Sistema de visión con ReacTIVision

1. Objetivo de la práctica

Comprender los principios de funcionamiento de los sistemas de visión artificial por reconocimiento de marcadores y distinguir los usos, ventajas y desventajas de ellos.

2. Metas

Para la realización de la práctica se deben cumplir las siguientes metas:

- Realizar las configuraciones básicas del software ReacTIVision para el reconocimiento de los marcadores.
- Usar las funciones básicas y los datos proporcionados por ReacTIVision para extraer la información requerida para el control de robots móviles (coordenadas de posición y orientación de los marcadores).
- Establecer la comunicación entre ReacTIVision y Matlab para enviar la información útil y poder utilizarla para programar el comportamiento de los robots móviles desde el entorno de Matlab y Simulink.
- Comprender el funcionamiento básico del software de Matlab y Simulink para utilizar los datos obtenidos mediante ReacTIVision.

3. Antecedentes

Con la creciente evolución y mejora de los sistemas de procesamiento y de cómputo se ha ido incrementando también el uso de sistemas basados en visión artificial para la instrumentación de robots y ambientes inteligentes.

Las aplicaciones de la visión artificial son muy variadas, desde el reconocimiento de marcadores específicos, como lo que se realizará en esta práctica, hasta el reconocimiento de bordes, formas, objetos e incluso rostros. Para lograr realizar este reconocimiento de manera correcta se utilizan distintos algoritmos que pueden ser englobados finalmente en algún tipo de software especializado que facilite y permita el uso de estas funciones de manera general. Uno de estos programas que permiten el uso de visión artificial es ReacTIVision, que posibilita la identificación de marcadores (*fiducials*).

ReacTIVision es un software de *Open Source* que goza de licencia libre para utilizarla con fines educativos y que fue desarrollado por Martin Kaltenbrunner y Ross Bencia. Este sistema se basa en la identificación de símbolos por medio de un dispositivo de adquisición de imágenes (cámara digital) para realizar su localización con respecto al sistema coordenado del mismo dispositivo. Los símbolos se encuentran codificados utilizando números enteros y la documentación básica provee 255 de ellos, cada uno único por su dibujo representativo denominado *fiducial*, o amiba por su traducción al español. Un ejemplo se muestra en la figura 1.

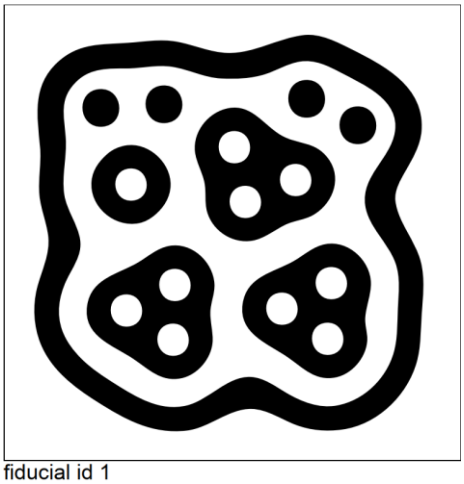


Figura 1. Ejemplo de un fiducial (id=1).

El sistema ReacTIVision, es extremadamente intuitivo y fácil de utilizar debido a que funciona bajo el concepto de “plug and play”, ya que no es necesaria una configuración compleja, sino que basta con conectar la cámara o el sistema de adquisición de imágenes y correr el programa para que comience a funcionar.

Por otro lado, también es importante mencionar que el programa ReacTIVision puede comunicar la información de localización y ángulo del *fiducial* a través del protocolo TUIO para objetos multimedia.

4. Conocimientos previos

Los conocimientos necesarios para la realización de la práctica:

- Conocimientos básicos de computación y programación.
- Conocimientos básicos de Matlab.
- Conocimientos básicos de Simulink.

5. Materiales y Equipo

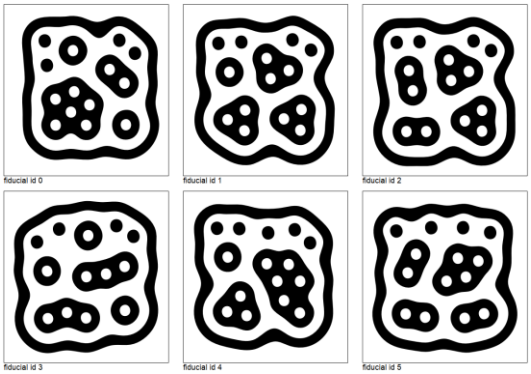
Para la realización de la práctica es necesario contar con lo siguiente:

- Una computadora con Matlab y Simulink. (1)
- Cámara web. (2)
- Conexión a internet para descarga de ReacTIVision y Visual Studio. (3)
- Marcadores (*fiducials*) impresos. (4)

(1)

(2)

(3)



(4)

6. Preparativos previos y recomendaciones

- Es recomendable establecer una carpeta específica en la computadora para el desarrollo de la práctica.
- Se recomienda descargar los archivos requeridos con anticipación.

7. Desarrollo de la práctica

La práctica consta de las siguientes partes:

a) Descarga de ReacTIVision

b) Configuración de la cámara

c) Obtención de la posición y orientación

d) Envío de datos mediante el protocolo UDP

e) Recepción de datos mediante el protocolo UDP desde Simulink.

a) Descarga de ReacTIVision

Tanto el Software de ReacTIVision como los diferentes marcadores compatibles pueden ser descargados directamente desde el sitio oficial del software que se encuentra en la siguiente liga:

http://reactivision.sourceforge.net/

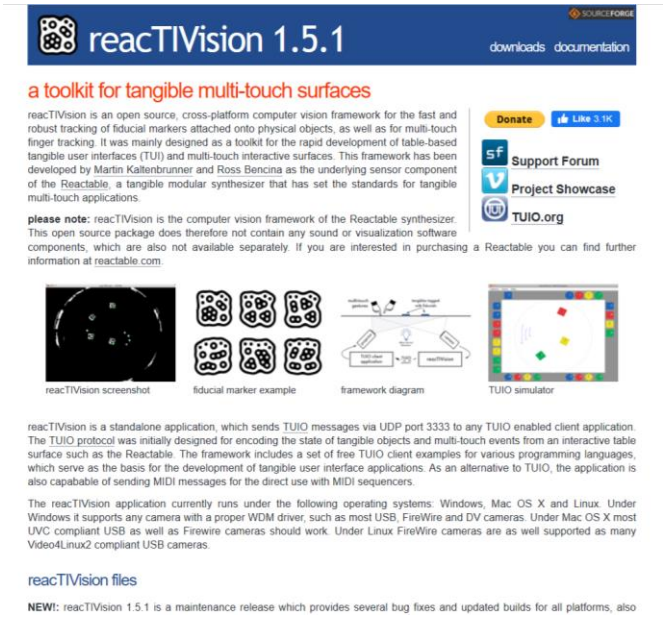


Figura 2. Página de descarga de ReacTIVision.

Debajo de la sección denominada **reacTIVision files -> reacTIVision vision engine** se debe seleccionar y descargar el archivo correspondiente al sistema operativo que se esté utilizando. Una vez descargado, debe descomprimirse la carpeta en una ubicación conocida, la cual debe incluir los siguientes archivos.

