Tutorial Image Acquisition Toolbox

Sistemas de Visión Artificial

Universidad de Alcalá

Index

1 Install Your Image Acquisition Device 3
2 Installing the Support Packages for Image Acquisition Toolbox Adaptors 3
3 The Image Acquisition Tool Desktop 4
Opening the Tool 4
Parts of the Desktop 5
4 Basic Image Acquisition Procedure 6
Step 1: Install and configure your image acquisition device 7
Step 2: Retrieve hardware information 7
Step 3: Create a video input object 10
Step 4: Preview the video stream (Optional) 11
Step 5: Configure image acquisition object properties (Optional) 12
Step 6: Acquire Image Data 15
Step 7: Clean Up 17
References 18

In this practice we will follow the instructions provided in the Matlab Documentation for the Image Acquisition Toolbox to setup our Image Acquisition Hardware and to use the Image Acquisition Toolbox software to create an example: a simple motion detection application. It will detect movement in a scene by performing a pixel-to-pixel comparison in pairs of incoming image frames. If nothing moves in the scene, pixel values remain the same in each frame. When something moves in the image, the application displays the pixels that have changed values.

As well as **Matlab**, with the **Image Acquisition Toolbox** installed, an **image acquisition device** must be connected to the system. The device can be a professional grade image acquisition device, such as a frame grabber, or a generic Microsoft® Windows image acquisition device, such as a webcam.

1 Install Your Image Acquisition Device

The first step is **installing the Image Acquisition Device**. Follow the setup instructions that come with it. Setup typically involves:

- Installing the frame grabber board in your computer.
- Installing any software drivers required by the device. These are supplied by the device vendor.
- Connecting a camera to a connector on the frame grabber board.
- Verifying that the camera is working properly by running the application software that came with the camera and viewing a live video stream.

Generic Windows image acquisition devices, such as webcams and digital video camcorders, typically do not require the installation of a frame grabber board. You connect these devices directly to your computer via a USB or FireWire port.

After installing and configuring your image acquisition hardware, **start MATLAB** on your computer by double-clicking the icon on your desktop. You do not need to perform any special configuration of MATLAB to perform image acquisition.

2 Installing the Support Packages for Image Acquisition Toolbox Adaptors

https://es.mathworks.com/help/imaq/installing-the-support-packages-for-image-acquisition-toolbox-adaptors.html

The fists step before starting to work with the image acquisition device will be to install its Support Package with its functionality. Starting with R2014a, each adaptor is available separately through MATLAB Add-Ons, and **you must install the appropriate support packages to use the toolbox with your hardware**.

To install a support package:

1. On the MATLAB **Home tab**, in the **Environment section,** click **Add-Ons > Get Hardware Support Packages**.
2. In the Add-On Explorer, scroll to the **Hardware Support Packages** section, and click **show all** to find your support package.
3. You can refine the list by selecting **Imaging/Cameras** in the **Refine by Hardware Type** section on the left side of the Explorer.
4. **Select your adaptor**, for example “Image Acquisition Toolbox Support Package for OS Generic Video Interface”, “MATLAB Support Package for USB Webcams” or “Image Acquisition Toolbox Support Package for GigE Vision® Hardware”, from the list. For any cameras that use the Windows Video (winvideo), Macintosh Video (macvideo), or Linux Video (linuxvideo) adaptors, use the support package called **Image Acquisition Toolbox Support Package for OS Generic Video Interface**. The correct files will be installed, depending on your operating system.

If you use multiple adaptors, you need to install the support package for each one you use.

3 The Image Acquisition Tool Desktop

Opening the Tool

Image Acquisition Toolbox™ functionality is available in a desktop application. You connect directly to your hardware in the tool and can preview and acquire image data. You can log the data to MATLAB® in several formats, and also generate a VideoWriter file, right from the tool.

The Image Acquisition Tool provides a desktop environment that integrates a preview/acquisition area with Acquisition Parameters so that you can change settings and see the changes dynamically applied to your image data.

