ESCUELA POLITECNICA NACIONAL

Laboratorio de Investigación en Inteligencia y Visión Artificial

gForce-mex interfaz con Matlab

Guía de instalación **paso a paso**

Jonathan Zea

Versión 1

7 de marzo de 2021

Contenido

Revisión de versiones

Fecha	Versión	Actualización	Autor
7/3/2021	1	Primera versión	Jonathan Zea

Resumen

En este documento se explica detalladamente el proceso de instalación y configuración de la interfaz para el brazalete GForce-Pro con Matlab utilizando funciones C++ MEX.

GForce-Pro

Descripción del dispositivo

Un archivo descolgado de internet dice:

gForce-Pro Features

- Wireless: BLE4.1
- Distance: 10 Meters
- Power: 0.1W
- Gesture Numbers: 8
- Gestures: Customer Re-Definable
- EMG Raw Data: Streaming Supported
  - Sample Rate: 1000Hz (max)
  - ADC: 8bit
  - Channels: 8
  - Gain: 1000
  - Filter: 20~500Hz Hardware BPF
- Quaternion: Supported
  - Sample Rate: 50Hz
- Motion Sensor Raw: Supported
  - Sample Rate: 50Hz
  - ADC: 16bit
  - 9-Axis: ACC、GYRO、MAG

- Software:
  - SDK For Windows
  - SDK For Android
  - SDK For Arduinio/STM32
  - Unity3D SDK
  - OTrain EMG Gesture Training APP
  - OPlatform EMG Waveform Scope and Data Record APP

Figura 1 Características del dispositivo según gForcePro_spec_v1.0-eng

Colocación del brazalete

En el sitio web del dispositivo se encuentra una guía del **gForce-100** con la siguiente información:

oymotion.github.io/gForce100_manual_v1.1-eng.pdf at master · oymotion/oymotion.github.io

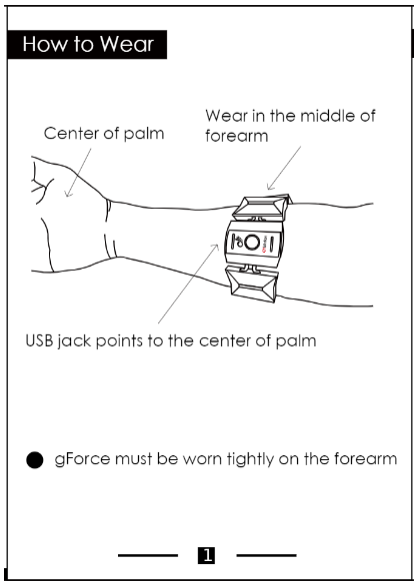


Figura 2 Colocación del gForce-100

Se podría asumir la misma recomendación para el gForce-Pro; sin embargo, una nota en gForceJoint - OYMotion sobre la colocación del dispositivo hace especial énfasis en el gForce-100.

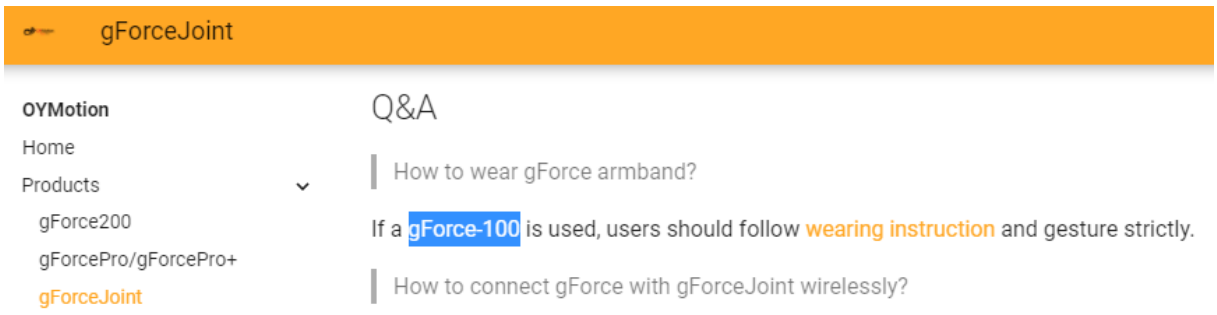


Figura 3 Sitio web de Oy Motion acerca de la colocación de los brazaletes

Instalación de gForce_mex (compilación de la función C++ MEX)

Esta guía termina con la compilación exitosa de la función **gForce_mex,** función C++ MEX para interfaz del GForce-Pro con Matlab.

La guía se debe entender en 3 partes:

- Instalación de dependencias y configuración inicial
- Prueba de concepto sobre compilación desde VS (sin referencias al gForce SDK) (función ejemplo app_matlab_mex)
- Compilación de gForce_mex

Prerrequisitos

- Visual Studio 2019 o superior
    - Desarrollo para el escritorio con C++ (ATL importante)

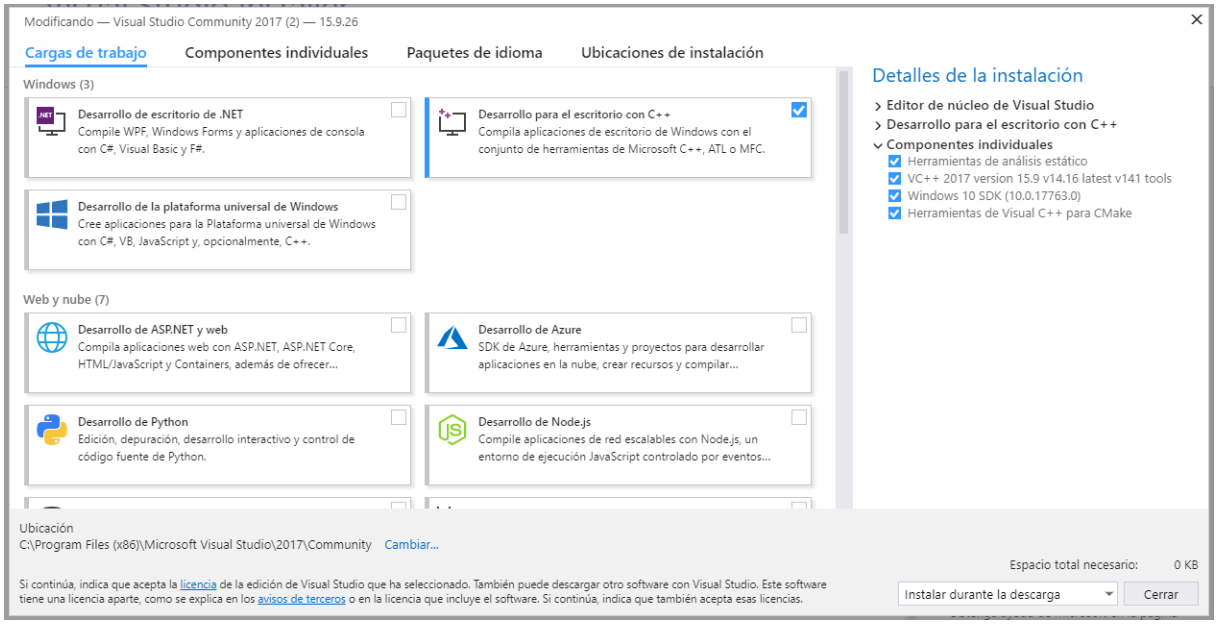


Figura 4 Interfaz de Visual Studio Installer

[Descargar Visual Studio 2019 para Windows y Mac (microsoft.com)]