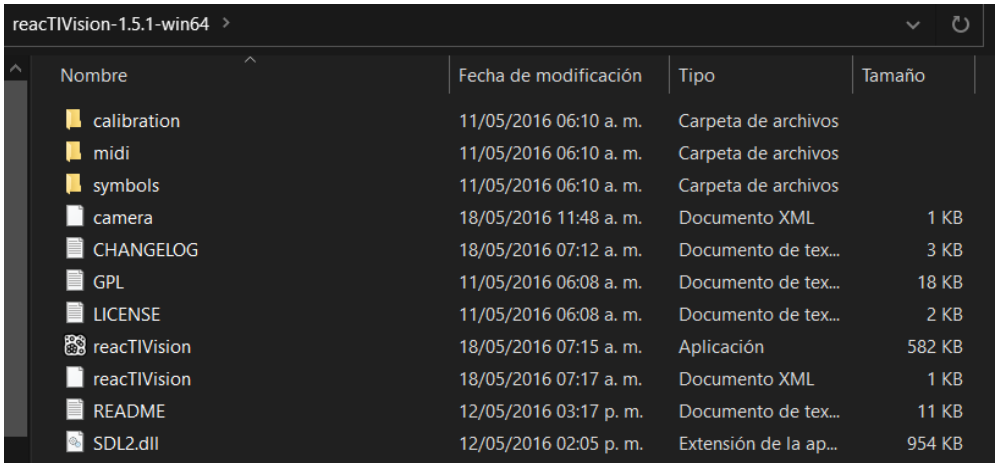


Figura 3. Contenido de la carpeta de ReacTIVision.

En esta práctica se utilizarán principalmente 2 archivos de esta carpeta, por un lado, el archivo camera.xml, en donde se realizarán las configuraciones de la cámara, y reacTIVision.exe, que es el programa ejecutable para correr el sistema de visión.

Además del software de ReacTIVision también se debe descargar un programa cliente que reciba los datos que va a mandar ReacTIVision, para esto se encuentran disponibles en la misma página de descarga diferentes clientes escritos en distintos lenguajes de programación. Estos clientes pueden ser descargados y modificados para utilizar el software en distintas aplicaciones, en este caso se utilizará el cliente en C# que se encuentra bajo la sección reacTIVision TUIO Clients -> TUIO 1.1 -> TUIO-Clients 1.1.5 -> TUIO11_NET-1.1.5.zip.
https://sourceforge.net/projects/reactivision/files/TUIO%201.1/TUIO-Clients%201.1.5/TUIO11_NET-1.1.5.zip/download

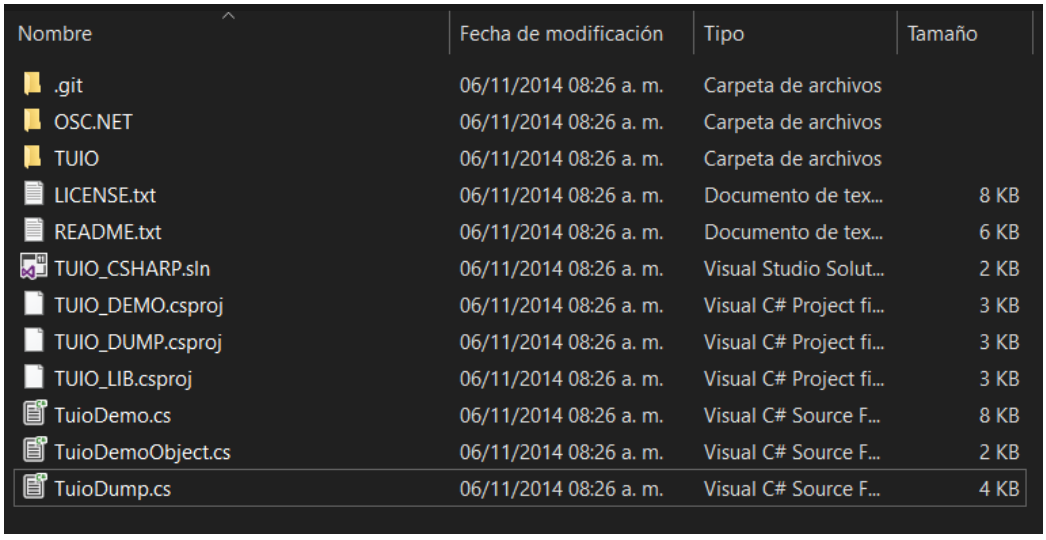
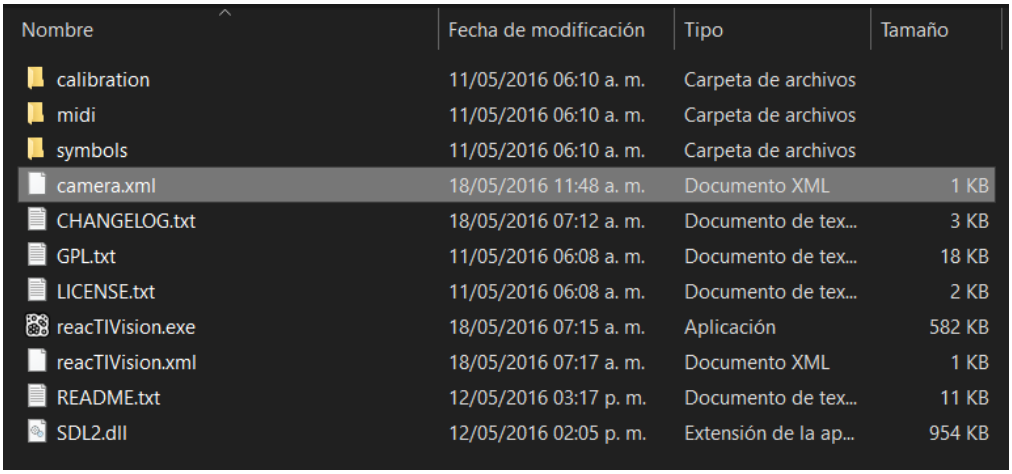


Figura 4. Contenido de la carpeta de TUIO11_NET.

b) Configuración de la cámara

Para poder utilizar el sistema de visión es necesario contar con una cámara conectada a la computadora junto con sus respectivos controladores. La mayoría de las cámaras web suelen funcionar bajo la dinámica “plug and play”, es decir, que basta con conectar la cámara a la computadora para que esta comience a instalar y configurar los controladores necesarios para poder utilizarla. En caso de no contar con una cámara web es posible realizar las pruebas básicas de la práctica utilizando la propia cámara de la laptop.

Una vez que la cámara fue conectada con la computadora se deberá indicar al programa ReacTIVision la cámara que se va a utilizar. Para lograr esto se debe modificar el archivo camera.xml que se encuentra dentro de la carpeta de ReacTIVision descargada anteriormente.



Nombre	Fecha de modificación	Tipo	Tamaño
calibration	11/05/2016 06:10 a. m.	Carpeta de archivos	
midi	11/05/2016 06:10 a. m.	Carpeta de archivos	
symbols	11/05/2016 06:10 a. m.	Carpeta de archivos	
camera.xml	18/05/2016 11:48 a. m.	Documento XML	1 KB
CHANGELOG.txt	18/05/2016 07:12 a. m.	Documento de tex...	3 KB
GPL.txt	11/05/2016 06:08 a. m.	Documento de tex...	18 KB
LICENSE.txt	11/05/2016 06:08 a. m.	Documento de tex...	2 KB
reacTIVision.exe	18/05/2016 07:15 a. m.	Aplicación	582 KB
reacTIVision.xml	18/05/2016 07:17 a. m.	Documento XML	1 KB
README.txt	12/05/2016 03:17 p. m.	Documento de tex...	11 KB
SDL2.dll	12/05/2016 02:05 p. m.	Extensión de la ap...	954 KB

Figura 5. Archivo camera.xml para configuración básica de la cámara.

