CX - Application Note for

MATLAB Image Acquisition Toolbox

Rev 1.0

AT - Automation Technology GmbH

MathWorks®

GiGE VISION

GENICAM

Table of Content

Table of Content 2
MATLAB Image Acquisition Toolbox (imaqtool) 3
Start the Image Acquisition Toolbox 3
Connect a device 3
Get further Information 4
Start Preview 4
Start Acquisition 5
Region of Interest 5
Work with 4-D uint array 7
Connect to a device with functions 9
Video input and video source object 10
Take frames via functions 11
Source code 13
Connect camera and acquire ten frames 13
Show camera information 14
Plot first frame from local ImaqData 14
Document Revision 15

MATLAB Image Acquisition Toolbox (imaqtool)

Start the Image Acquisition Toolbox

Start the Image Acquisition Tool in MATLAB over the desktop 'APPS' tab and then select 'Image Acquisition'. Another way is to write `imaqtool` inside the command window to launch the Image Acquisition Tool. The Image Acquisition Tool looks like figure 1.

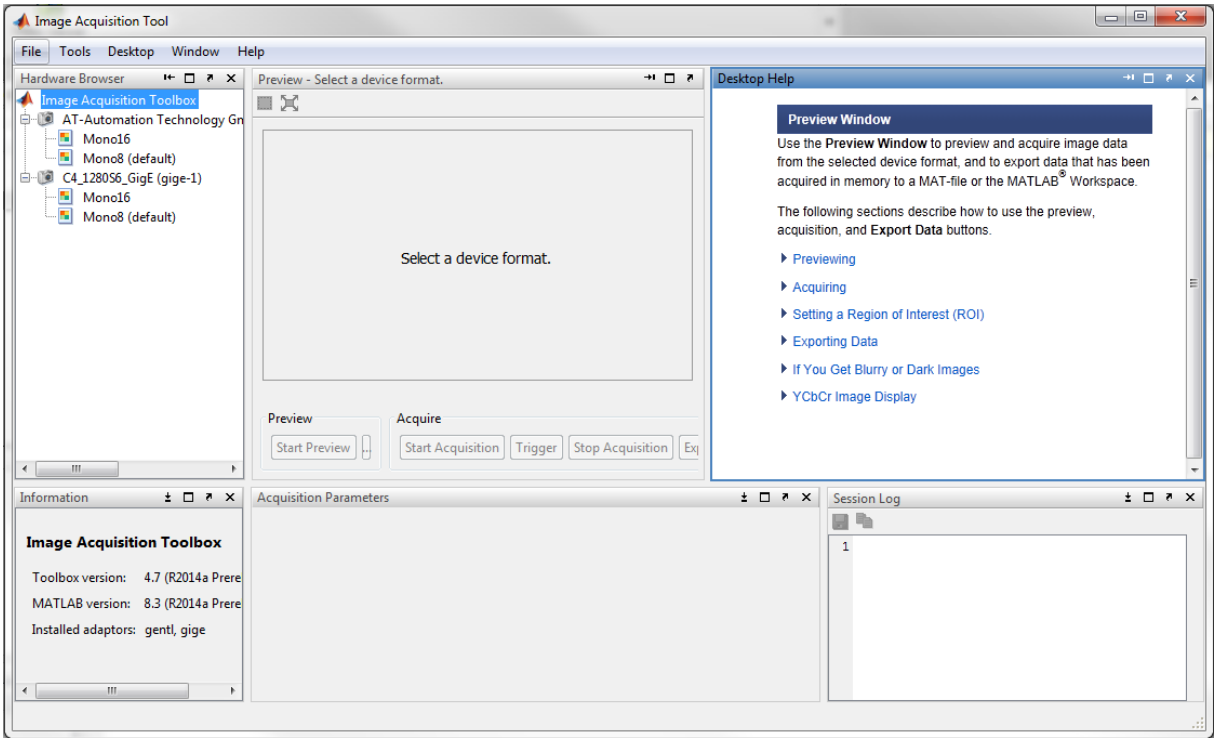

Figure 1: Image Acquisition Tool

Connect a device

To find a connected device select 'Tools' and click 'Refresh Image Acquisition Hardware (imaqreset)'. In the Hardware Browser is now a selection of different types to connect with the device.