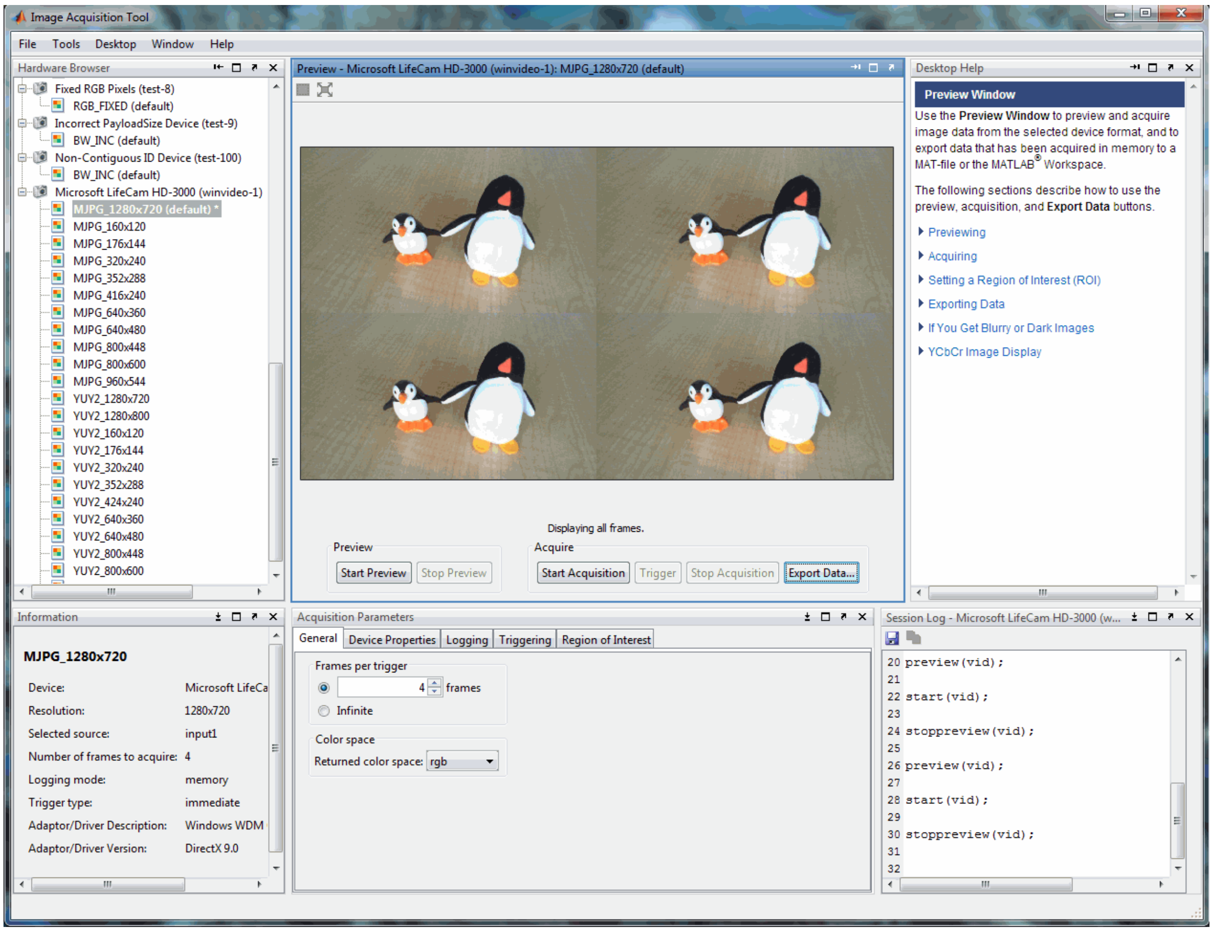
To open the Image Acquisition Tool, **do one of the following**:

- Type **imaqtool** at the MATLAB command line.
- Select **Image Acquisition** on the **Apps** tab in MATLAB.

The right pane in the tool is the **Desktop Help** pane. As you work in the tool the Help will provide information for the part of the interface that you are working in. If the **Desktop Help** is closed, you can open it be selecting **Desktop > Desktop Help**.

Parts of the Desktop

https://es.mathworks.com/help/imaq/the-image-acquisition-tool-desktop.html



Use the **Preview Window** to preview and acquire image data from the selected device format, and to export data that has been acquired in memory to a MAT-file or the MATLAB® Workspace.

The following sections describe how to use the preview, acquisition, and **Export Data** buttons.

