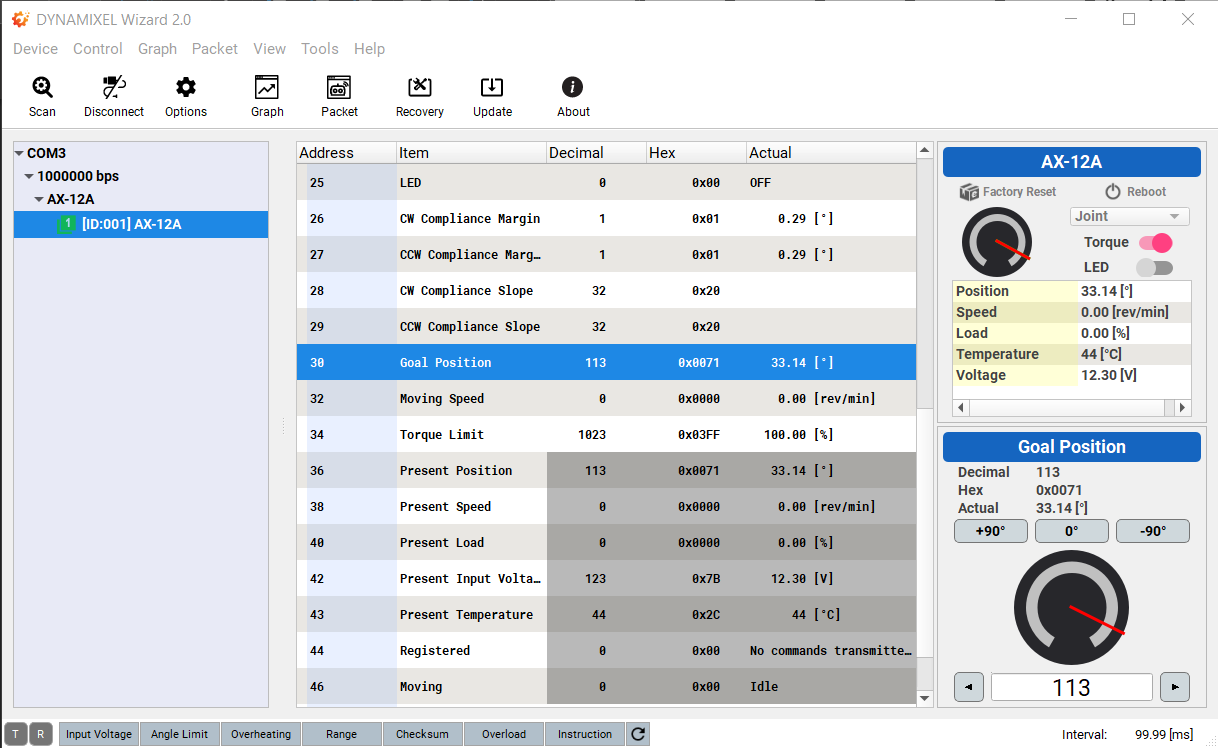
*Fig. 8 Registro de la Dirección 4.*

* Address 24: Torque Enable: Habilita la activación de torque en el motor. Al activarse, inhabilita los registros superiores que regulan las condiciones de operación del motor (Límite de Ángulo, Temperatura, Voltaje, etc.), Fig. 9.