Para editar el archivo puede hacerse uso de cualquier editor de texto, ya que la modificación que se hará es muy sencilla se recomienda el uso del bloc de notas. El contenido del archivo camera.xml debe contener lo mostrado en la figura 6.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<portvideo>
    <camera id="0">
        <capture width="640" height="480" fps="max" compress="true" />
        <settings brightness="default" contrast="default" gain="default" sh
        <frame width="max" height="max" xoff="0" yoff="0" />
    </camera>
</portvideo>
```

Figura 6. Contenido del archivo camera.xml.

En este archivo se pueden ver y modificar las configuraciones básicas de la cámara. La línea resaltada en color azul en la figura 6 es donde se indica qué cámara es la que va a utilizar el programa. A cada cámara conectada se le asigna un identificador único para poder hacer referencia a ella. Por lo general el id=0 hace referencia a la propia cámara web de la computadora, y a cada cámara adicional conectada se le asignan los números siguientes 1, 2, 3, etc. En esta línea se debe modificar el número del identificador para asignar el que corresponda con la cámara que se desea utilizar. En caso de tener varias cámaras conectadas y no saber que identificador le corresponde a la que se va a utilizar se puede dejar por el momento el id=0, y en caso de que no sea la cámara deseada se puede regresar posteriormente a modificar esta configuración a id=1, 2, 3, etc.

Una vez realizada esta configuración se debe verificar que la imagen de la cámara se puede visualizar correctamente, para ello se debe correr el programa *reacTIVision.exe* que se muestra en la figura 7.

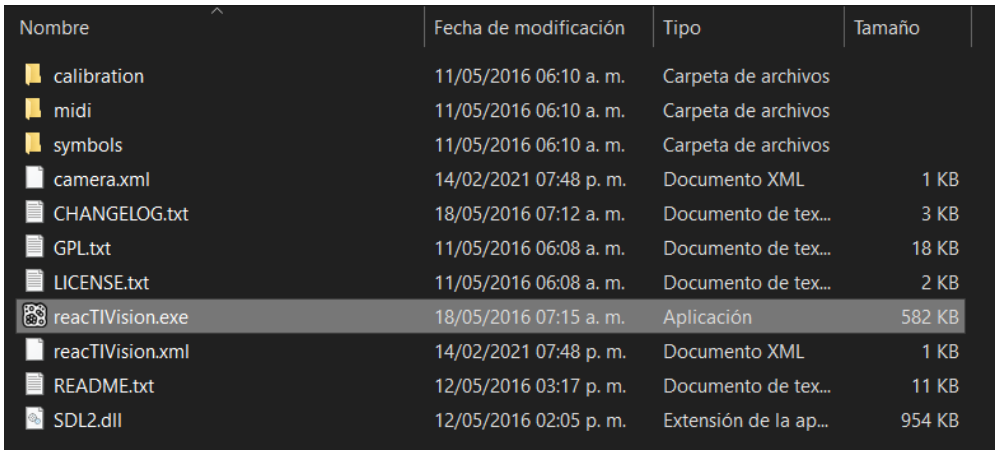


Figura 7. Archivo para ejecutar ReacTIVision.

Al ejecutar dicho archivo deberán abrirse 2 nuevas ventanas, la primera de ellas (figura 8) muestra la información básica del programa y de la cámara que se está utilizando, así como las teclas que se pueden utilizar para cambiar la configuración. La segunda ventana muestra la imagen de la cámara que está siendo utilizada para la detección de los marcadores (Figura 9).

Para poder verificar el funcionamiento del programa se deberán imprimir algunos marcadores (*fiducials*) para realizar las pruebas correspondientes. Estos se pueden ubicar en la misma carpeta que se descargó de la página de ReacTIVision en el archivo:

reacTIVision-1.5.1-win64/symbols/amoeba/legacy.pdf

En este archivo se encuentran 180 distintos marcadores que pueden utilizarse con el programa, para la realización de la práctica bastará con imprimir los primeros 5.

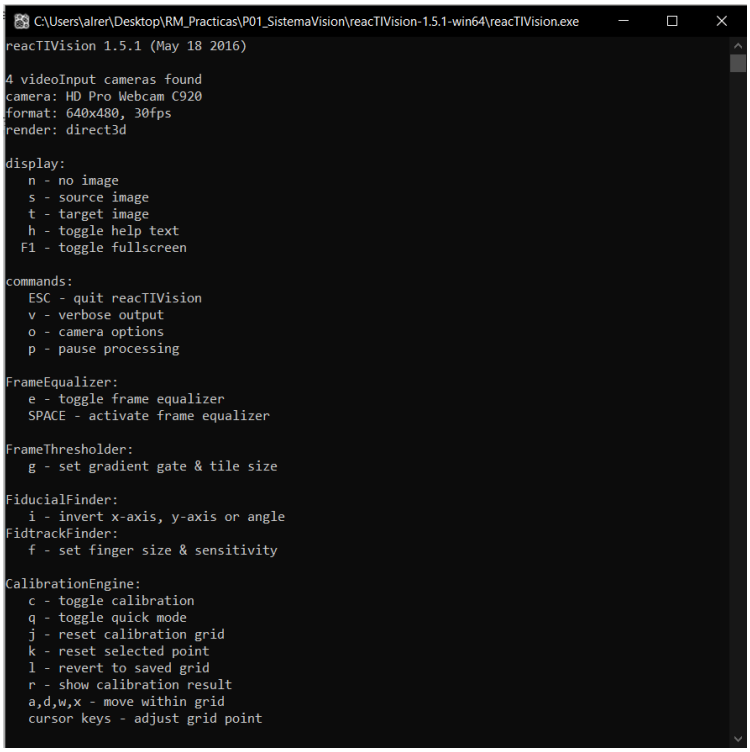


Figura 8. Teclas de configuración de ReacTIVision.

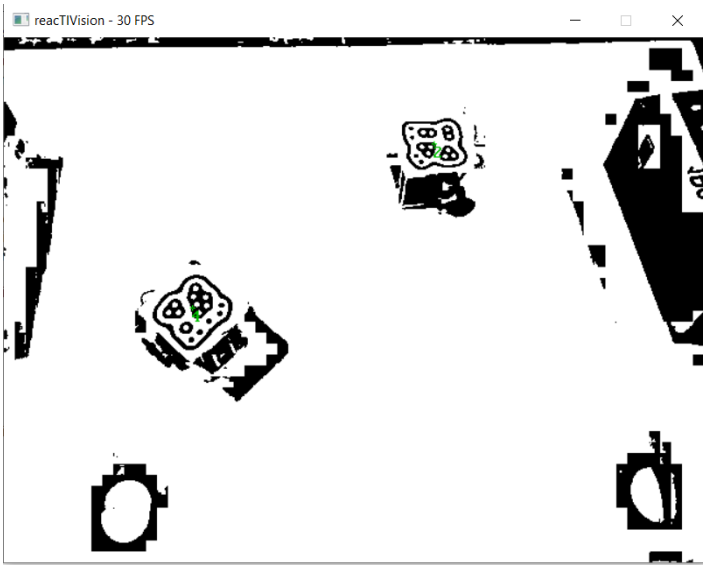


Figura 9. Imagen de cámara utilizada por ReacTIVision.

Poniendo los marcadores dentro del área de visión de la cámara se debe verificar que en la ventana de ReacTIVision aparece una pequeña cruz verde al centro de cada uno de los marcadores, además del número del marcador del que se trata. Esto quiere decir que el marcador es identificado correctamente por el programa, en caso de que no se identifiquen correctamente los marcadores se recomienda modificar las opciones de la cámara (brillo, contraste, exposición, etc.), para esto se debe tener activa la ventana de ReacTIVision y se debe presionar la tecla “o” para acceder a las opciones de la cámara.

Antes de continuar con los pasos siguientes de la práctica se debe realizar lo siguiente:

- Configurar correctamente la cámara que se va a utilizar para identificar los marcadores adecuadamente.
- Modificar las opciones de la cámara para lograr la mejor visualización de cada uno de los marcadores utilizados.
- Probar el funcionamiento de las diferentes opciones de configuración de ReacTIVision.

c) Obtención de la posición y orientación

Para programar y compilar la aplicación que permite el envío de la información de ReacTIVision a Matlab se hará uso del software Visual Studio Community, el cual puede descargarse de manera gratuita desde:

https://visualstudio.microsoft.com/es/vs/community/

Durante el proceso de instalación, para el desarrollo de esta práctica se deberán instalar las cargas de trabajo correspondientes a:

- Desarrollo de escritorio de .NET
- Desarrollo para el escritorio con C++

En la figura 10 se muestran estas cargas de trabajo seleccionadas.

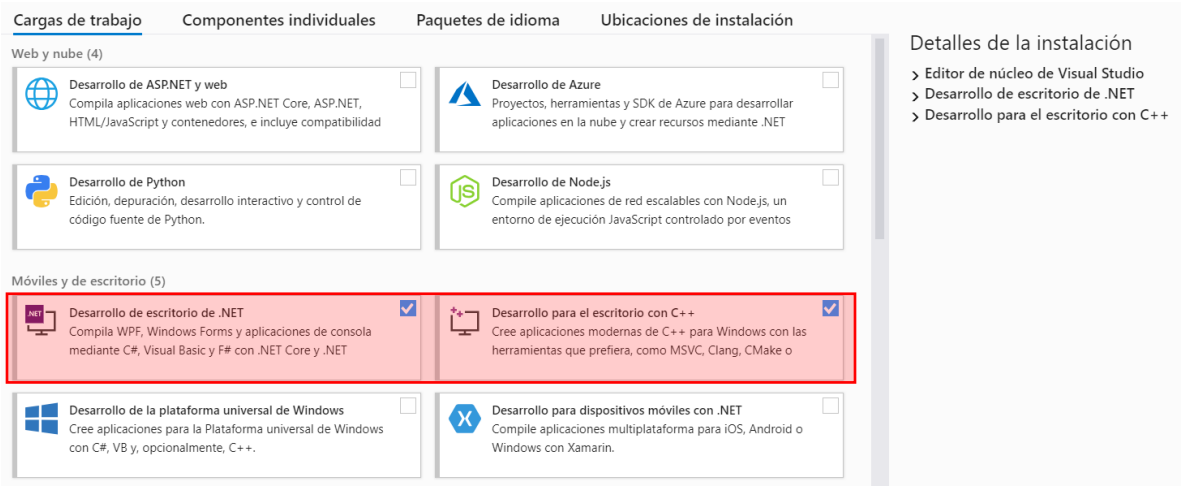


Figura 10. Cargas de trabajo para instalación de Visual Studio.

Una vez instalado Visual Studio Community se debe abrir con ese programa el archivo del proyecto que se encuentra en la carpeta previamente descargada de la página de ReacTIVision: TUIO11_NET/TUIO_CSHARP.sln.

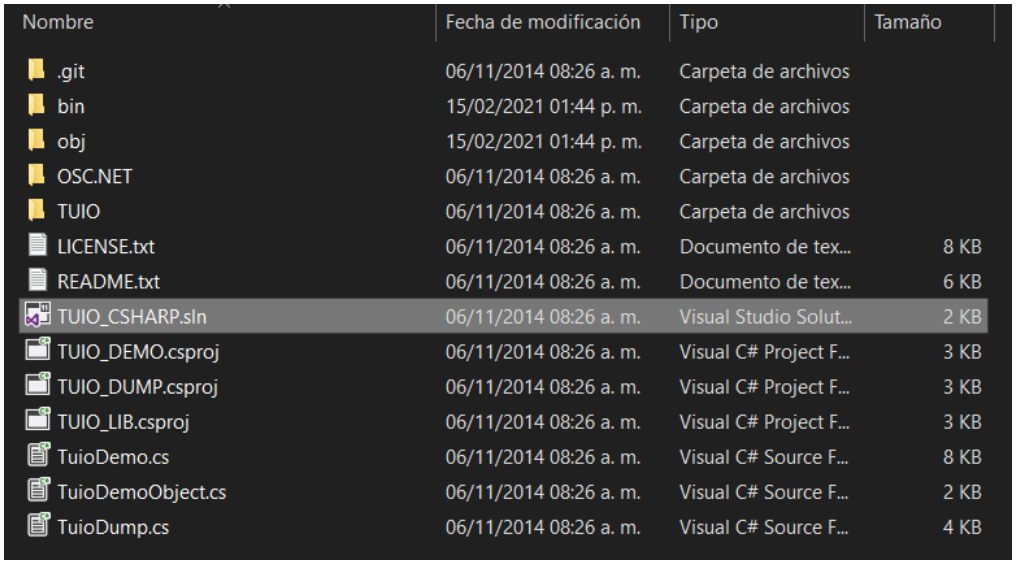


Figura 11. Archivo del proyecto para abrir en Visual Studio.

Tras abrir dicho archivo también se debe verificar que en la misma carpeta se generan dos nuevas carpetas “bin” y “obj”. Con el proyecto abierto en Visual Studio se debe verificar en la parte superior que el proyecto activo sea TUIO_DUMP, ya que éste será el que se utilizará como base (figura 12).

A continuación, se puede dar clic en el botón “Iniciar” que se encuentra del lado derecho. Con esto se compilará y se ejecutará el programa de consola para la obtención de los datos provenientes de ReacTIVision. En principio se verá una ventana de consola con el mensaje “listening to TUIO messages at port 3333”, para poder ver el programa en funcionamiento se debe correr también la aplicación de ReacTIVision (reacTIVision.exe), con lo que comenzará a mostrarse en la ventana de consola la información sobre los marcadores que han sido detectados por la aplicación dentro del área de visión de la cámara.

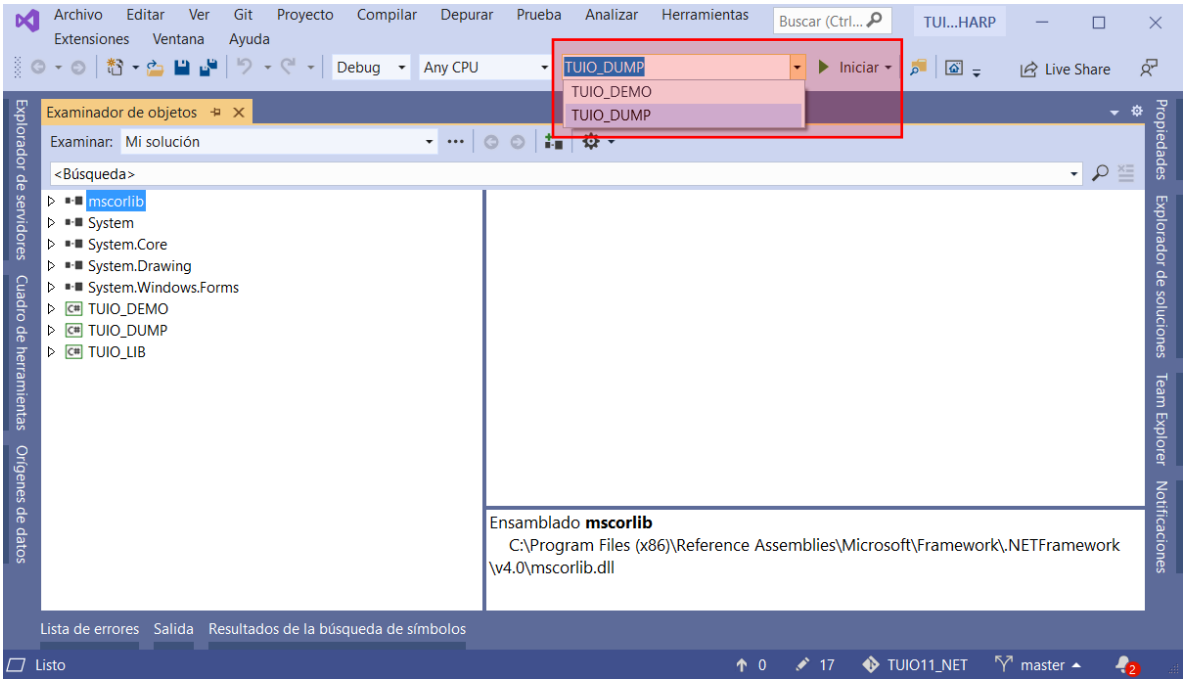


Figura 12. Selección del proyecto TUIO_DUMP en Visual Studio.

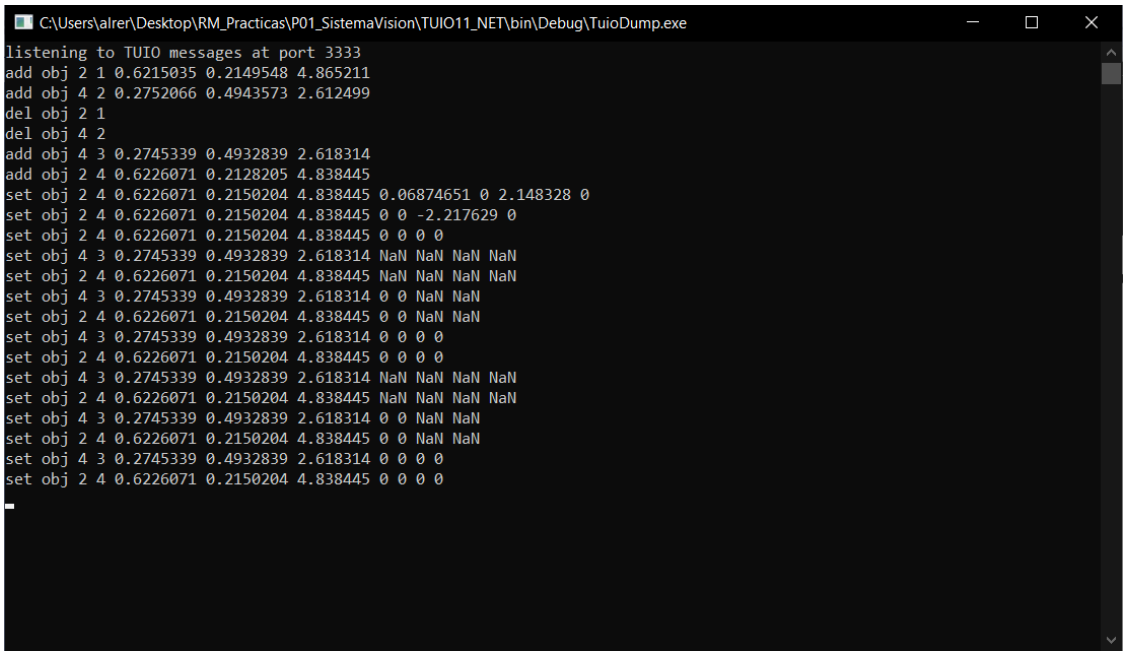


Figura 13. Ejecución de programa TUIO_DUMP con ReacTIVision.

En la figura 13 se muestra la aplicación de consola que resulta de correr el programa TUIO_DUMP y correr el programa ReacTIVision con los marcadores 2 y 4 en el área de visión de la cámara. A continuación, se revisará en mayor detalle el funcionamiento del programa TUIO_DUMP y la manera en que puede modificarse para obtener la información deseada y mandarla a otra aplicación como Matlab.

Para abrir el código que se está ejecutando en Visual Studio se debe elegir la opción de "Abrir archivo", y posteriormente seleccionar el archivo TUIO11_NET/TuioDump.cs.

Este programa está escrito en C#, por lo que para la implementación del programa deberá hacerse referencia a documentación relativa a este lenguaje de programación. Lo primero que se observa es que se genera una clase que implementa la interfaz TuioListener:

public class TuioDump : TuioListener

Al final de dicha clase se tiene la función Main en la que se configura el cliente para la comunicación con ReacTIVision. La estructura básica para inicializar la sesión TUIO es la siguiente:

```
MyApplication app = new MyApplication();

TuioClient client = new TuioClient();

client.addTuioListener(app);

client.start();
```

Se recomienda mantener la configuración que viene por default en la función Main. El resto del código está compuesto por varios métodos que se ejecutan ante diferentes eventos de ReacTIVision; para el uso de marcadores se utilizan los primeros 3 métodos que se encuentran resumidos en la siguiente tabla.

Método	Evento
addTuioObject(TuioObject tobj)	Este método es llamado cuando algún marcador se hace visible en el área de visión de la cámara.
updateTuioObject(TuioObject tobj)	Este método es llamado cuando algún marcador visible se mueve (cambia su posición u orientación).
removeTuioObject(TuioObject tobj)	Este método es llamado cuando algún marcador se quita del área de visión de la cámara o deja de ser visible.

Dentro de estos 3 métodos se puede programar lo que se desea realizar ante estos eventos, desde imprimir en consola la posición del marcador, hasta mandar la información de la posición y orientación a otra aplicación.

También se observa que estos métodos generan un elemento de tipo *TuioObject* llamado *tobj*, lo cual se identifica dentro de los paréntesis del método. A partir de este objeto se puede obtener la información del marcador que ocasionó la llamada a dicho método. Las propiedades básicas que se utilizarán para conocer la posición y orientación del marcador son las siguientes:

Propiedad	Información
tobj.SymbolID	Devuelve el número del marcador en cuestión.
tobj.X	Devuelve la posición X del marcador con respecto al cuadro de visión de la cámara, representado con un valor

	decimal entre 0 y 1, en donde el extremo izquierdo de la imagen tiene coordenada X=0 y el extremo derecho de la imagen tiene coordenada X=1.
tobj.Y	Devuelve la posición Y del marcador con respecto al cuadro de visión de la cámara, representado con un valor decimal entre 0 y 1, en donde el extremo superior de la imagen tiene coordenada Y=0 y el extremo inferior de la imagen tiene coordenada Y=1.
tobj.Angle	Devuelve el ángulo de orientación del marcador en radianes, de tal forma que una posición vertical hacia la parte superior corresponde a un ángulo de 0 rad, y el ángulo se incrementa al girar el marcador en sentido antihorario.

*Entre las opciones de ajuste de ReacTIVision es posible invertir tanto los ejes como el sentido de giro en caso de ser necesario.

*Para más información sobre la implementación del protocolo TUIO se puede referir a la página:

https://www.tuio.org/

Antes de continuar con los pasos siguientes de la práctica:

- Verificar el funcionamiento de los métodos y de las propiedades presentadas aquí en conjunto con el programa de ReacTIVision.
- Modificar el programa TuioDump.cs para que imprima en consola el número de marcador, las coordenadas y el ángulo de los marcadores correctamente cada vez que algún marcador se mueve.
- Efectuar los cambios necesarios para que la información de las coordenadas se obtenga en unidades de longitud reales (metros, centímetros, etc.), en lugar de las coordenadas por default que van de 0 a 1 con respecto al área de visión de la cámara, así como también que se tengan con respecto a los ejes de referencias deseados.

  Sugerencias:
  - Se puede realizar la conversión de unidades entre las coordenadas de la imagen y las distancias reales.
  - Se pueden utilizar marcadores adicionales que sirvan como referencia y calcular las distancias entre coordenadas.

d) Envío de datos mediante el protocolo UDP.

Una forma de pasar la información desde la aplicación a algún otro programa es mediante protocolos de comunicación en red. Uno de los protocolos de este tipo es el protocolo UDP (User Datagram Protocol), el cual permite el envío de datagramas en redes basadas en IP sin la necesidad de que se establezca previamente una conexión. Esto ofrece la ventaja de que permite el envío de datos más rápido en comparación con el protocolo TCP, y permite que se continúe la transmisión aún en el caso de que se presente algún problema con algún paquete o se pierda algún paquete de información. En contraposición, la desventaja de este protocolo de comunicación radica en que no se tiene una

confirmación ni garantía de que los paquetes de datos enviados fueron entregados completos y de manera correcta, por lo que esto deberá ser considerado al implementar este protocolo.

Para generar un socket para comunicación mediante el protocolo UDP en C# se utiliza la siguiente estructura:

```
Socket socket = new Socket(AddressFamily.InterNetwork, SocketType.Dgram,
ProtocolType.Udp);

IPAddress ipaddress = IPAddress.Parse(direccion);
IPEndPoint endpoint = new IPEndPoint(ipaddress, puerto);
byte[] buffer = Encoding.ASCII.GetBytes(mensaje);

socket.SendTo(buffer, endpoint);
```

En este código es necesario establecer previamente la dirección IP del dispositivo al que se va a enviar el paquete (***direccion***), el puerto dentro de esa dirección que se utilizará para la comunicación (***puerto***), y el mensaje que se desea enviar (***mensaje***), en este caso se programó para que sea un dato de tipo *string*.

La dirección debe corresponder con la dirección IP del dispositivo en el que se va a recibir la información, para esta práctica supondremos que se va a mandar a un programa de Matlab corriendo en la misma máquina que ReacTIVision, por lo que la dirección IP será la correspondiente al localhost, es decir, al mismo dispositivo (127.0.0.1). Con respecto al puerto, se recomienda utilizar alguno entre 49152 y 65535, ya que estos son puertos dinámicos que no son registrados por la IANA para aplicaciones específicas.

```
private readonly string direccion = "127.0.0.1";
private readonly int puerto = 58624;
string g = "Mensaje";
```

Un punto importante es que para poder implementar el código anterior para el envío de datos mediante el protocolo UDP es necesario incluir las librerías correspondientes que permiten usar estos comandos. Por ello, se deben incluir las siguientes líneas al principio del código (fuera de la clase TuioDump).

```
using System.Net;
using System.Net.Sockets;
using System.Text;
```

Al programar esto dentro la aplicación TUIO_DUMP es posible mandar datos a al dispositivo con la dirección IP especificada y al puerto seleccionado.

e) Recepción de datos mediante protocolo UDP desde Simulink

A continuación, se procederá a realizar el programa en Simulink que recibe los datos enviados mediante el protocolo UDP. Como primer paso se debe realizar la instalación de Simulink Desktop Real-Time, ya que esto es necesario para poder interactuar con dispositivos en tiempo real, en este caso se utilizará para realizar la comunicación mediante el protocolo UDP.

Para instalarlo se debe introducir en la ventana de comandos de Matlab la siguiente instrucción:

```
sldrtkernel -install
```

En seguida se deberá confirmar para continuar con la instalación, en caso de tener ya instalado esto o tener una versión diferente se mostrará un mensaje indicando la situación.

Una vez finalizada la instalación se deberá verificar que la instalación fue completada de manera correcta mediante el comando:

```
rtwho
```

Con esto se deberá mostrar la versión de Simulink Desktop Real-Time que se tiene instalada. Para más información sobre este *kernel* se puede hacer referencia a la documentación de Matlab en el siguiente enlace:

https://la.mathworks.com/help/sldrt/ug/real-time-windows-target-kernel.html

Una vez finalizada la instalación de Simulink Desktop Real-Time se procederá a crear un modelo nuevo de Simulink, este se puede iniciar introduciendo el comando ***simulink*** en la ventana de comandos de Matlab y seleccionando la opción para crear un nuevo modelo en blanco.

Con la instalación que se realizó de Simulink Desktop Real-Time se incluyen también los bloques necesarios para recibir y enviar datos mediante el mismo protocolo UDP que se está utilizando en el programa de C#. Para recibir los datos en Simulink se utilizará el bloque “Packet Input”, el cual se encuentra en la librería de bloques de Simulink en la sección llamada “Simulink Desktop Real-Time”. Así mismo, en la misma sección también se encuentra el bloque que puede ser utilizado para enviar datos mediante este protocolo, “Packet Output”.

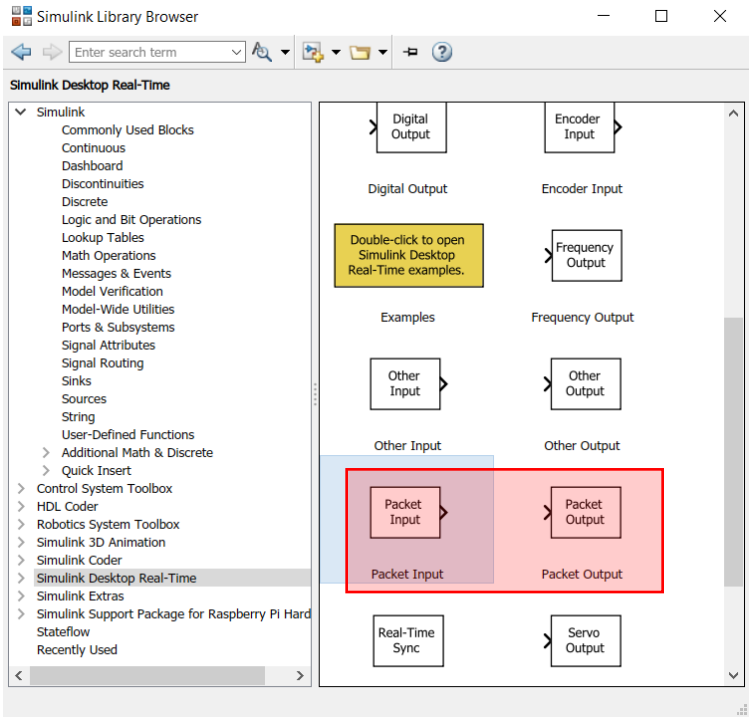


Figura 14. Bloques para el envío y recepción de datos en Simulink.

Ya que nuestro objetivo es recibir los datos que se envían desde el otro programa necesitamos utilizar el bloque “Packet Input”, éste puede arrastrarse con el ratón para colocarlo dentro del modelo en blanco de Simulink. Después de colocar el bloque, se debe configurar haciendo doble clic sobre él, lo cual deberá desplegar una ventana con opciones de configuración como la que se presenta en la figura 15.

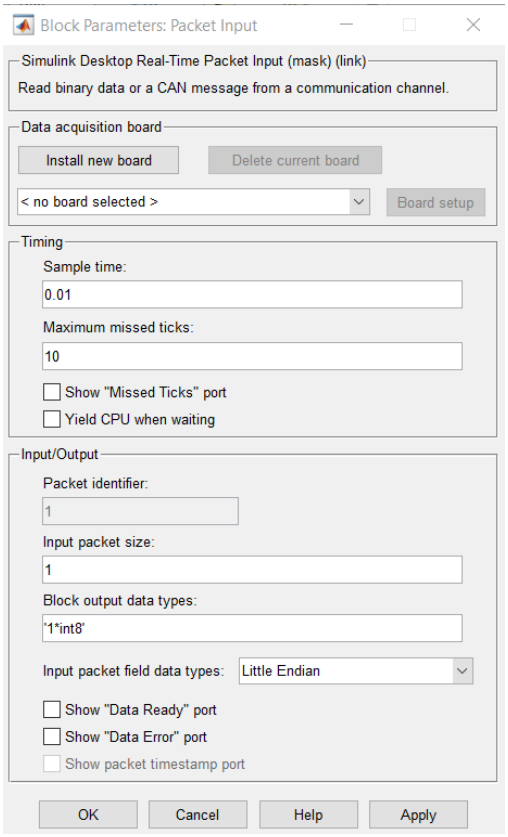


Figura 15. Configuración del bloque Packet Input en Simulink.

Aquí se observan varias opciones de configuración:

- Sample time: es el tiempo de muestreo que se considerará entre los paquetes.
- Maximum missed Ticks: es la cantidad de paquetes que pueden no recibirse antes de que se detecte como un error. Recordemos que el protocolo UDP realiza el envío de datos sin verificación, por lo que existe la posibilidad de perder paquetes de datos. Al configurar esta opción como “inf” se indica que el programa continúe sin importar la cantidad de paquetes de datos que no sean recibidos.

- Input packet size: indica la cantidad de datos que van a leerse en cada paquete, en este caso puede ser la cantidad de símbolos que contiene la cadena que se quiere recibir.
- Block output data types: indica los tipos de datos que se tendrán a la salida del bloque. Por ejemplo, en este caso suponemos que queremos recibir una cadena que tiene un número de un solo dígito y queremos que salga como entero, por lo tanto, se introduce *'1*int8'*. Si quisiéramos leer una cadena de 7 caracteres se puede expresar como *'1*int8'*. Con esto, se tendrá la cantidad de salidas especificadas, una por cada carácter de la cadena que se leyó.

  Importante: con esta configuración la salida será el número correspondiente al código ASCII del símbolo. Por ejemplo, si se recibe la cadena '1', la salida del bloque será el número 49, ya que este valor es el que corresponde al símbolo '1' de acuerdo con el código ASCII. Una forma de convertirlo al número correcto sería restando 48 del número que sale del bloque. Una tabla con el código ASCII puede encontrarse en la siguiente liga: https://elcodigoascii.com.ar/.

Para configurar el protocolo, la dirección y el puerto con el que se realizará la comunicación se debe dar clic en el botón "Install new board", y a continuación, seleccionar la opción "Standard Devices" → "UDP Protocol", con lo que se abrirá una nueva venta (figura 17). En esta nueva ventana se deberán introducir los datos correspondientes a la dirección IP y el puerto que se están utilizando para la comunicación, estos datos deben coincidir con los que se están utilizando para el envío del paquete desde el programa de C#.

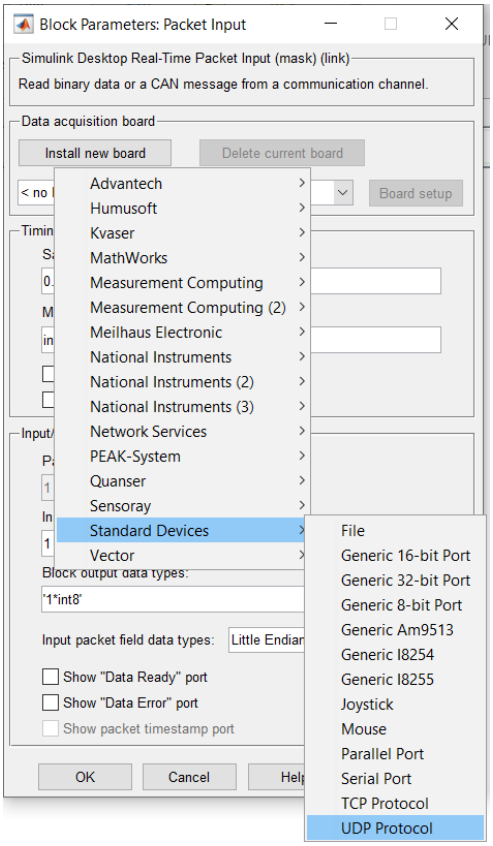


Figura 16. Nueva tarjeta para comunicación mediante el protocolo UDP.

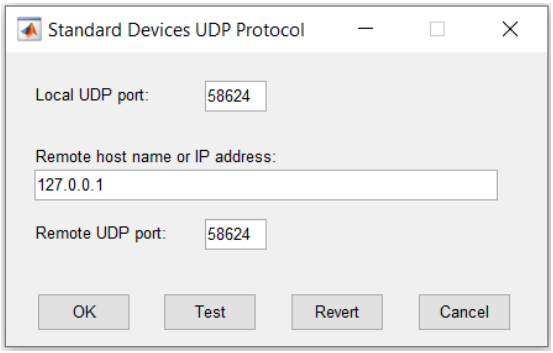


Figura 17. Configuración de la dirección IP y el puerto para la comunicación.

Tras haber introducido los datos es recomendable dar clic en el botón “Test”, con la finalidad de que el puerto y dirección IP estén disponibles para la conexión. De ser así se mostrará el mensaje de confirmación que se muestra en la figura 18.

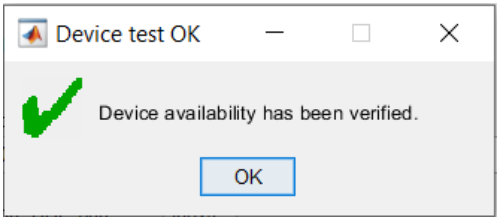


Figura 18. Verificación de la disponibilidad de la dirección IP y puerto.

Finalmente, se deben confirmar los cambios y guardar la tarjeta para la comunicación.

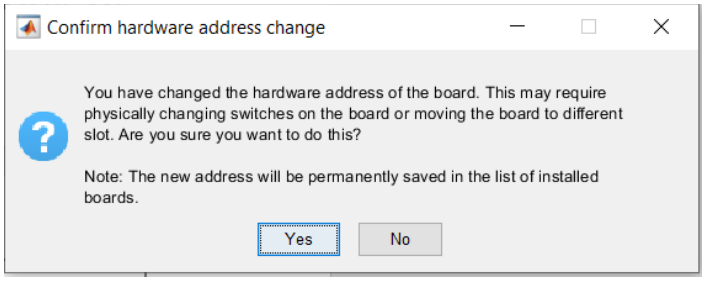


Figura 19. Confirmación de cambios y guardar la tarjeta de comunicación.

*Es importante recordar que tras hacer modificaciones en las configuraciones se debe seleccionar “Apply” o “OK” en la ventana correspondiente para que los cambios tengan efecto.