MJPG_1280x720

Device:	Microsoft LifeCa
Resolution:	1280x720
Selected source:	input1
Number of frames to acquire:	4
Logging mode:	memory
Trigger type:	immediate
Adaptor/Driver Description:	Windows WDM
Adaptor/Driver Version:	DirectX 9.0

Frames per trigger
(●) 4 frames
( ) Infinite

Color space
Returned color space: rgb

```
preview(vid);

start(vid);

stoppreview(vid);

preview(vid);

start(vid);

stoppreview(vid);

```

Fig. 1. Image Acquisition Tool Window

The Image Acquisition Tool has the following panes.

- **Hardware Browser** – Shows the image acquisition devices currently connected to your system. Each device is a separate node in the browser. All of the formats the device supports are listed under the device. Each device's default format is indicated in parentheses. Select the device format or camera file you want to use for the acquisition. When the format is selected, you can then set acquisition parameters and preview your data. See Selecting Your Device in Image Acquisition Toolfor more information about using the **Hardware Browser**.

- **Preview window** – Use to preview and acquire image data from the selected device format, and to export data that has been acquired in memory to a MAT-file, the MATLAB Workspace, VideoWriter, or to tools provided by the Image Processing Toolbox™ software. See Previewing and Acquiring Data in Image Acquisition Tool for more information about using the **Preview window**.

- **Acquisition Parameters** – Use these tabs to set up general acquisition parameters, such as frames per trigger and color space, device-specific properties, logging options, triggering options, and region of interest. Settings you make on any tab will apply to the currently selected device format in the **Hardware Browser**. See Setting Acquisition Parameters in Image Acquisition Tool for more information about using the **Acquisition Parameters**. Also see the Help for each tab while using the tool for more details. When you click any tab, the help for that tab will appear in the **Desktop Help** pane.
- **Information pane** – Displays a summary of information about the selected node in the **Hardware Browser**.
- **Session Log** – Displays a dynamically generated log of the commands that correspond to actions taken in the tool. You can save the log to a MATLAB code file or copy it.
- **Desktop Help** – Displays Help for the pane of the desktop that has focus. Click inside a pane for help on that area of the tool. For the **Acquisition Parameters** pane, click each tab to display information about the settings for that tab. If the **Desktop Help** is closed, you can open it by selecting **Desktop > Desktop Help**.

Get familiar with the graphical user interface. Check the available color spaces, resolutions, change the configuration and preview (using the Preview button). Explore the options. You will see that the code will appear in the session log pane. You can copy it and save it in a file, or paste it in the Matlab Command Window to execute it (AFTER closing the Image Acquisition Tool or a "device already in use" error will appear).

4 Basic Image Acquisition Procedure

https://es.mathworks.com/help/imaq/basic-image-acquisition-procedure.html.

This section illustrates the basic steps required to **create an image acquisition application** by implementing a simple motion detection application. The application **detects movement** in a scene by performing a pixel-to-pixel comparison in pairs of incoming image frames. If nothing moves in the scene, pixel values remain the same in each frame. When something moves in the image, the application displays the pixels that have changed values.

To use the Image Acquisition Toolbox software to acquire image data, you must perform the following basic steps:

Step	Description
Step 1:	Install and configure your image acquisition device.

Step	Description
Step 2:	Retrieve information that uniquely identifies your image acquisition device to the Image Acquisition Toolbox software
Step 3:	Create a video input object
Step 4:	Preview the video stream (Optional)
Step 5:	Configure image acquisition object properties (Optional)
Step 6:	Acquire image data
Step 7:	Clean up

Step 1: Install and configure your image acquisition device

See previous section: "Install Your Image Acquisition Device".

Step 2: Retrieve hardware information

In this step, you get several pieces of information that the toolbox needs to uniquely identify the image acquisition device you want to access. You use this information when you create an image acquisition object, described in Step 3: Create a Video Input Object.

The following table lists this information. You use the *imaqhwinfo* function to retrieve each item.

Device Information	Description
Adaptor name	An adaptor is the software that the toolbox uses to communicate with an image acquisition device via its device driver. The toolbox includes adaptors for certain vendors of image acquisition equipment and for particular classes of image acquisition devices. See next section "Determining the Adaptor Name" for more information.
Device ID	The device ID is a number that the adaptor assigns to uniquely identify each image acquisition device with which it can communicate. See next section "Determining the Device ID" for more information. Note: Specifying the device ID is optional; the toolbox uses the first available device ID as the default.
Video format	The video format specifies the image resolution (width and height) and other aspects of the video stream. Image acquisition devices typically support multiple video formats. See next section

	"Determining the Supported Video Formats" for more information. Note: Specifying the video format is optional; the toolbox uses one of the supported formats as the default.