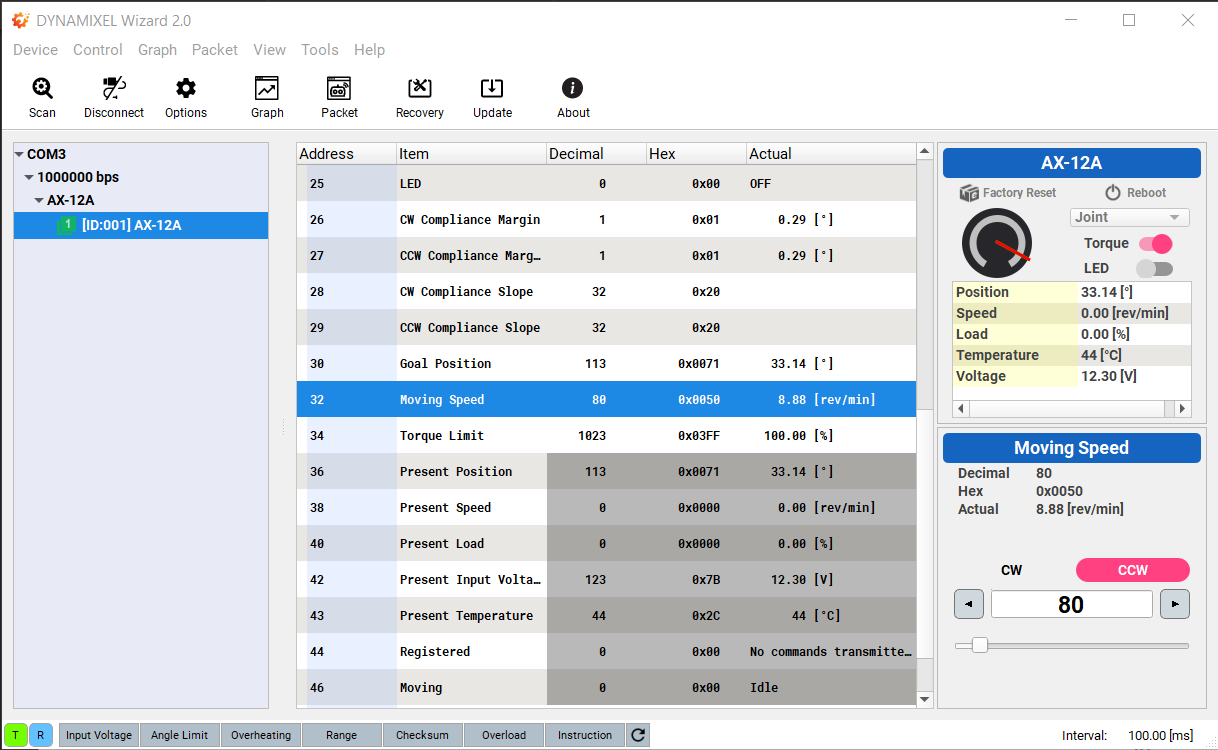
*Fig. 9 Registro de la Dirección 24.*

* Address 30 - Goal Position: Permite controlar de manera manual al motor. Con este registro podemos verificar si funciona de manera adecuada, Fig. 10



*Fig. 10 Registro de la Dirección 30.*

* Address 32: Moving Speed: Permite asignar la velocidad con la que operarán los motores. El valor predefino es 0, el cual indica que usará el máximo disponible, Fig. 11. Para un desempeño seguro, se recomienda situar este valor entre 50 y 100.



*Fig. 11 Registro de la Dirección 32.*

Al tener asignados estos valores, ya es posible trabajar de manera adecuada con los motores Dynamixel y programarlos para su uso.

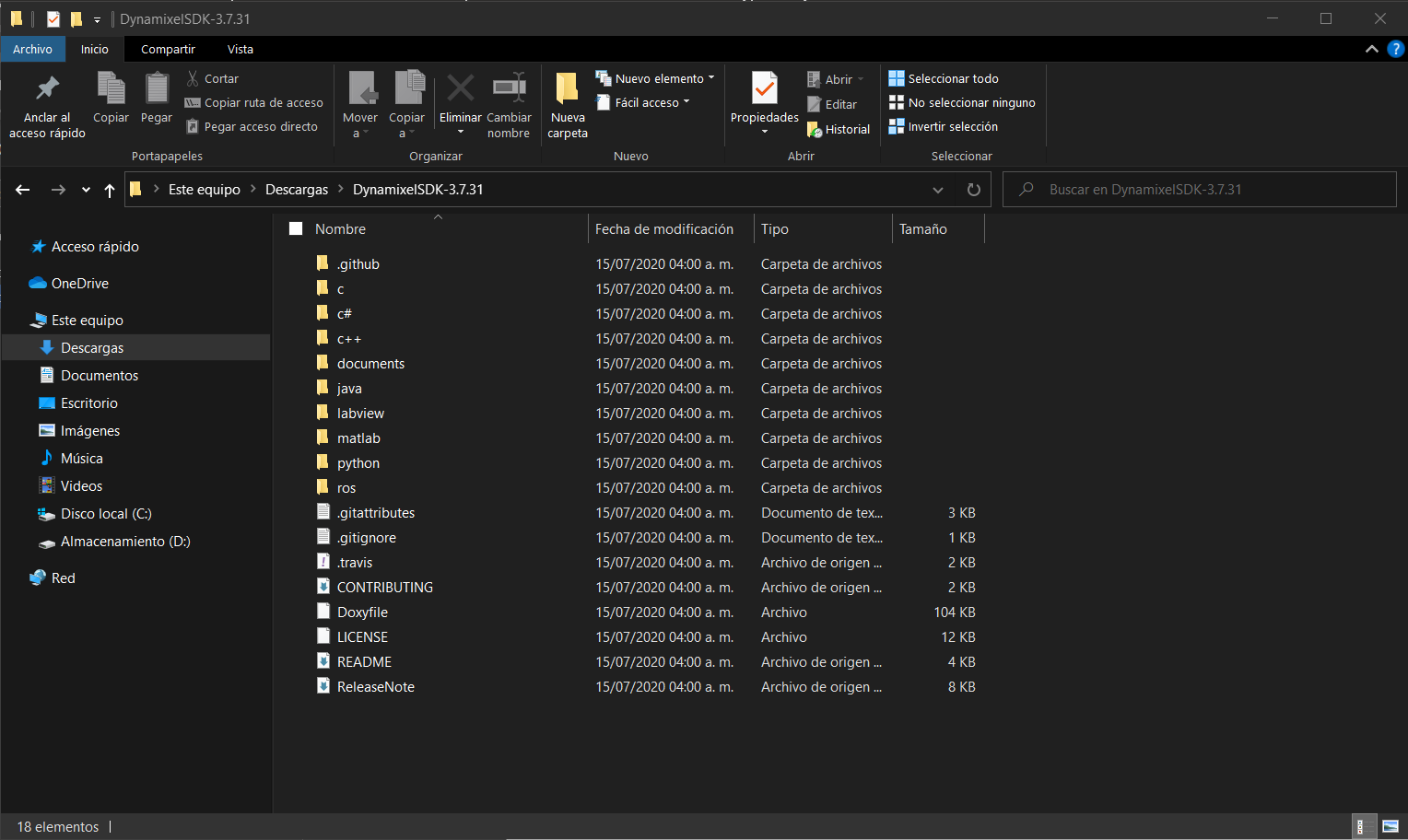
# Configuración del DynamixelSDK.