- Mingw64 para Matlab

[MATLAB Support for MinGW-w64 C/C++ Compiler - File Exchange - MATLAB Central (mathworks.com)]

Variables de entorno

Copie el código suministrado a su computador. Dentro de este directorio, la carpeta:

```
.\gForce_Pro\bin\windows\
```

debe ser añadida a variables de entorno del sistema.

De no estar configurada correctamente las variables de entorno (ya compilada la función C++ Mex) se obtiene el siguiente error.

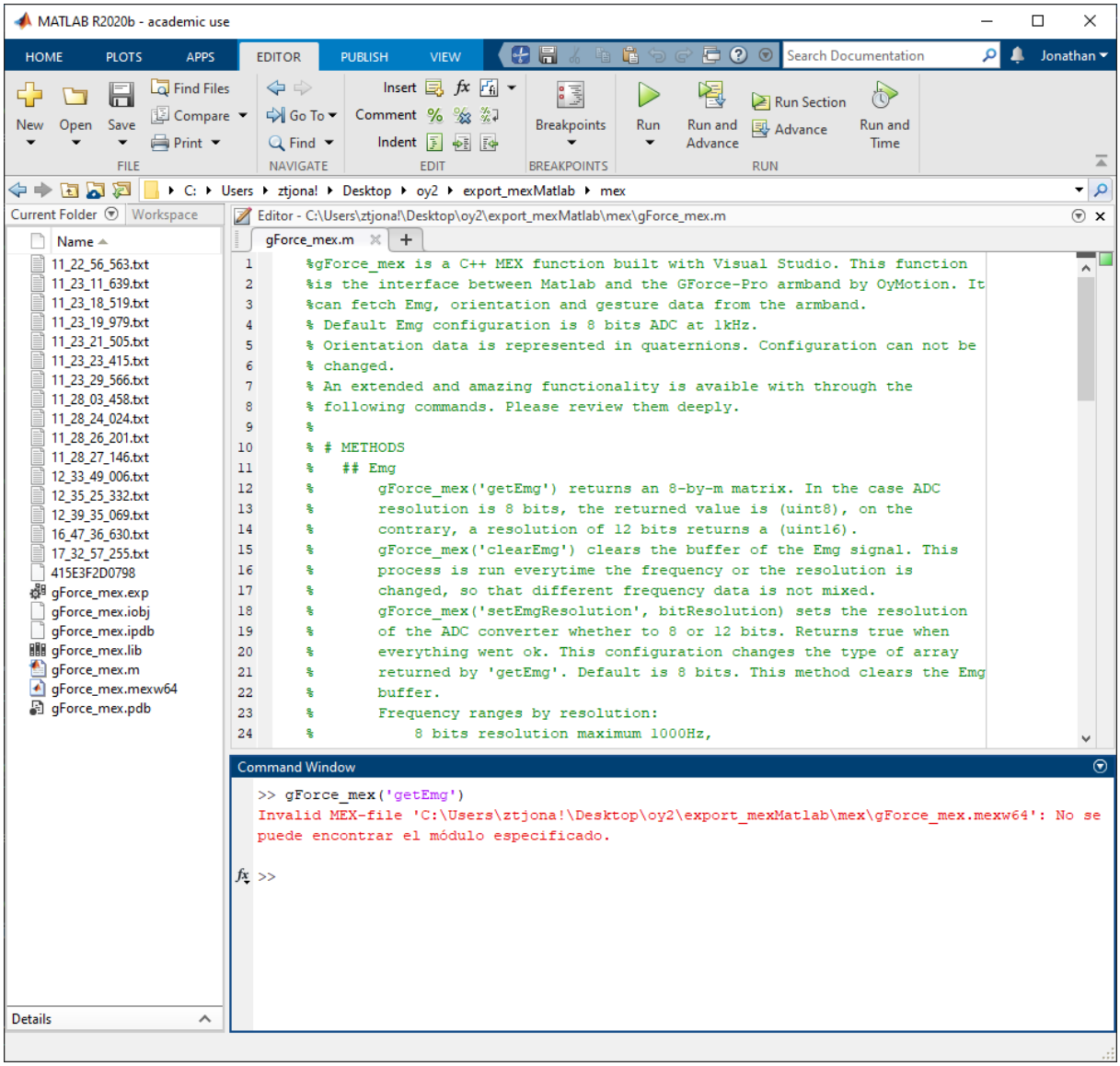


Figura 5 Error en Matlab al no incluir variables de entorno

Dongle Driver

Siga minuciosamente el siguiente tutorial para que el USB *dongle* esté correctamente instalado (i.e. hasta que se cree un puerto COM).

[FT232R USB UART Driver | USB Driver (usb-drivers.org)]

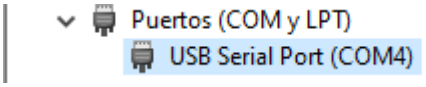


Figura 6 Administrador de dispositivos en la correcta instalación del drive

Descripción del código suministrado

Este documento viene acompañado de los siguientes directorios:

- app_mat_mex_v0

  Función C++ Mex de ejemplo para comprobar la instalación y funcionamiento correcto de VS

- gForce_Pro

  Función C++ mex interfaz de Gforce-pro con Matlab. Que a su vez contiene:

  - **\export_mexMatlab**: directorio "ligero" diseñado para contener la la función mex, contodas sus dependencias. Contiene:
    - **\bin**: donde se localiza los archivos .dll,
    - **\lib**: donde se localiza los archivos .lib,
    - **\include\:** donde se localiza los headers del SDK C++,
    - **\mex\:** donde se ha crear la función C++ mex compilada **gForce_mex**.
      NOTA: en este directorio el SDK de gForce_Pro genera varios archivos de texto ilegibles.
      Este directorio contiene además la documentación de la función mex **gForce_mex.m**.
  - **\gForce_mex\:** directorio con el proyecto de Visual Studio para la compilación de la función Mex.
    Nota: se generaron estos dos directorios para separar la función mex del proyecto de Visual Studio, ya que los proyectos en VS son sumamente pesados (>200Mb).