Determining the Adaptor Name

To determine the name of the adaptor, enter the ***imaqhwinfo*** function at the MATLAB prompt without any arguments.

```
>> imaqhwinfo
ans =
   InstalledAdaptors: {'dcam'  'winvideo'} → seleccionar winvideo
      MATLABVersion: '7.4 (R2007a)'
        ToolboxName: 'Image Acquisition Toolbox'
     ToolboxVersion: '2.1 (R2007a)'1
```

In the data returned by ***imaqhwinfo***, the InstalledAdaptors field lists the adaptors that are available on your computer. In this example, ***imaqhwinfo*** found two adaptors available on the computer: ***'dcam'*** and ***'winvideo'***. The listing on your computer might contain only one adaptor name. Select the adaptor name that provides access to your image acquisition device. For more information, see Determining the Device Adaptor Name.

Determining the Device ID

To find the device ID of a particular image acquisition device, enter the ***imaqhwinfo*** function at the MATLAB prompt, specifying the name of the adaptor as the only argument. (You found the adaptor name in the first call to ***imaqhwinfo***, described in Determining the Adaptor Name.) In the data returned, the DeviceIDs field is a cell array containing the device IDs of all the devices accessible through the specified adaptor.

Note: This example uses the DCAM adaptor. You should substitute the name of the adaptor you would like to use.

```
>> info = imaqhwinfo('dcam')

info =

     AdaptorDllName: [1x77 char]
   AdaptorDllVersion: '2.1 (R2007a)'
        AdaptorName: 'dcam'
          DeviceIDs: {[1]}
         DeviceInfo: [1x1 struct]
```

Determining the Supported Video Formats

To determine which video formats an image acquisition device supports, look in the ***DeviceInfo*** field of the data returned by ***imaqhwinfo***. The ***DeviceInfo*** field is a structure array where each structure provides information about a particular device. To view the device information for a particular device, you can use the device ID as a reference into the structure array. Alternatively, you can view the information for a particular device by calling the ***imaqhwinfo*** function, specifying the adaptor name and device ID as arguments.

To get the list of the video formats supported by a device, look at ***SupportedFormats*** field in the device information structure. The ***SupportedFormats*** field is a cell array of strings where each string is the name of a video format supported by the device. For more information, see Determining Supported Video Formats.

```
>> dev_info = imaqhwinfo('dcam',1)

dev_info =

       DefaultFormat: 'F7_Y8_1024x768'
  DeviceFileSupported: 0
          DeviceName: 'XCD-X700  1.05'
            DeviceID: 1
 VideoInputConstructor: 'videoinput('dcam', 1)'
     SupportedFormats: {'F7_Y8_1024x768'  'Y8_1024x768'}
```

A different example:

```
>> formato2=imaqhwinfo('winvideo',2)

formato2 =

  struct with fields:
          DefaultFormat: 'RGB24_640x480'
      DeviceFileSupported: 0
             DeviceName: 'Logitech HD Webcam C270'
               DeviceID: 2
   VideoInputConstructor: 'videoinput('winvideo', 2)'
  VideoDeviceConstructor: 'imaq.VideoDevice('winvideo', 2)'
        SupportedFormats: {1×38 cell}   → it only shows the type of the variable. If we want to see all of
them:

>> formato2.SupportedFormats
ans =
  1×38 cell array
  Columns 1 through 5
   {'I420_1024x576'}  {'I420_1184x656'}  {'I420_1280x720'}  {'I420_1280x960'}  {'I420_160x120'}
  Columns 6 through 10
   {'I420_176x144'}  {'I420_320x176'}  {'I420_320x240'}  {'I420_352x288'}  {'I420_432x240'}
  Columns 11 through 15
   {'I420_544x288'}  {'I420_640x360'}  {'I420_640x480'}  {'I420_752x416'}  {'I420_800x448'}
  Columns 16 through 20
   {'I420_800x600'}  {'I420_864x480'}  {'I420_960x544'}  {'I420_960x720'}  {'RGB24_1024x576'}
  Columns 21 through 25
   {'RGB24_1184x656'}        {'RGB24_1280x720'}        {'RGB24_1280x960'}        {'RGB24_160x120'}
{'RGB24_176x144'}
  Columns 26 through 30
   {'RGB24_320x176'}    {'RGB24_320x240'}    {'RGB24_352x288'}    {'RGB24_432x240'}
{'RGB24_544x288'}
  Columns 31 through 35
   {'RGB24_640x360'}    {'RGB24_640x480'}    {'RGB24_752x416'}    {'RGB24_800x448'}
{'RGB24_800x600'}
  Columns 36 through 38
   {'RGB24_864x480'}  {'RGB24_960x544'}  {'RGB24_960x720'}
```