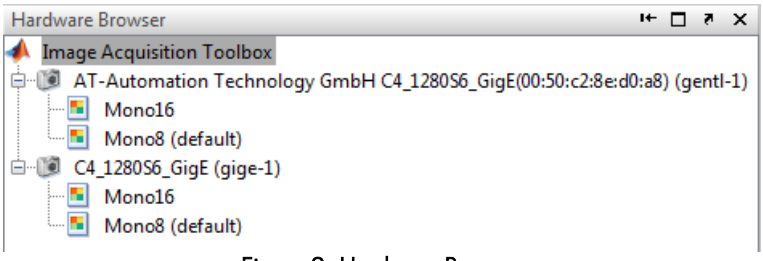

Figure 2: Hardware Browser

You can select between the 'gentl' and the 'gige' adaptor. For the next steps the GigE adaptor in Mono8 format is selected.

Get further Information

After selecting GigE adaptor and Mono8 format you can see further information about the device (Figure 3).

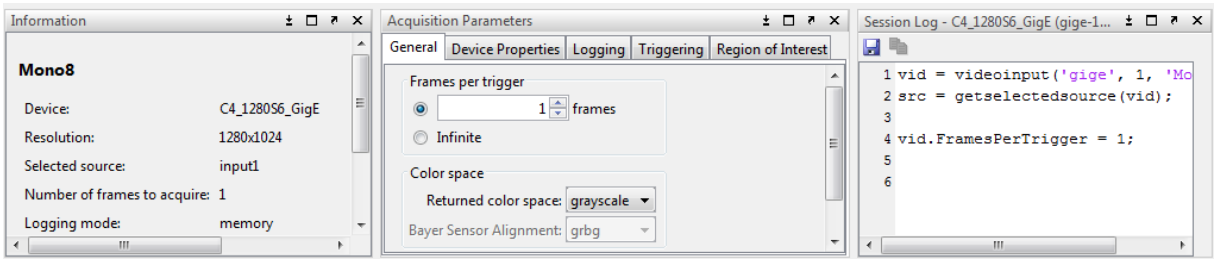

Figure 3: Information, Parameters and Session Log

The left window shows the information of the device like the device name, resolution and so on. In the middle you can change the acquisition parameters for example how many frames you want to grab. On the right is the 'Session Log', which shows the needed MATLAB functions for the selection you've done. In figure 4 you can see the functions for selecting 'gige' adaptor 'Mono8' format and take one frame.

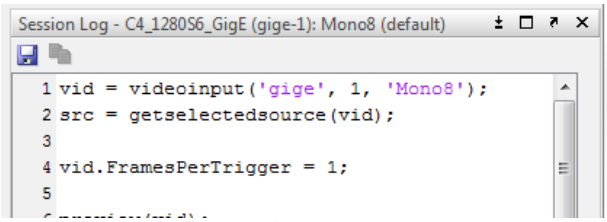

Figure 4: Session Log

Start Preview

To start a preview click the button 'Start Preview'. While running the preview you can sharpen the image with the lens of the camera.

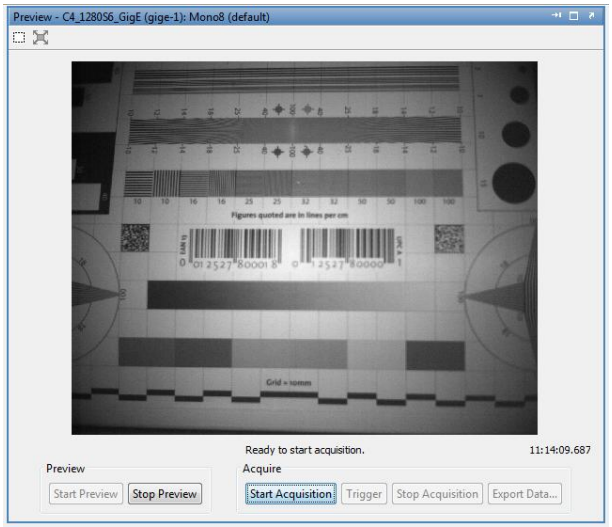

Figure 5: Preview for sharpen the image

After you have sharpen the image click 'Stop Preview' to end the preview.

Start Acquisition

Before you start an acquisition you can change some parameters. First just grab ten frames and save them to the workspace.