La función gForce_mex.mexw64 se encuentra en:

```
.\gForce_Pro\export_mexMatlab\mex\gForce_mex.mexw64
```

Copie **export_mexMatlab** a su proyecto en Matlab, o añada esta carpeta al *path* de Matlab.

Configuración de la función C++ mex en Visual Studio (VS)

La función C++ mex **gForce_mex** debe ser compilada nuevamente en cada sistema operativo y para cada nueva versión de Matlab. La compilación de esta función se realiza en Visual Studio.

A pesar de que se recomienda que las funciones mex sean compiladas directamente en Matlab, el SDK de gForce-Pro tiene varias referencias, configuraciones y dependencias de Windows que lo convierte en intratable en Matlab, siendo un poco menos difícil compilar la función mex en Visual Studio. El procedimiento para compilar funciones Mex en VS se encuentra en el siguiente tutorial:

Compiling MEX Files without the mex Command - MATLAB Answers - MATLAB Central (mathworks.com)

El proyecto en VS adjunto **gForce_mex.sln** tiene la mayoría de las configuraciones listas. Sin embargo, hay unas pocas que deben ser realizadas dependiendo del escenario (i.e. versión de Matlab, sistema operativo, etc.).

Primeros pasos en Visual Studio

En las siguientes secciones se describirá el procedimiento para compilar una función mex en VS. Allí, se describe paso a paso la compilación del proyecto **app_mat_mex_v0**. El mismo proceso lo debe llevar a cabo para el proyecto verdadero **gForce_mex**.

Se utiliza el proyecto **app_mat_mex_v0** como primer paso para comprobar la correcta compilación de una función MEX. Esto debido a que **app_mat_mex_v0** no contiene ninguna referencia al SDK de GForce_pro.

Abra el archivo:

```
.\gForce_Pro\app_mat_mex_v0\app_mat_mex_v0.sln
```

Al hacerlo, se debe iniciar VS. Localice el archivo principal **app_mat_ex_v0.cpp** dentro de "Source Files" en "Solution Explorer".

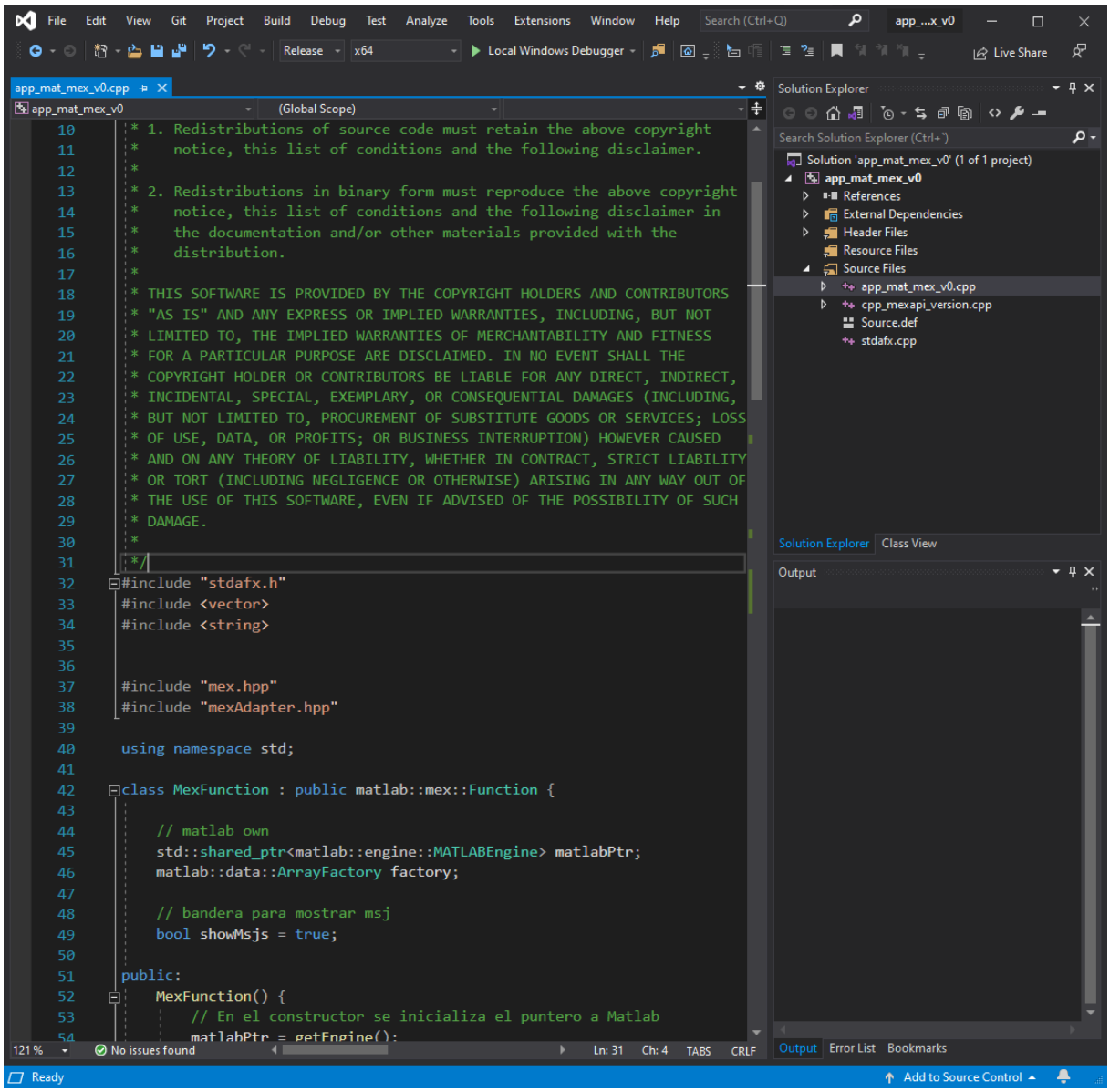


Figura 7 Interfaz de VS

Incluyendo las dependencias de las funciones C++ MEX (*headers*)

Todas las funciones C++ MEX deben incluir los *headers*:

```
#include "mex.hpp"
#include "mexAdapter.hpp"
```

en el código fuente.