Al tener la configuración de los motores, es posible generar un mejor control de los Dynamixel. Cabe resaltar que Dynamixel Wizard 2.0 solo se puede utilizar para conocer la condición de operación de los motores. Si se desea utilizarlos en alguna aplicación específica, es necesario recurrir a un kit de desarrollo. ROBOTIS pone a disposición un SDK (Software Development Kit, por sus siglas en ingles) que permite trabajar en software específico o en un lenguaje de programación.

Este SDK permite la comunicación de los motores mediante el envío de instrucciones a los registros almacenados en memoria, esto mediante el uso de su encoder para poder interpretar esta información, por tal motivo, el conocer que registros son necesarios activar, ayuda al desarrollo de aplicaciones con este tipo de motores. El software de desarrollo DynamixelSDK tiene su código base en el lenguaje C, pero permite utilizarse en otros lenguajes, tales como C++, C#, Python, Java, MATLAB, LabVIEW y ROS. Al querer utilizar alguno de ellos, es indispensable preparar el SDK para que el programa pueda interpretar el código principal.

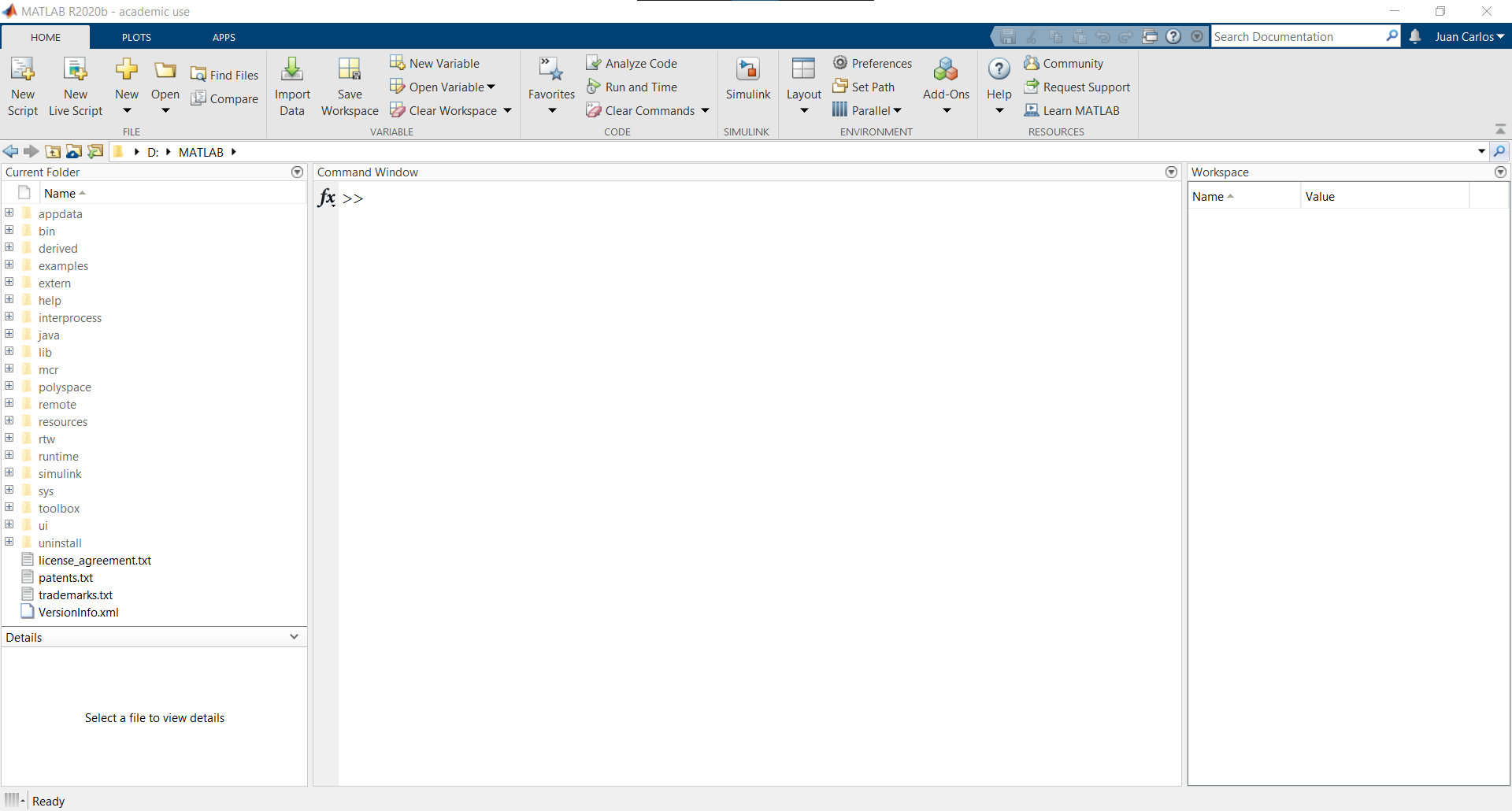
A continuación, se describe el proceso para obtener y configurar el DynamixelSDK para su uso mediante MATLAB.