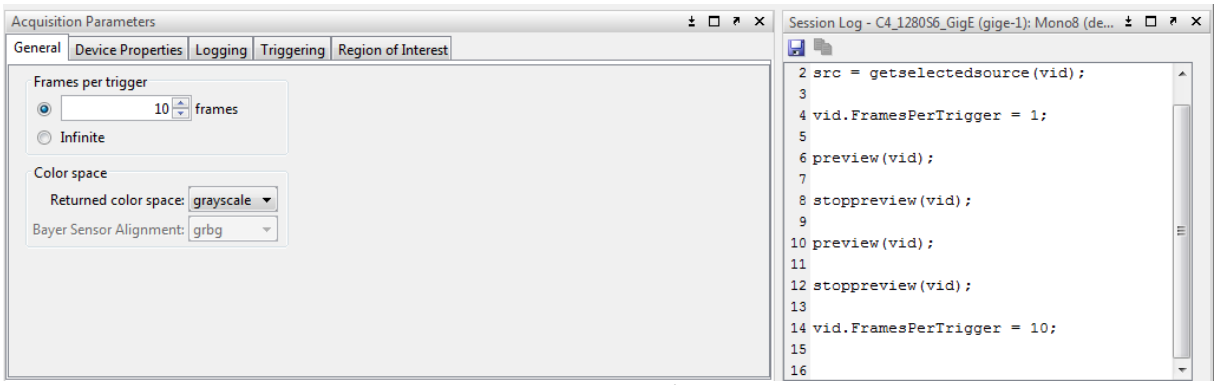

Figure 6: grab ten frames

Now click 'Start Acquisition' to grab the frames. The program automatically stops after grabbing the frames and your Preview window should look like these.

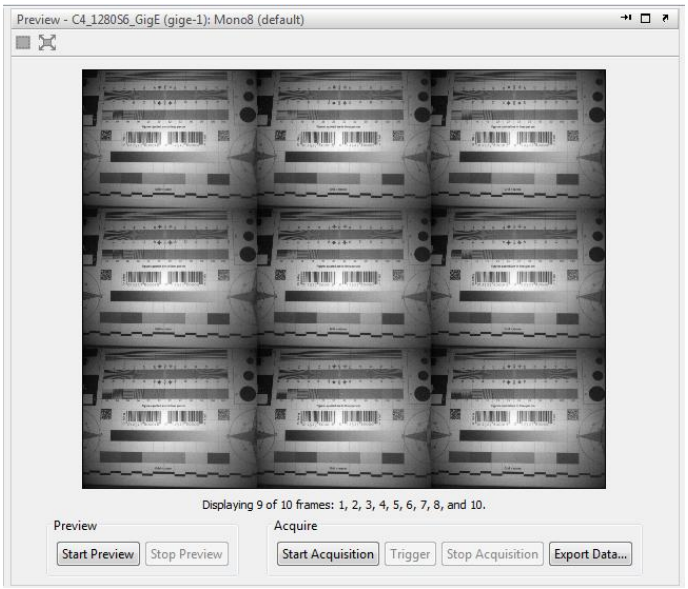

Figure 7: Grab ten frames

To save the data to workspace click 'Export Data…' select 'Data Destination: MATLAB Workspace' and type a name for the variable. In the workspace is now a four-dimensional array with the data.

To save the data to a '.mat' file click 'Export Data…' select 'Data Destination: MAT-File'. The data is saved to the current folder.

Region of Interest

For our cameras the acquisition parameters for the region of interest (ROI) should be untouched because they are read only. But it is possible to change the height and the y-offset of the area of interest (AOI) in the tab 'Device Properties'. Figure 8 shows the default settings for the resolution height for the C4-1280 camera.

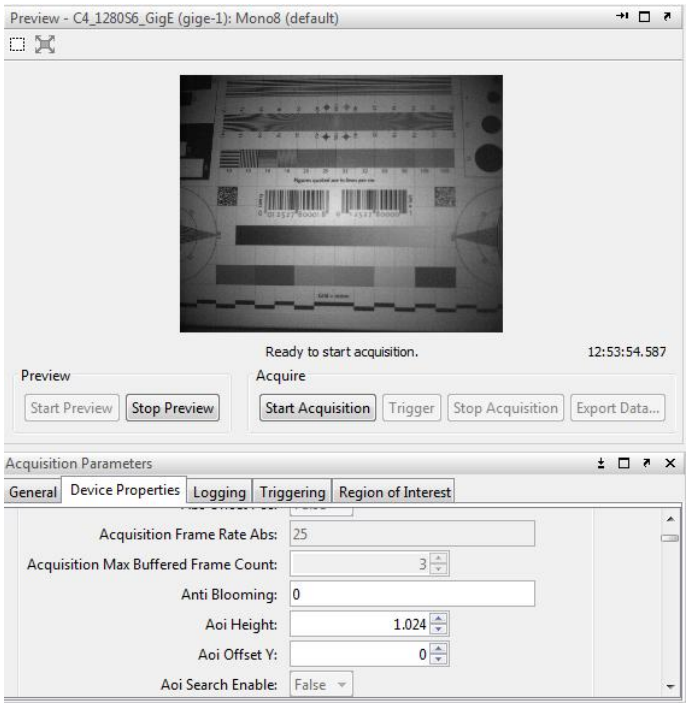

Figure 8: Full resolution in y-direction

To get the upper part of the frame change the 'Aoi Height' to 500 (figure 9).