En Visual Studio es necesario configurar la ruta de estos archivos *headers*. En caso de no existir o estar mal configurado, Visual Studio muestra la siguiente alerta:

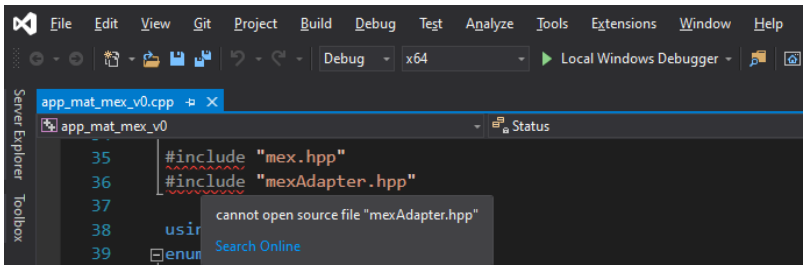


Figura 8 Alerta cuando está mal configurada la ruta de los headers mex

El ejemplo **app_mat_mex_v0** tiene configurada las rutas:

```
C:\Program Files\MATLAB\R2020b\extern\version
C:\Program Files\MATLAB\R2020b\extern\include
```

correspondientes a la versión Matlab 2020 b.

Configuración del *path* de los *headers* requeridos para las funciones MEX

Este proceso es necesario cuando se usa versiones de Matlab diferentes a la R2020b, o cuando la instalación se ubicada en directorios diferentes. En caso de utilizar Matlab R2020b y haber comprobado la existencia de los dos directorios anteriores, **se puede saltar esta sección**.

Para acceder a las **Propiedades**, dar clic derecho en el nombre del proyecto (e.g. **app_mat_mex_v0)**, luego en **Propiedades** como se indica en la siguiente figura.

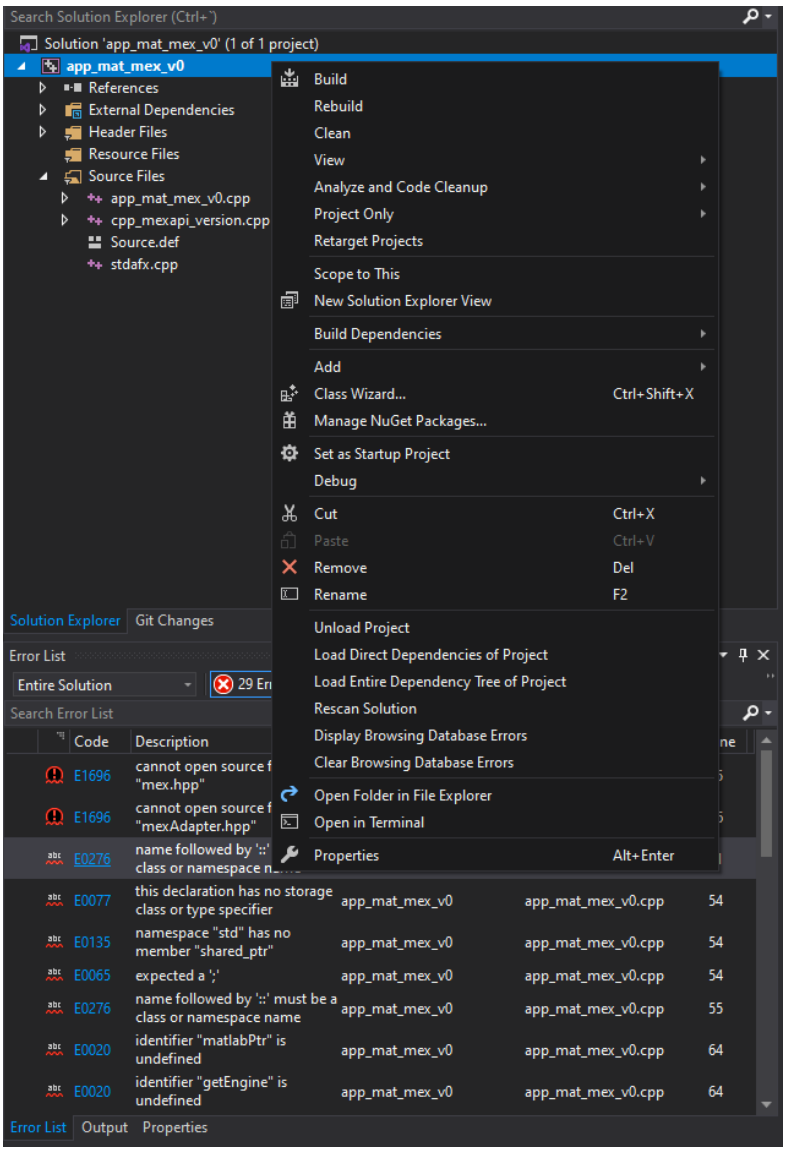


Figura 9 Accediendo al panel de Propiedades

En C/C++ → General → *Additional Include Directories* (como se muestra en la figura) añada los *paths* correspondientes.

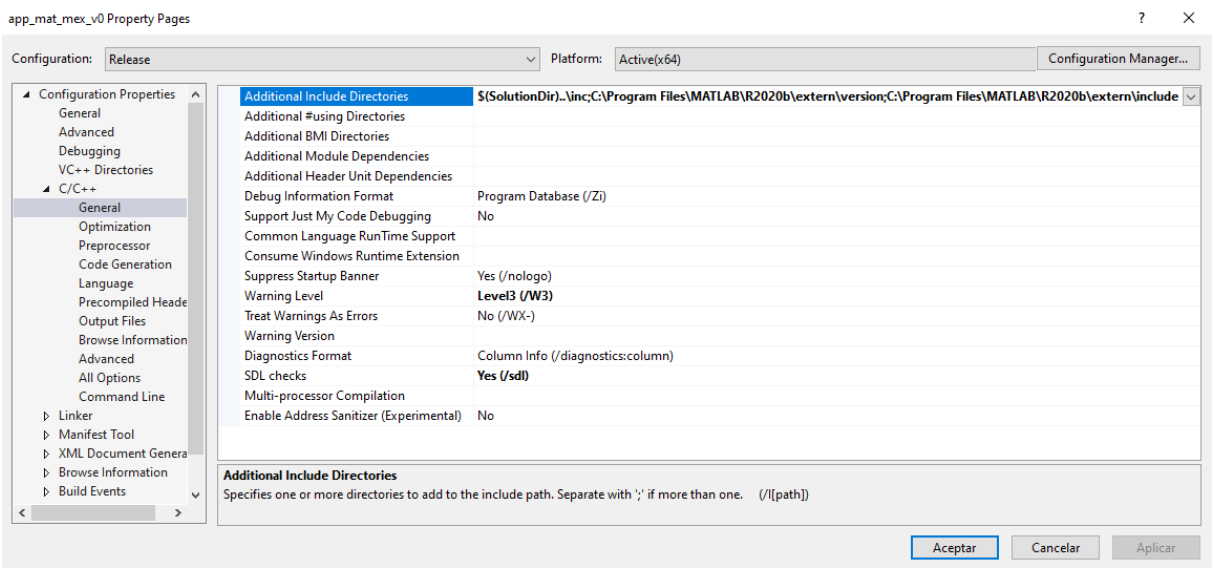


Figura 10 Panel de Propiedades

IMPORTANTE

Cerciórese que esté realizando los cambios en la configuración **Release** y **Active(x64)** en la parte superior del panel de Propiedades. Adicionalmente, verifique que en la ventana principal de Visual Studio se encuentre trabajando en **Release** y **x64** como se muestra a continuación. Hecho esto, debe desaparecer cualquier alerta debajo de las directrices *include*.