* Se procede a descargar el DynamixelSDK de su repositorio de GitHub, el cual puede acceder en el siguiente enlace: [Releases · ROBOTIS-GIT/DynamixelSDK (github.com)](https://github.com/ROBOTIS-GIT/DynamixelSDK/releases)
* Una vez descargado, se procede a descomprimir el archivo en una carpeta de fácil acceso. Dentro de ella, encontraremos diversas carpetas que contienen los archivos a utilizar para cada programa/lenguaje en el que se desee trabajar, Fig. 12. Para nuestro caso en particular, utilizaremos los archivos dentro de la carpeta **“matlab”**. Esta carpeta contiene las funciones desarrolladas en scripts de MATLAB (archivos .m), que definen las instrucciones para controlar los motores. De igual forma, se utilizarán los archivos dentro de la carpeta **“c”**

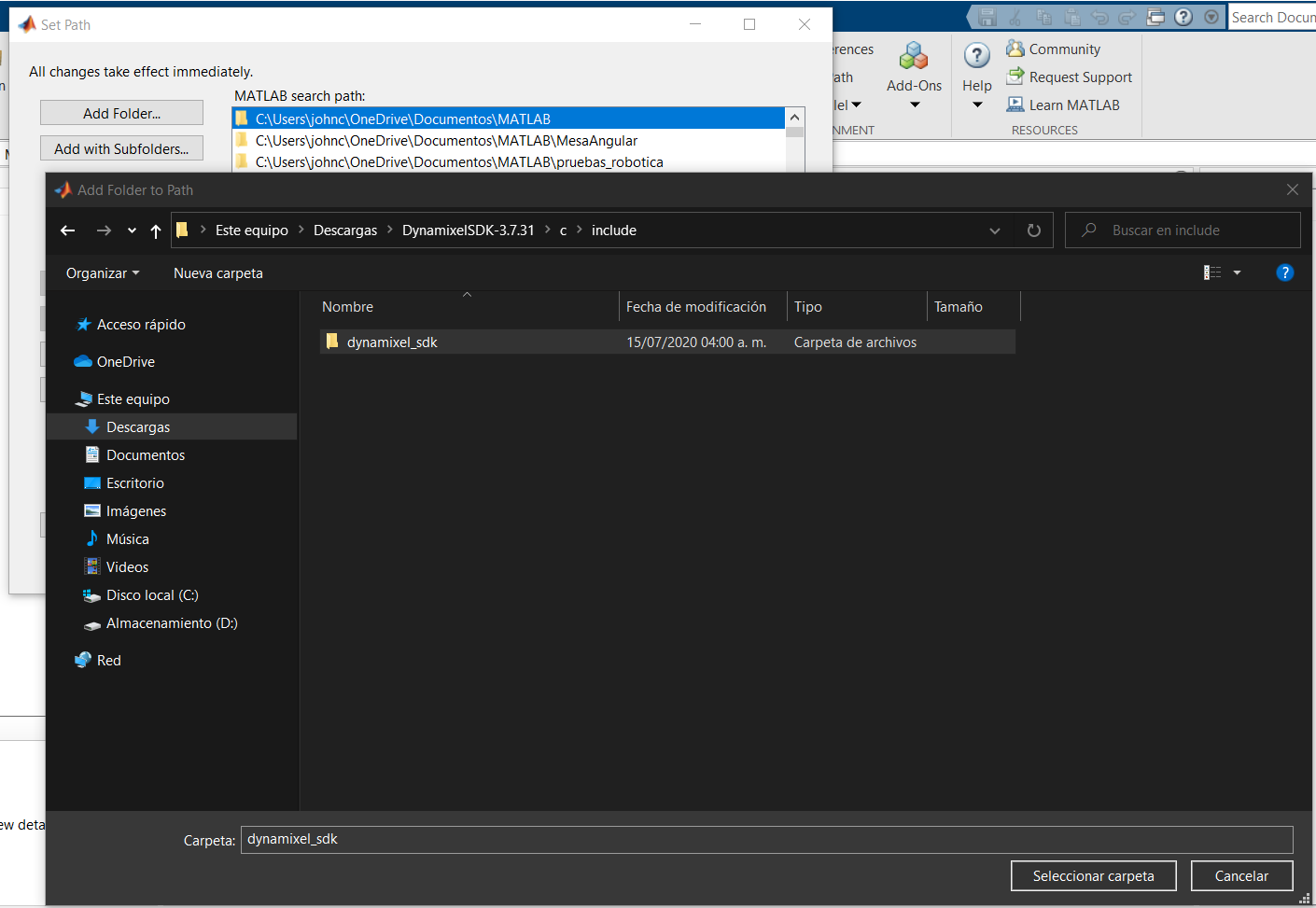


*Fig. 12 Carpeta del DynamixelSDK descomprimida.*

* Para poder utilizar estos scripts, es necesario incluir las librerías que definen a estas funciones, las cuales están escritas en C, por lo que se requiere que MATLAB conozca dónde se encuentran localizadas. Para realizarlo, dentro de MATLAB, primero seleccionamos **“Set Path”**, como lo muestra la Fig. 13.
* Se abrirá una ventana en donde podemos observar las carpetas que ya tiene reconocidas MATLAB. Para agregar la carpeta que contiene las funciones del SDK, seleccionamos “**Add folder…”** y buscamos dentro de la carpeta del SDK la siguiente ruta: **[Carpeta del DynamixelSDK/c/include/dynamixel\_sdk]**, como se muestra en la Fig. 14. Al tener seleccionada esa carpeta, damos clic en **“Seleccionar carpeta”.**