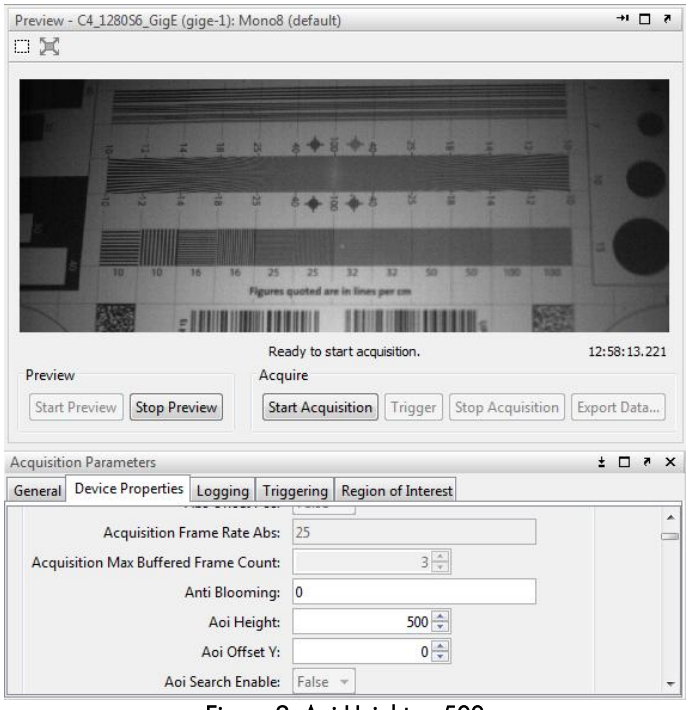

Figure 9: Aoi Height = 500

To get the lower part of the frame set the 'Aoi Offset Y' also to 500 (figure 10)

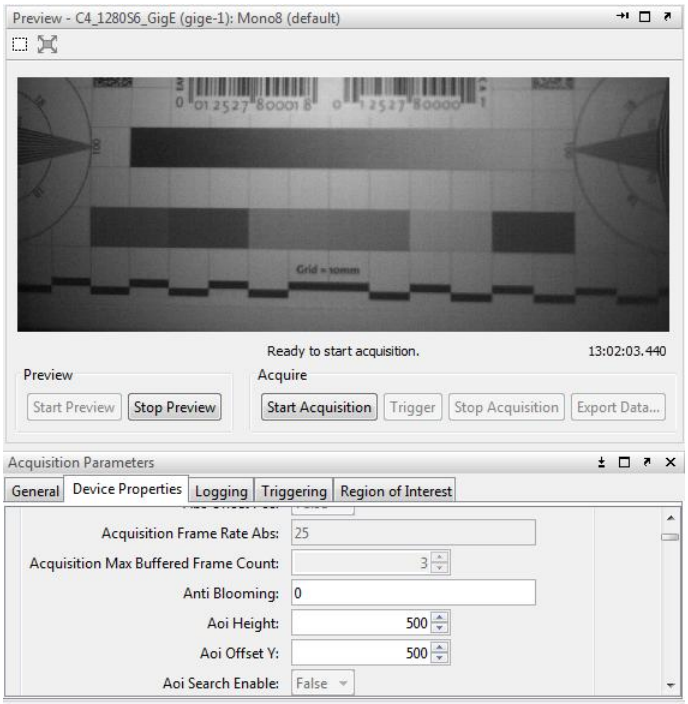

Figure 10: Aoi Height = Aoi Offset Y = 500

Work with 4-D uint array

After saving the data from the acquisition in a '.mat' file you can get the saved data in your workspace by double-click on the '.m' file. In this example the filename is 'TenFrames.mat' and the variable name is 'frames'.