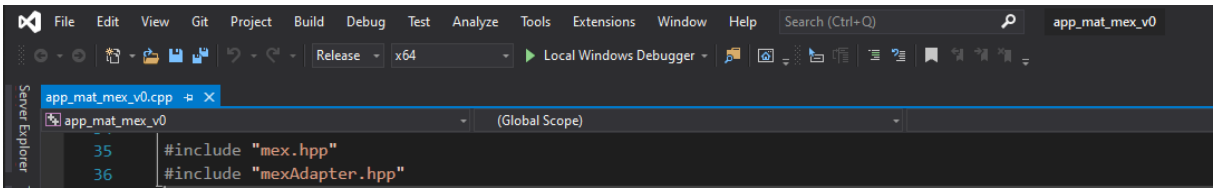


Figura 11 Release y x64 en Visual Studio

Configuración de las librerías libMatlabDataArray.lib, libmx.lib, libmex.lib y libmat.lib

Estas librerías son necesarias para la compilación de funciones mex.

Estas ya están declaradas en linker→ input→Additional Dependencies.

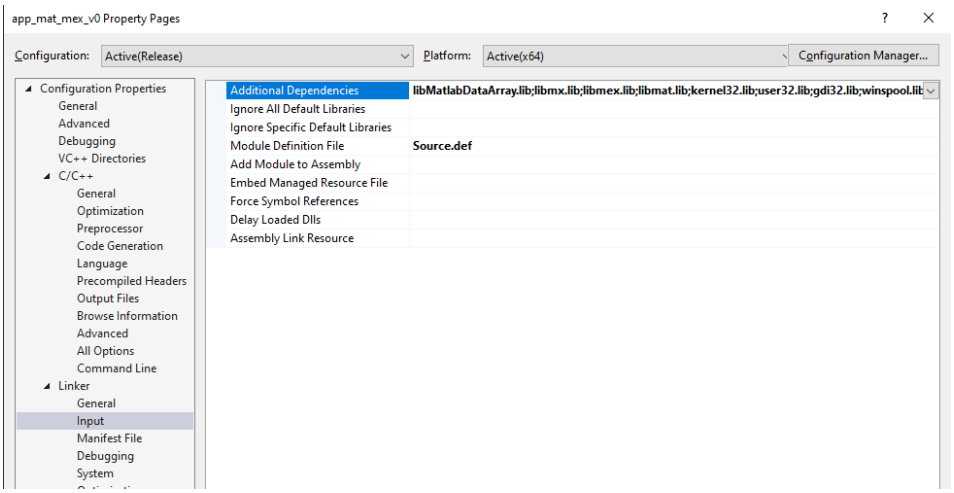
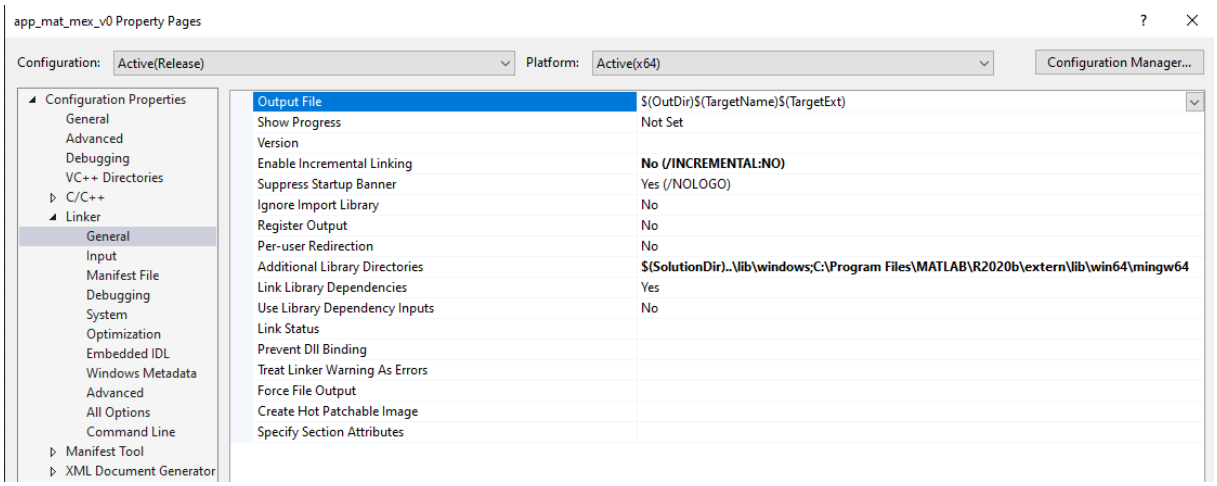


Figura 12 Incluyendo libMatlabDaraArray.lib y compañía.

Sin embargo, su ruta puede cambiar, ya que depende de la versión de Matlab. Para configurar este parámetro se debe ir a Linker → General→Additional Library directories



Y se debe cambiar el directorio:

```
C:\Program Files\MATLAB\R2020b\extern\lib\win64\mingw64\
```

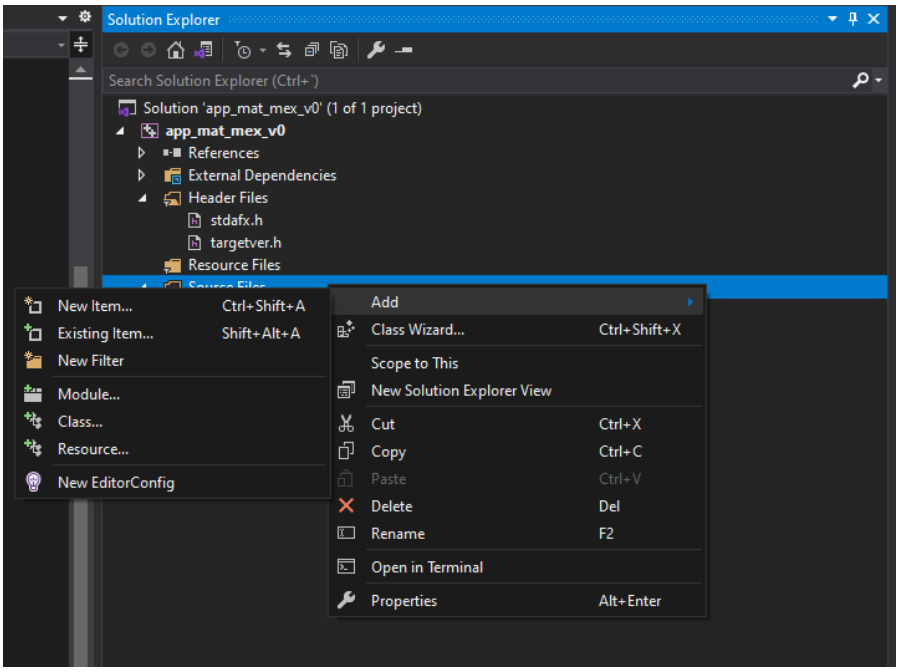
Por el directorio correspondiente.