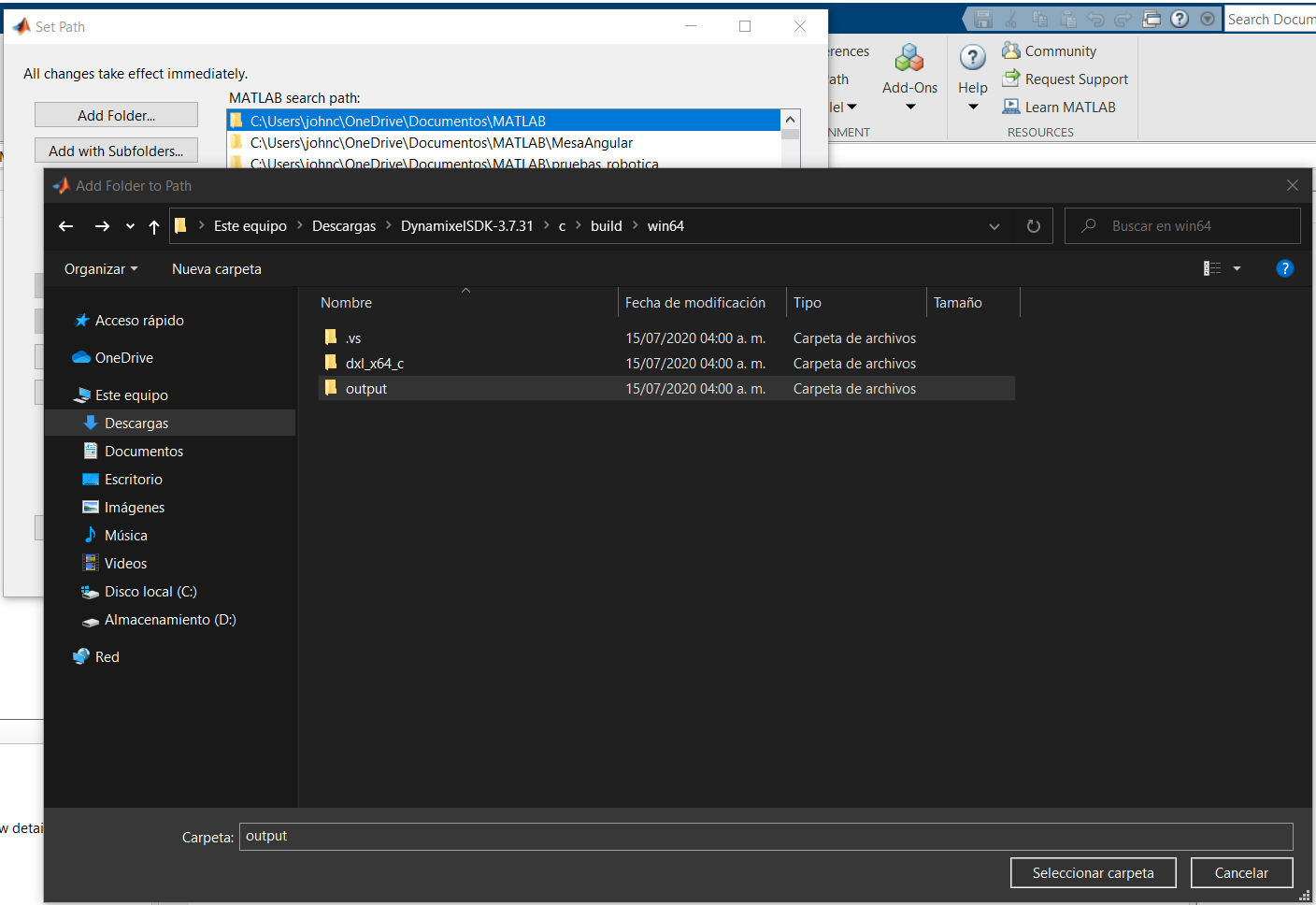
**

*Fig. 13 Ventana principal de MATLAB.*