Step 3: Create a video input object

In this step you create the video input object that the toolbox uses to represent the connection between MATLAB and an image acquisition device. Using the properties of a video input object, you can control many aspects of the image acquisition process. For more information about image acquisition objects, see Creating Image Acquisition Objects.

To create a video input object, use the ***videoinput*** function at the MATLAB prompt. The ***DeviceInfo*** structure returned by the ***imaqhwinfo*** function contains the default videoinput function syntax for a device in the ***VideoInputConstructor*** field. For more information the device information structure, see previous section Determining the Supported Video Formats.

The following example creates a video input object for the DCAM adaptor. Substitute the adaptor name of the image acquisition device available on your system.

```
>> vid = videoinput('dcam',1,'Y8_1024x768')
```

The ***videoinput*** function accepts three arguments: the adaptor name, device ID, and video format. You retrieved this information in step 2. The adaptor name is the only required argument; the videoinput function can use defaults for the device ID and video format. To determine the default video format, look at the DefaultFormat field in the device information structure. See previous section Determining the Supported Video Formats for more information.

Instead of specifying the video format, you can optionally specify the name of a device configuration file, also known as a **camera file**. Device configuration files are typically supplied by frame grabber vendors. These files contain all the required configuration settings to use a particular camera with the device. See Using Device Configuration Files (Camera Files) for more information.

Viewing the Video Input Object Summary

To view a summary of the video input object you just created, enter the variable name ***vid*** at the MATLAB command prompt. The summary information displayed shows many of the characteristics of the object, such as the number of frames that will be captured with each trigger, the trigger type, and the current state of the object. You can use video input object properties to control many of these characteristics. See Step 5: Configure Object Properties (Optional) for more information.

```
>> vid

Summary of Video Input Object Using 'XCD-X700  1.05'.

   Acquisition Source(s):  input1 is available.

  Acquisition Parameters:  'input1' is the current selected source.
                           10 frames per trigger using the selected source.
                           'Y8_1024x768' video data to be logged upon START.
                           Grabbing first of every 1 frame(s).
                           Log data to 'memory' on trigger.

Trigger Parameters:  1 'immediate' trigger(s) on START.
         Status:  Waiting for START.
                  0 frames acquired since starting.
                  0 frames available for GETDATA.
```

Step 4: Preview the video stream (Optional)

After you create the video input object, MATLAB is able to access the image acquisition device and is ready to acquire data. However, before you begin, you might want to see a preview of the video stream to make sure that the image is satisfactory. For example, you might want to change the position of the camera, change the lighting, correct the focus, or make some other change to your image acquisition setup.

This step is **optional** at this point in the procedure because you can preview a video stream at any time after you create a video input object.

To preview the video stream in this example, enter the preview function at the MATLAB prompt, specifying the video input object created in step 3 as an argument.

```
>>preview(vid)
```

The ***preview*** function opens a Video Preview figure window on your screen containing the live video stream. To stop the stream of live video, you can call the ***stoppreview*** function. To restart the preview stream, call preview again on the same video input object.

While a preview window is open, the video input object sets the value of the Previewing property to 'on'. If you change characteristics of the image by setting image acquisition object properties, the image displayed in the preview window reflects the change. Fig. 2 shows the Video Preview window for the example.

To close the Video Preview window, click the Close button in the title bar or use the ***closepreview*** function, specifying the video input object as an argument.

```
>>closepreview(vid)
```

Calling ***closepreview*** without any arguments closes all open Video Preview windows.