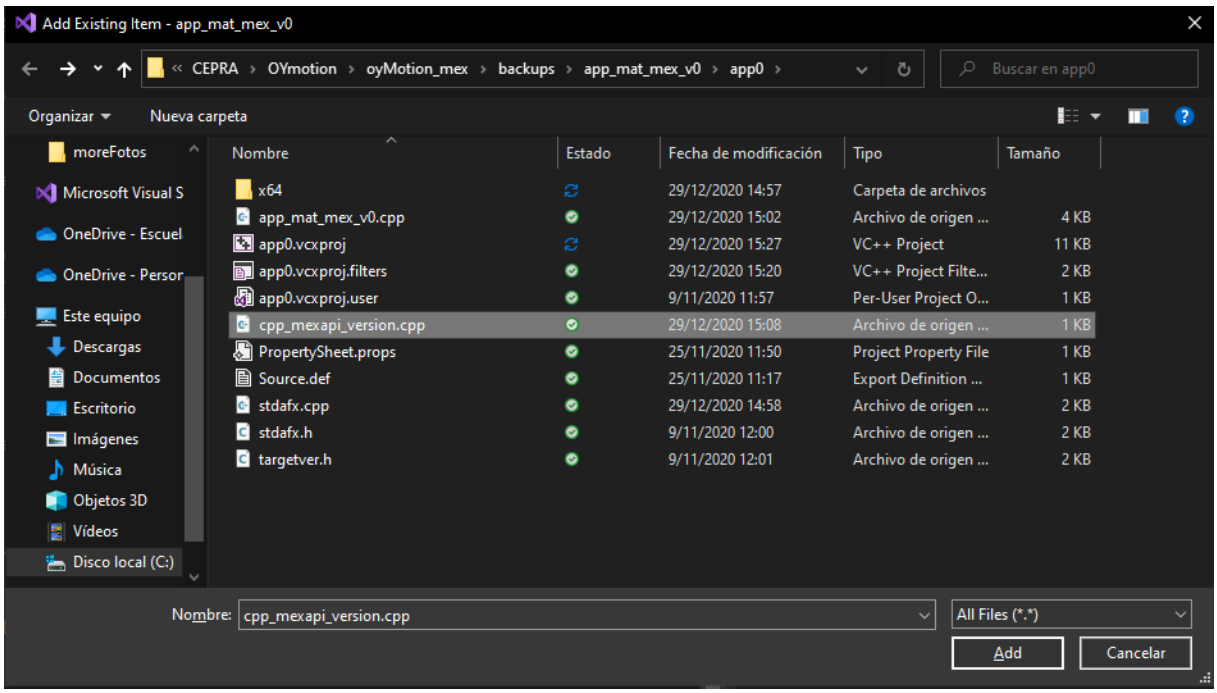
cpp_mexapi_version

El tutorial referenciado en Configuración de la función C++ mex en Visual Studio (VS), menciona que se debe añadir el archivo **cpp_mexapi_version.cpp** directamente desde los directorios de Matlab. Sin embargo, la ejecución del gForce SDK require que se hagan cambios a este archivo por lo que:

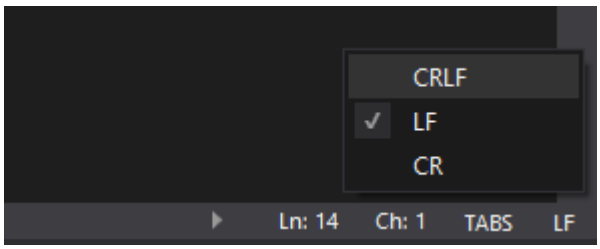
Se debe crear una copia del archivo **cpp_mexapi_version.cpp** en la raíz del proyecto. Este archivo se encuentra en `C:\Program Files\MATLAB\R2020b\extern\version\`

Después de haber realizado la copia, se debe añadirla al proyecto como se muestra a continuación.



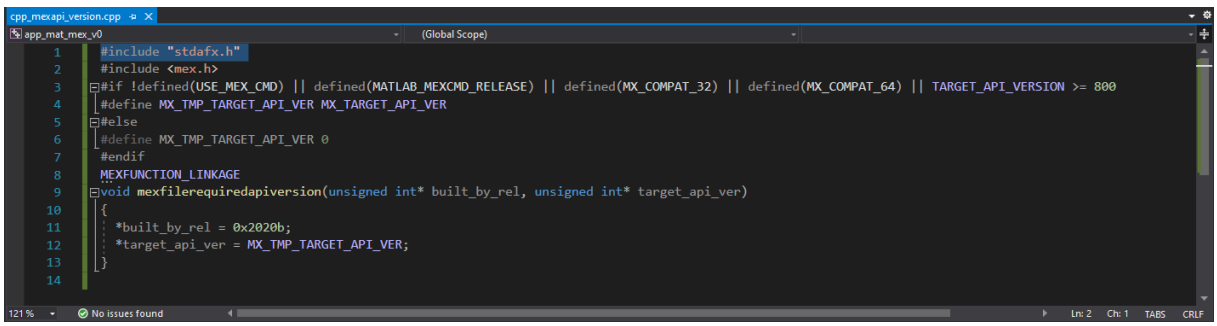


Y se debe hacer los siguientes cambios:



Por cosas de la vida, además, se debe incluir al inicio de este archivo el *header*:

```
#include "stdafx.h"
```