**

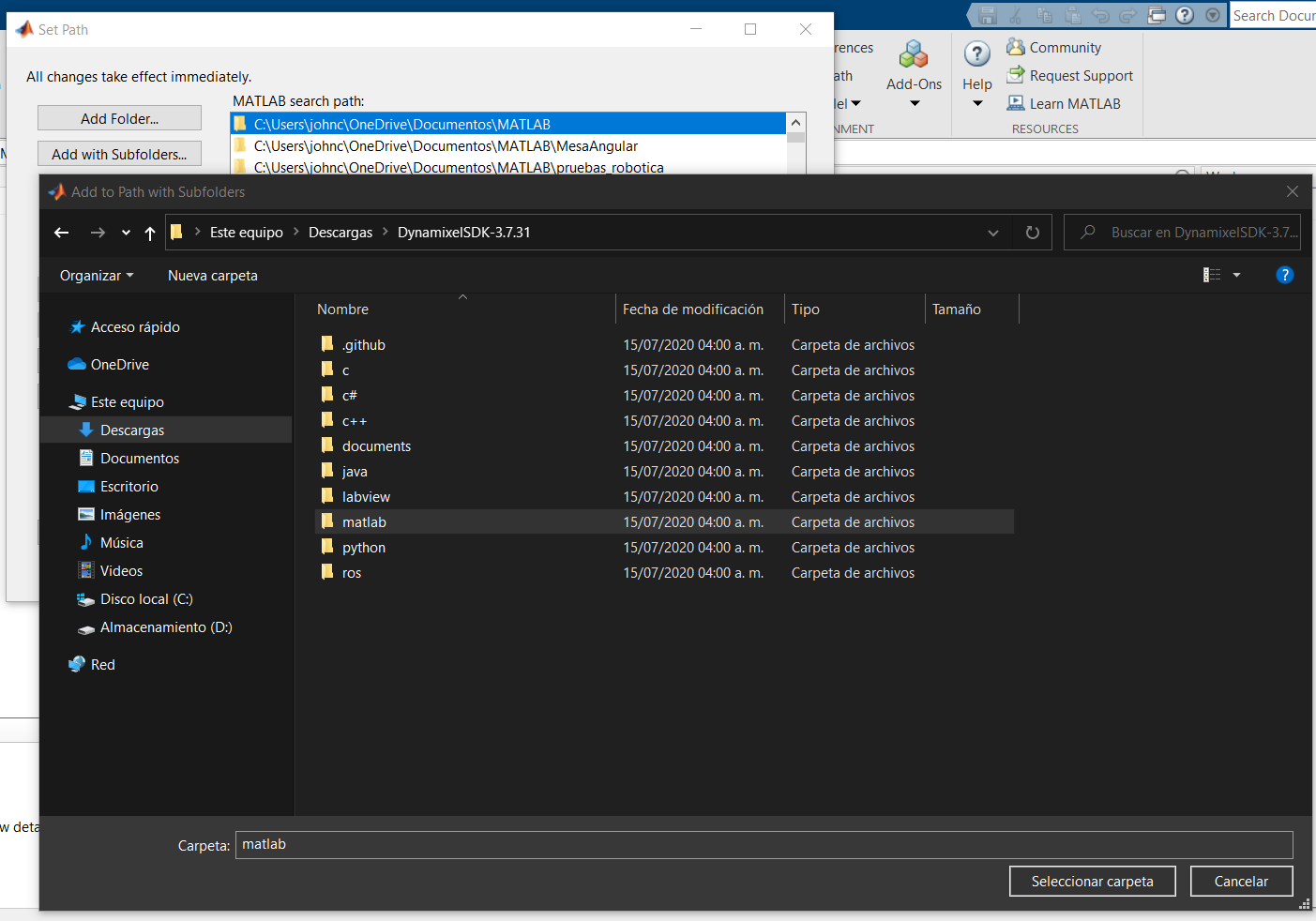
*Fig. 14 Selección de carpetas las funciones del SDK a MATLAB.*