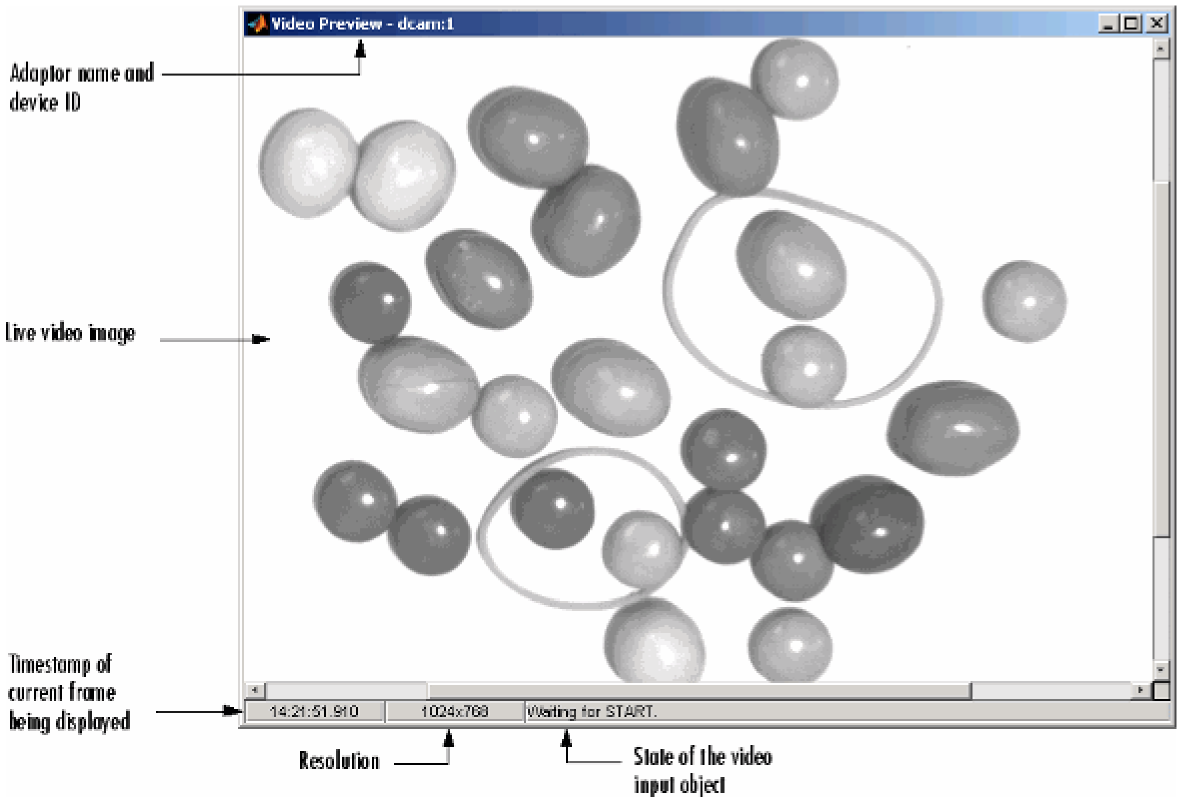

Video Preview - dcam:1
Adaptor name and device ID
Live video image
Timestamp of current frame being displayed
14:21:51.910
1024x768
Waiting for START.
Resolution
State of the video input object

Fig. 2. Video Preview Window

Step 5: Configure image acquisition object properties (Optional)

After creating the video input object and previewing the video stream, you might want to modify characteristics of the image or other aspects of the acquisition process. You accomplish this by setting the values of image acquisition object properties. This section:

- Describes the types of **image acquisition objects** used by the toolbox
- Describes how to **view all the properties** supported by these objects, with their current values
- Describes how to **set** the values of object properties

Types of Image Acquisition Objects

The toolbox uses two types of objects to represent the connection with an image acquisition device:

- Video input objects
- Video source objects

A **video input object** represents the connection between MATLAB and a video acquisition device at a high level. The properties supported by the video input object are the same for every type of device. You created a video input object using the ***videoinput*** function in step 3.

When you create a video input object, the toolbox automatically creates one or more video source objects associated with the video input object. Each video source object represents a collection of one or more physical data sources that are treated as a single entity. The number of video source objects the toolbox creates depends on the device and the video format you specify. At any one time, only one of the video source objects, called the selected source, can be active. This is the source used for acquisition. For more information about these image acquisition objects, see Creating Image Acquisition Objects.