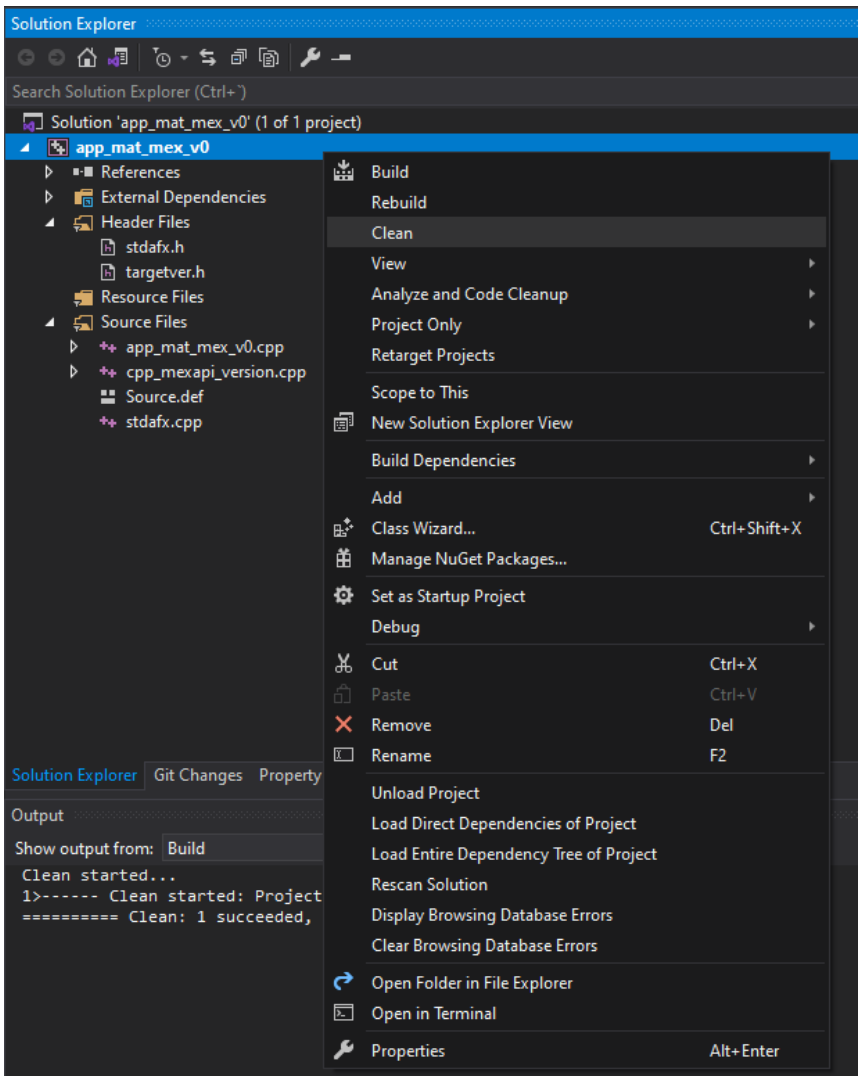


Compilación

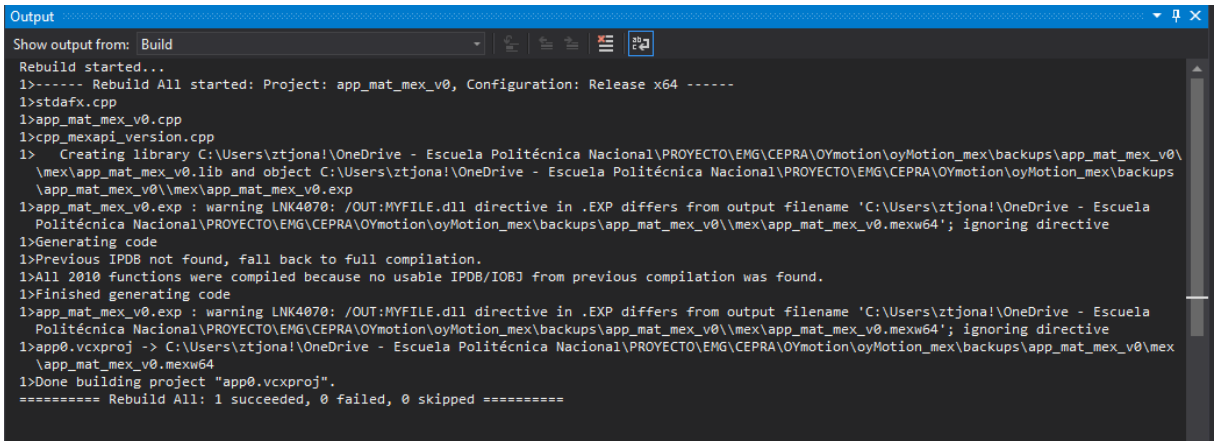
Comprobación de dependencias (compilación de la función C++ MEX de ejemplo)

Para comprobar la correcta instalación y configuración de todas las dependencias, se incluyó un ejemplo llamado **app_mat_mext_v0**.

Después de completados los pasos anteriores, se debe dar en el nombre del proyecto, botón derecho, **Clean** y luego de hecho esto, se debe dar clic en **Build** como se muestra en las figuras.

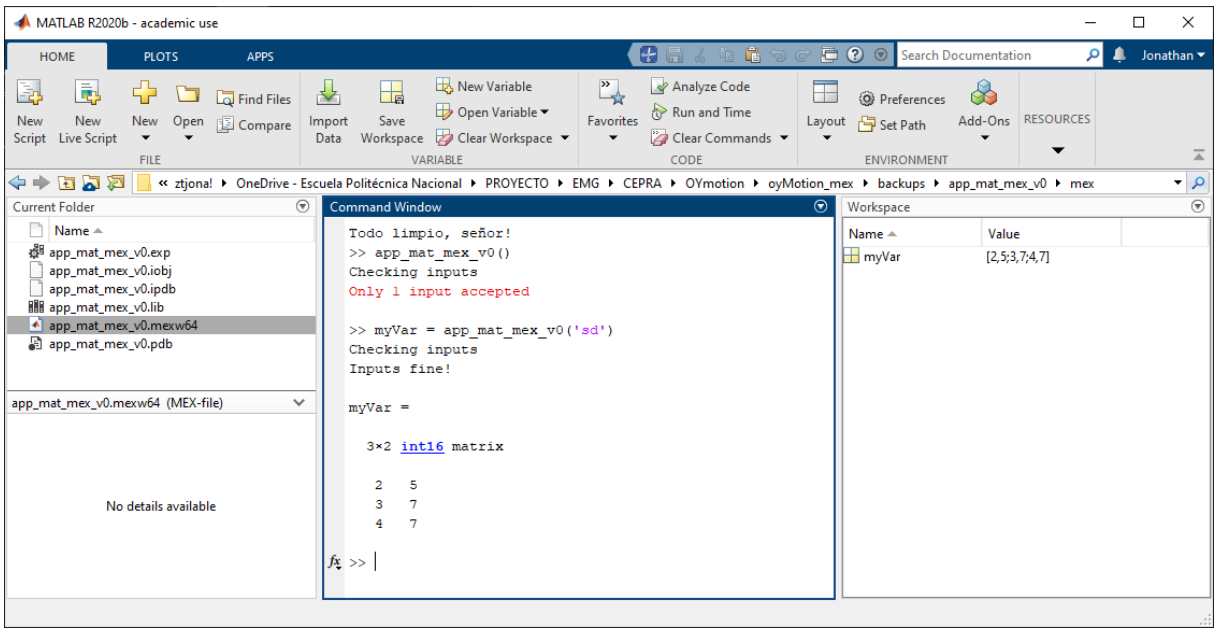


El panel de salida nos indica el estado de la compilación. Debe verse similar a la siguiente figura:



Ahora se puede proceder a ejecutar la función C++ MEX en Matlab. La función C++ MEX se genera dentro de la carpeta **.\mex** relativa al directorio del proyecto (de ser necesario, esto es reconfigurable en el panel de Propiedades). Diríjase a ese directorio y ejecute la función C++ MEX con el comando.

```
myVar = app_mat_mex_v0('textoRandom')
```



Compilando la función importante

Repita el proceso anterior, pero para el proyecto:

```
.\gForce_Pro\gForce_mex\gForce_mex.sln
```

Localice el archivo principal **main_mex.cpp** dentro de "Source Files" en "Solution Explorer".

Ejecución final

Después de resueltos todos los inconvenientes imaginables, ya debería poder ejecutar la función **gForce_mex** en Matlab. Ingrese los comandos siguientes:

```
>> gForce_mex('vibrate', 500)
>> data = gForce_mex('getEmg');
```

y compruebe que obtuvo una respuesta similar a la de la figura.

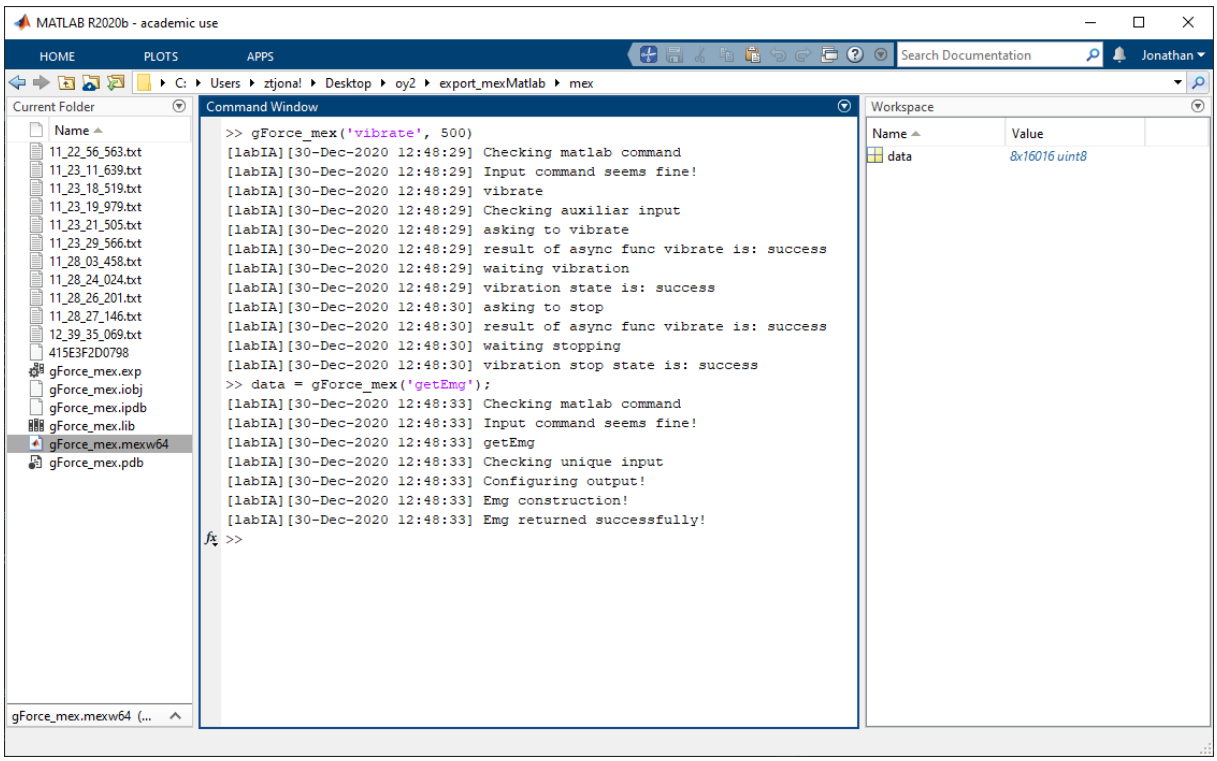


Figura 13 Ejecución exitosa de la función C++ MEX (gForce_mex)

Para más información sobre la función **gForce_mex** diríjase a la ayuda con el comando:

```
>> help gForce_mex
```

```
>> help gForce_mex
 gForce_mex is a C++ MEX function built with Visual Studio. This function
 is the interface between Matlab and the GForce-Pro armband by OyMotion. It
 can fetch Emg, orientation and gesture data from the armband.
  Default Emg configuration is 8 bits ADC at 1kHz.
  Orientation data is represented in quaternions. Configuration can not be
  changed.
  An extended and amazing functionality is avaible with through the
  following commands. Please review them deeply.

  # METHODS
    ## Emg
        gForce_mex('getEmg') returns an 8-by-m matrix. In the case ADC
        resolution is 8 bits, the returned value is (uint8), on the
        contrary, a resolution of 12 bits returns a (uint16).
        gForce_mex('clearEmg') clears the buffer of the Emg signal. This
        process is run everytime the frequency or the resolution is
        changed, so that different frequency data is not mixed.
        gForce_mex('setEmgResolution', bitResolution) sets the resolution
        of the ADC converter whether to 8 or 12 bits. Returns true when
        everything went ok. This configuration changes the type of array
        returned by 'getEmg'. Default is 8 bits. This method clears the Emg
        buffer.
        Frequency ranges by resolution:
            8 bits resolution maximum 1000Hz,
            12 bits resolution maximum 500Hz. In the case frequency was in
            a higher value than 500Hz, it is changed automatically.
        gForce_mex('setEmgSamplingRate', frequency) sets the sampling rate
        of the Emg ADC converter. View Frequency ranges by resolution.
        Returns true when everything went ok. This method clears the Emg
        buffer.
    ## Orientation
```

Figura 14 Documentación de gForce_mex

Lea atentamente la documentación, únicamente se detalla la funcionalidad de gForce_mex.

Despedida

Si llegó hasta aquí cumpliendo todos los pasos, ha configurado correctamente la interfaz gForce_mex para comunicación del brazalete GForce_pro y Matlab. Ahora puede incluir **export_mexMatlab** en todos sus proyectos o directorios de preferencia.