* Posteriormente, es necesario añadir la ubicación de las librerías del SDK. Repitiendo los mismos pasos que en el punto anterior, seleccionamos “**Add folder…”** y buscamos dentro del SDK la siguiente ruta: **[Carpeta del DynamixelSDK/c/build/win64/output]**, como se muestra en la Fig. 15. Es importante destacar que existen versiones para cada sistema operativo (Windows, Linux y Mac) y tipo de sistema (32 bits y 64 bits), por lo que hay que seleccionar la que corresponda. Para este caso en particular, se utilizó la versión de Windows 64 bits.

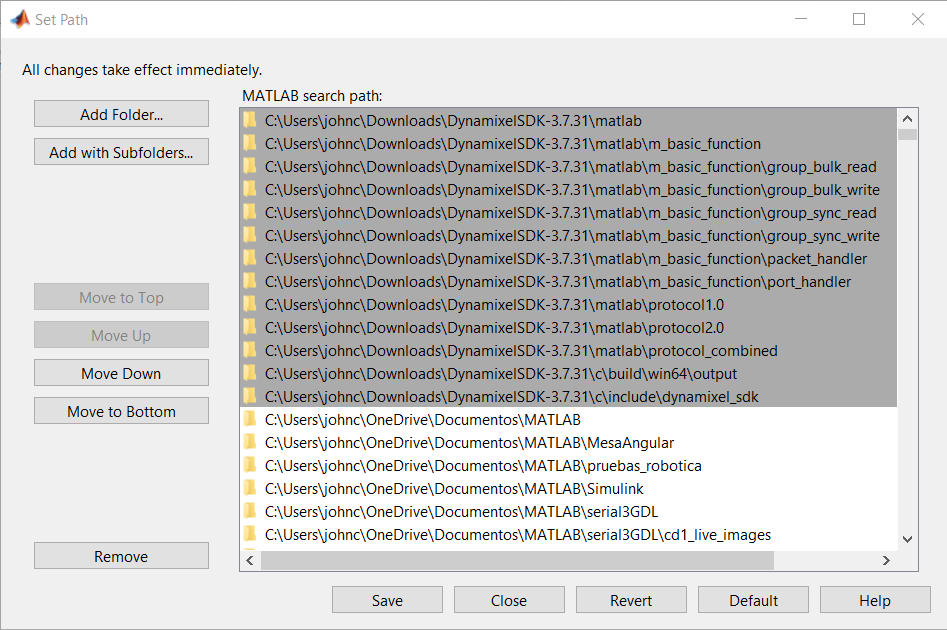


*Fig. 15 Selección de carpetas para las librerías del SDK a MATLAB.*

* Finalmente, se agregan los scripts de la carpeta **“matlab”**. Seleccionamos “**Add with subfolders…”** y añadimos la carpeta, como lo muestra la Fig. 16.
* Al finalizar, damos clic en **“Save”** para guardar los cambios realizados. Para salir, damos clic en **“Close”**, como lo muestra la Fig. 17.