Regresando al modelo en donde se colocó el bloque, también será necesario crear un elemento que nos permita visualizar los datos que se están obteniendo. Una posibilidad es utilizando el elemento “Display”, ubicado dentro de la librería de bloques de Simulink en la sección “Simulink” → “Sinks”, el cual puede conectarse a la salida del bloque “Packet Input”.

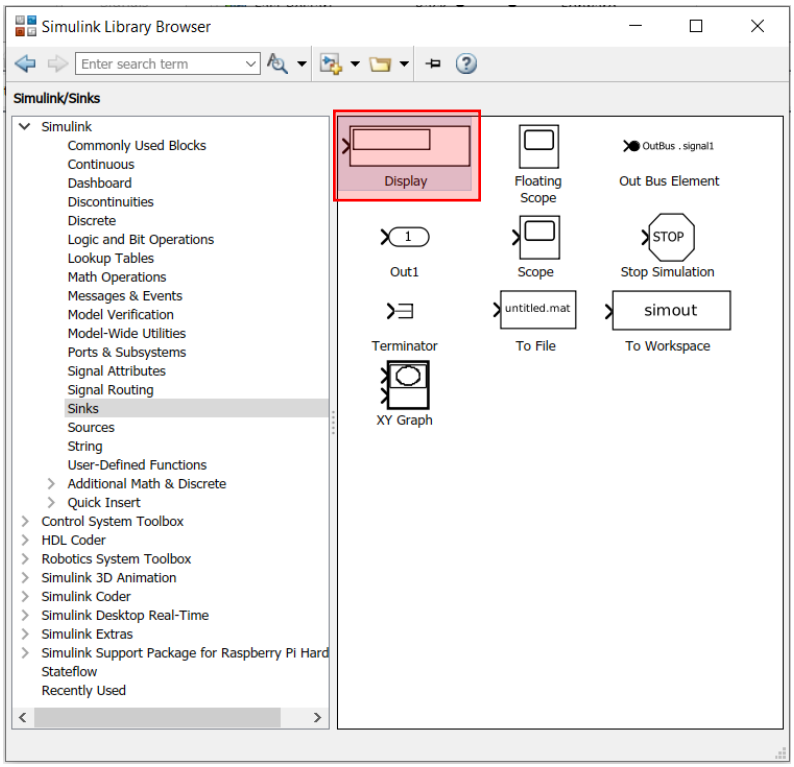


Figura 20. Elemento “Display” para visualizar datos.

Para probar la comunicación es necesario guardar el modelo y correrlo haciendo clic en el botón “Run” ubicado en la parte superior de la ventana del modelo de Simulink (figura 21). También es necesario indicar el tiempo en segundos que durará la simulación en el recuadro “Stop Time” (figura 21), inicialmente viene definido como 10.0 lo que quiere decir que la simulación se detendrá después de 10 segundos. Para mantener la simulación corriendo indefinidamente se puede introducir como valor “inf”.

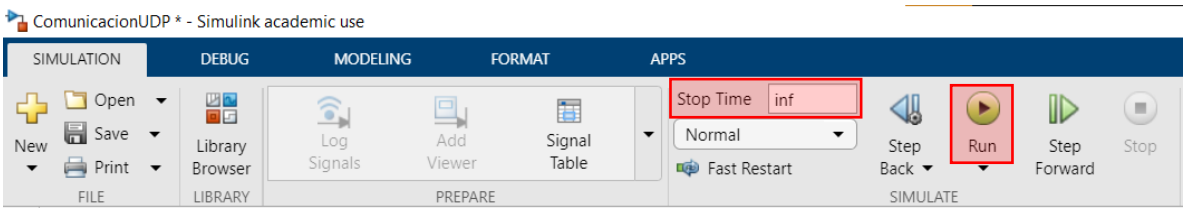


Figura 21. Botón para correr y especificar el tiempo de simulación.

Una vez corriendo el modelo de Simulink se deberá correr también el programa de C# en Visual estudio ya con la configuración para el envío de datos mediante el socket con protocolo UDP. Si todo se realizó de manera correcta se deberá visualizar en el Display en Simulink el dato enviado.

Para este ejemplo se envió como mensaje, desde el programa en C#, la cadena "1". Como se observa en la figura 22, el dato leído desde Matlab es el entero 49, que corresponde al Código ASCII del número "1".

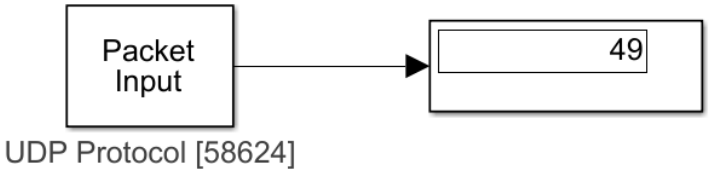


Figura 22. Recepción de datos en Simulink.

Con esto ya se tienen las herramientas básicas necesarias para enviar los datos desde ReacTIVision hasta Simulink y poder hacer uso de esta información en Simulink para desarrollar programas más complejos de control de robots móviles.

Realiza las siguientes actividades para concluir con la práctica:

- Prueba la comunicación mediante el protocolo UDP entre el programa en C# y Simulink enviando un solo dígito como se mostró en el desarrollo de la práctica.
- Realiza las modificaciones necesarias en el programa de C# para que se envíen como una cadena de texto los datos correspondientes al número del marcador identificado, las coordenadas y la orientación del marcador.
- Desarrolla en Simulink un programa que reciba esta cadena, separe la información (número de marcador, coordenada X, coordenada Y, ángulo) y la muestre de forma numérica.

Recomendaciones:

- Dado que el protocolo UDP no tiene verificación en la recepción de los paquetes se recomienda enviar la cadena con un símbolo adicional que permita identificar en dónde inicia una cadena y dónde inicia la siguiente. Además, para facilitar la separación de los datos se puede mandar la misma cantidad de dígitos para cada dato, completando con ceros donde haga falta.
  Por ejemplo, si se quiere mandar id=1, x=127, y=34 y ang=65, utilizando como identificador de inicio de cadena el carácter “M”, la cadena podría quedar de la siguiente manera:
  “**M**1**127**034**065**”
- Para el programa de Simulink se recomienda utilizar un bloque “MATLAB Function” (figura 23), en donde se puede programar lo necesario para identificar el inicio de la cadena con el identificador, la separación de la cadena en los dígitos que corresponden a cada dato y la conversión a valores numéricos utilizando el código ASCII.
  Más información sobre “MATLAB function”:
  https://la.mathworks.com/help/simulink/slref/matlabfunction.html

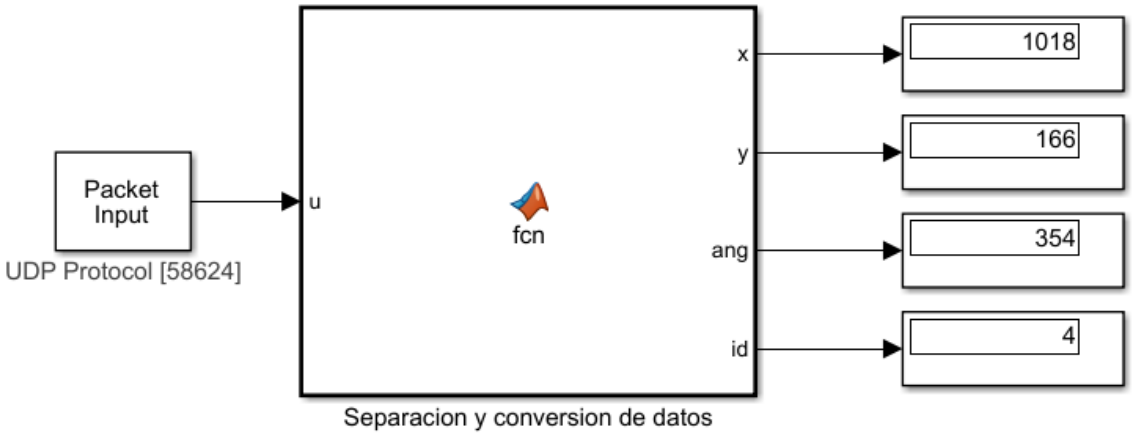


Figura 23. Propuesta de implementación en Simulink.

8. Resultados

Como resultados de esta práctica se deben entregar los programas realizados para lograr la obtención de la posición y orientación de los marcadores, así como un video en el que se explica y se demuestra el funcionamiento.

9. Conclusiones

La obtención de la posición y orientación de un robot es parte fundamental para lograr el comportamiento deseado en robótica móvil. El uso de cámaras y sistemas de visión con marcadores brindan la posibilidad de medir esto bajo ciertas condiciones, sin la necesidad de utilizar sensores más complejos o caros.

En esta práctica se realizó una introducción al software de ReacTIVision, con el cual es posible identificar marcadores dentro del área de visión de una cámara y obtener la información relativa a su posición y orientación. Esta información será de gran utilidad para poder programar y controlar robots móviles en prácticas subsecuentes. Así mismo, se introdujo la comunicación mediante el protocolo UDP que es una buena opción para enviar los datos hacia otro tipo de programas que facilitan la programación en el ámbito de la robótica móvil como es el caso de Matlab/Simulink.

Bibliografía

- Documentación de ReacTIVision.
- Documentación de Matlab/Simulink.