Viewing Object Properties

To view a complete list of all the properties supported by a video input object or a video source object, use the get function. To list the properties of the video input object created in step 3, enter this code at the MATLAB prompt.

```
>> get(vid)
```

The get function lists all the properties of the object with their current values.

```
General Settings:
  DeviceID = 1
  DiskLogger = []
  DiskLoggerFrameCount = 0
  EventLog = [1x0 struct]
  FrameGrabInterval = 1
  FramesAcquired = 0
  FramesAvailable = 0
  FramesPerTrigger = 10
  Logging = off
  LoggingMode = memory
  Name = Y8_1024x768-dcam-1
  NumberOfBands = 1
  Previewing = on
  ReturnedColorSpace = grayscale
  ROIPosition = [0 0 1024 768]
  Running = off
  Tag =
  Timeout = 10
  Type = videoinput
  UserData = []
  VideoFormat = Y8_1024x768
  VideoResolution = [1024 768]
```

You can find a description of the properties available in the toolbox in Image Acquisition Toolbox Properties.

To view the properties of the currently selected video source object associated with this video input object, use the ***getselectedsource*** function in conjunction with the get function. The ***getselectedsource*** function returns the currently active video source. To list the properties of the currently selected video source object associated with the video input object created in step 3, enter this code at the MATLAB prompt.

```
>> get(getselectedsource(vid))
```

The ***get*** function lists all the properties of the object with their current values.

Note: Video source object properties are device specific. The list of properties supported by the device connected to your system might differ from the list shown in this example.

```
General Settings:
  Parent = [1x1 videoinput]
  Selected = on
  SourceName = input1
  Tag =
  Type = videosource

 Device Specific Properties:
  FrameRate = 15
  Gain = 2048
  Shutter = 2715
```

Setting Object Properties

To set the value of a video input object property or a video source object property, you reference the object property as you would a field in a structure, using dot notation.

Some properties are read only; you cannot set their values. These properties typically provide information about the state of the object. Other properties become read only when the object is running. To view a list of all the properties you can set, use the set function, specifying the object as the only argument.

To implement continuous image acquisition, the example sets the ***TriggerRepeat*** property to ***Inf***. To set this property, enter this code at the MATLAB prompt.

```
>> vid.TriggerRepeat = Inf;
```

To help the application keep up with the incoming video stream while processing data, the example sets the FrameGrabInterval property to 5. This specifies that the object acquire every fifth frame in the video stream. (You might need to experiment with the value of the FrameGrabInterval property to find a value that provides the best response with your image acquisition setup.) This example shows how you can set the value of an object property by referencing the property as you would reference a field in a MATLAB structure.

```
>> vid.FrameGrabInterval = 5;
```

To set the value of a video source object property, you must first use the getselectedsource function to retrieve the object. (You can also get the selected source by searching the video input object Source property for the video source object that has the Selected property set to 'on'.)

To illustrate, the example assigns a value to the Tag property.

```
>> vid_src = getselectedsource(vid);
>> vid_src.Tag = 'motion detection setup';
```

Step 6: Acquire Image Data

After you create the video input object and configure its properties, you can acquire data. This is typically the core of any image acquisition application, and it involves these steps:

- **Starting the video input object** — You start an object by calling the ***start*** function. Starting an object prepares the object for data acquisition. For example, starting an object locks the values of certain object properties (they become read only). Starting an object does **not** initiate the acquiring of image frames, however. The initiation of data logging depends on the execution of a trigger.

  The following example calls the start function to start the video input object. Objects stop when they have acquired the requested number of frames. Because the example specifies a continuous acquisition, you must call the stop function to stop the object.

- **Triggering the acquisition** — To acquire data, a video input object must execute a trigger. Triggers can occur in several ways, depending on how the ***TriggerType*** property is configured. For example, if you specify an immediate trigger, the object executes a trigger automatically, immediately after it starts. If you specify a manual trigger, the object waits for a call to the trigger function before it initiates data acquisition. For more information, see Acquiring Image Data.

  In the example, because the ***TriggerType*** property is set to 'immediate' (the default) and the ***TriggerRepeat*** property is set to Inf, the object automatically begins executing triggers and acquiring frames of data, continuously.

- **Bringing data into the MATLAB workspace** — The toolbox stores acquired data in a memory buffer, a disk file, or both, depending on the value of the video input object ***LoggingMode*** property. To work with this data, you must bring it into the MATLAB workspace. To bring multiple frames into the workspace, use the ***getdata*** function. Once the data is in the MATLAB workspace, you can manipulate it as you would any other data. For more information, see Working with Acquired Image Data.