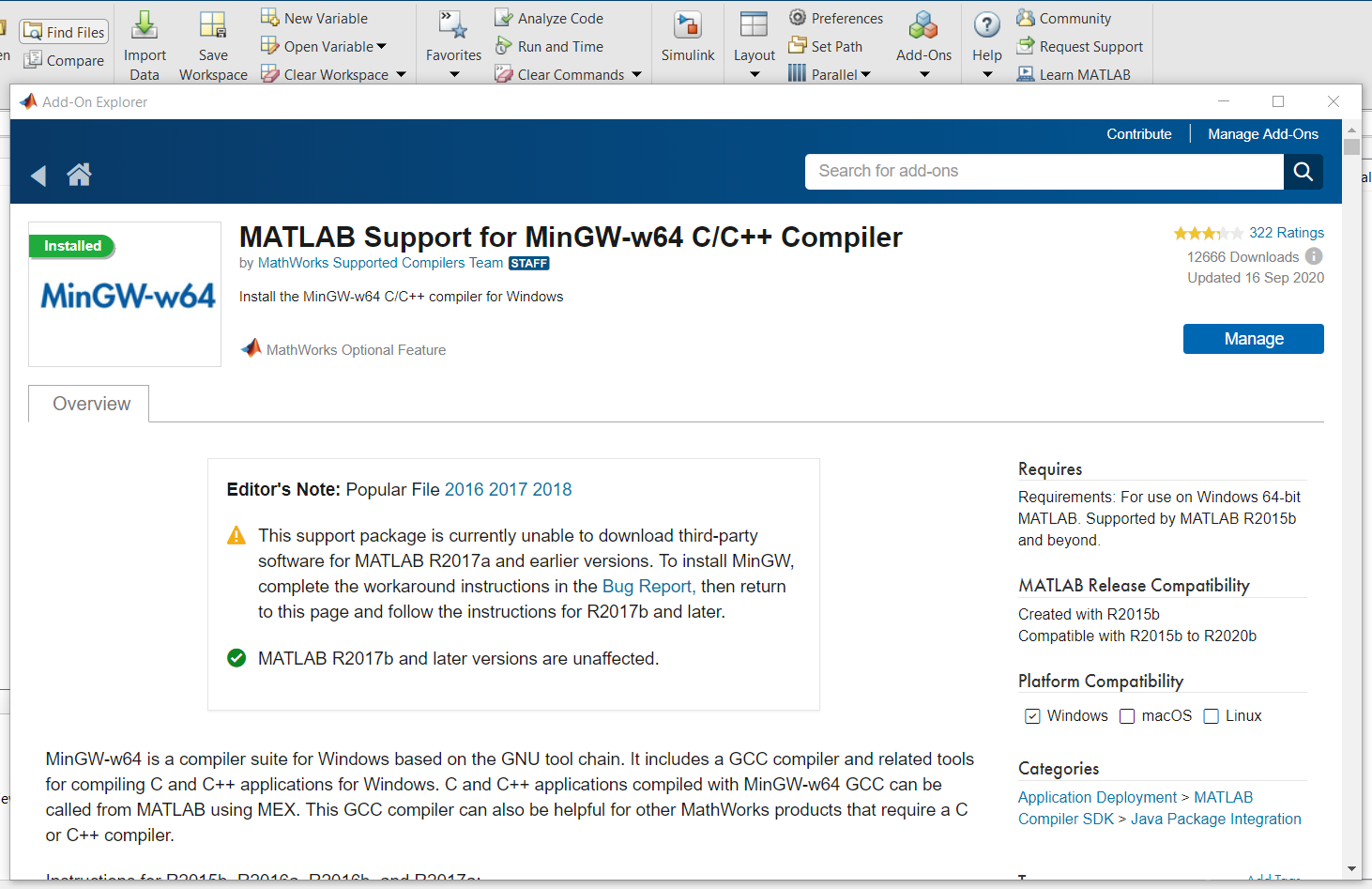


*Fig. 16 Selección de carpetas para los scripts del SDK a MATLAB.*



*Fig. 17 Vista final de las carpetas añadidas a MATLAB.*

Con esta configuración, las librerías y funciones pueden ser utilizadas directamente en MATLAB. Cabe resaltar que, al ejecutar estas funciones, pueda marcar un error al no poder interpretar las librerías. Por lo cual es necesario proveer a MATLAB de un compilador para su correcto funcionamiento. Recomendamos instalar **MATLAB Support for MinGW-w64 C/C++ Compiler,** el cual lo pueden instalar directamente del Add-On Explorer, dentro de la ventana principal de MATLAB.



*Fig. 18 Compilador recomendado para MATLAB.*