Note: The toolbox provides a convenient way to acquire a single frame of image data that doesn't require starting or triggering the object. See Bringing a Single Frame into the Workspace for more information.

Running the Example

To run the example, enter the following code at the MATLAB prompt. The example loops until a specified number of frames have been acquired. In each loop iteration, the example calls getdata to bring the two most recent frames into the MATLAB workspace. To detect motion, the example subtracts one frame from the other, creating a difference image, and then displays it. Pixels that have changed values in the acquired frames will have nonzero values in the difference image.

The getdata function removes frames from the memory buffer when it brings them into the MATLAB workspace. It is important to move frames from the memory buffer into the MATLAB workspace in a timely manner. If you do not move the acquired frames from memory, you can quickly exhaust all the memory available on your system.

```
imaqhwinfo

% Create video input object.
vid = videoinput('dcam',1,'Y8_1024x768')    % adapt the parameters to the output of imaqhwinfo

% Set video input object properties for this application.
% Note that example uses both SET method and dot notation method.

set(vid,'TriggerRepeat',100);   % Si queremos que grabe continuamente: set(vid,'TriggerRepeat',Inf);
vid.FrameGrabInterval = 5;

% Set value of a video source object property.
vid_src = getselectedsource(vid);
set(vid_src,'Tag','motion detection setup');

% Create a figure window.
figure;

% Start acquiring frames.
start(vid)

% if needed, a “pause(2)” sentence may be inserted here

% Calculate difference image and display it.
while(vid.FramesAvailable >= 2)
   data = getdata(vid,2);
   diff_im = imabsdiff(data(:,:,:,1),data(:,:,:,2));
   imshow(diff_im);
   drawnow % update figure window
  % if needed, a “pause(0.01)” sentence may be inserted here

end
```

```
stop(vid)
```

Note that a ***drawnow*** is used after the call to ***imshow*** in order to ensure that the figure window is updated. This is good practice when updating a GUI or figure inside a loop

The following figure shows how the example displays detected motion. In the figure, areas representing movement are displayed.

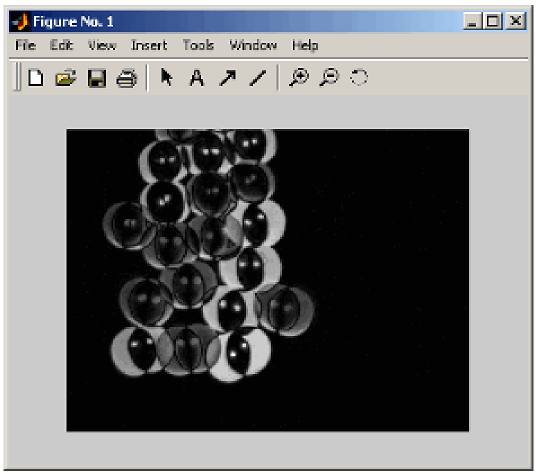


Fig. 3. Window Displayed by Example

Image Data in the MATLAB Workspace

In the example, the getdata function returns the image frames in the variable data as a 480-by-640-by-1-by-10 array of 8-bit data (uint8).

```
whos
  Name        Size          Bytes  Class
  data       4-D          3072000  uint8 array
  dev_info     1x1            1601  struct array
  info       1x1          2467  struct array
  vid        1x1          1138  videoinput object
  vid_src      1x1            726  videosource object
```

The height and width of the array are primarily determined by the video resolution of the video format. However, you can use the ROIPosition property to specify values that supersede the video resolution. Devices typically express video resolution as column-by-row; MATLAB expresses matrix dimensions as row-by-column.

The third dimension represents the number of color bands in the image. Because the example data is a grayscale image, the third dimension is 1. For RGB formats, image frames have three bands: red is the first, green is the second, and blue is the third. The fourth dimension represents the number of frames that have been acquired from the video stream.

Step 7: Clean Up

When you finish using your image acquisition objects, you can remove them from memory and clear the MATLAB workspace of the variables associated with these objects.

```
delete(vid)
clear
close(gcf)
```

For more information, see Deleting Image Acquisition Objects.

References

https://es.mathworks.com/help/imaq/getting-started-with-image-acquisition-toolbox.html

Image Acquisition Toolbox™ User's Guide. MathWorks. R2018a