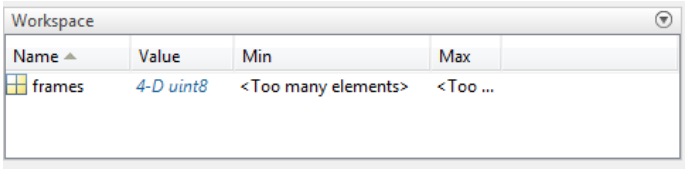

Figure 11: 4-D uint8 array

The elements of the 4-D uint8 array are

1. Height
2. Width
3. Colorspace (1: grayscale, 3: RGB)
4. Number of frames

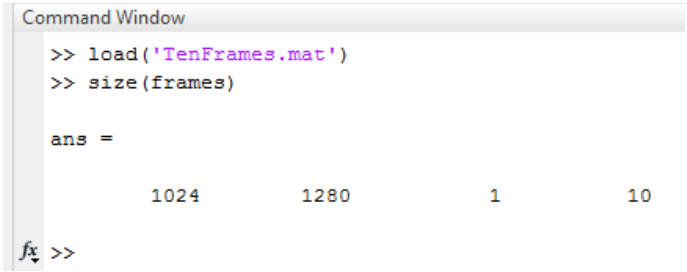

Figure 12: Size of the 4-D uint8 array

If you want to see the frames you've taken you can use the command 'implay(x)' (x stands for the array) and the 'Movie Player' will open. With the 'Movie Player' you can watch each single frame (in figure 13 the actual frame is number two of ten). You also can export the actual frame to the image tool. Go to 'File' and then click 'Export to Image Tool'. With the image tool the actual frame can be saved to an two-dimensional array to the workspace.

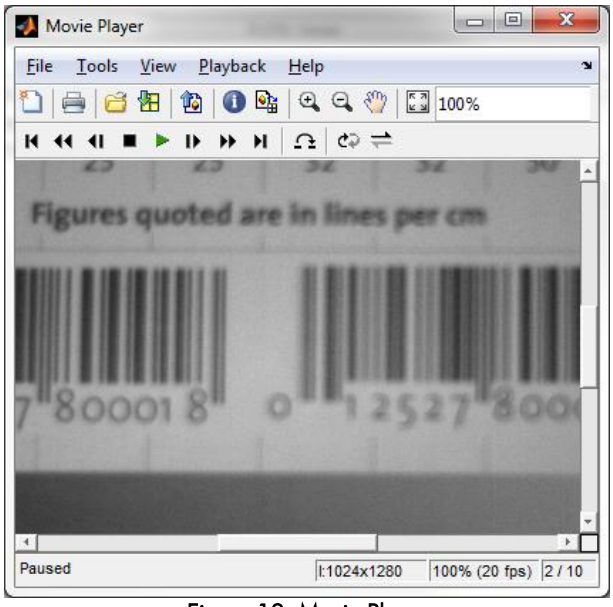

Figure 13: Movie Player

To make operations with the data it's more comfortable to create an '.m' file to get a three-dimensional array (height, width, frame) for the frames. Listing 1 shows an example how to create an 3-D array from the saved data and show the first frame.

ShowImages.m

```
1 -    clear all;
2 -    close all;
3 -    load('TenFrames.mat');
4 -    [h,w,cmp,numel]=size(frames);
5
6 -    for i = 1:numel
7 -        F(:,:,i)=frames(:,:,cmp,i);
8 -    end
9 -    imagesc(F(:,:,1))
10 -   colormap(gray)
```

Listing 1: Create a 3-D array

Connect to a device with functions

In the 'Session Log' of the Image Acquisition Tool you've seen the functions you need if you want to connect with a hardware device, change parameters and start acquisition.

clear	Clear image acquisition object from MATLAB workspace
delete	Remove image acquisition object from memory
disp	Display method for image acquisition objects
imaqfind	Find image acquisition objects
imaqhwinfo	Information about available image acquisition hardware
imaqmem	Limit memory or display memory usage for Image Acquisition Toolbox software
imaqreset	Disconnect and delete all image acquisition objects
imaqtool	Launch Image Acquisition Tool
start	Obtain exclusive use of image acquisition device
stop	Stop video input object
videoinput	Create video input object
imaq.VideoDevice	Acquire one frame at a time from video device

Table 1: Functions for device connection

To get information about connected hardware use the function 'imaqhwinfo'. Listing 2 shows some possibilities to get information about the hardware. The first function 'imaqhwinfo' shows the installed adaptors and actual version of the image acquisition toolbox. With 'imaqhwinfo('gige')' you get the Device ID from the connected hardware and with 'imaqhwinfo('gige',1)' the information about the camera like supported formats and the device name.

Command Window

```
>> imaqhwinfo

ans =

    InstalledAdaptors: {'gentl'  'gige'}
        MATLABVersion: '8.3 (R2014a Prerelease)'
          ToolboxName: 'Image Acquisition Toolbox'
       ToolboxVersion: '4.7 (R2014a Prerelease)'

>> imaqhwinfo('gige')

ans =

       AdaptorDllName: 'C:\MATLAB\SupportPackages\R2014aPr
    AdaptorDllVersion: '4.7 (R2014a Prerelease)'
          AdaptorName: 'gige'
            DeviceIDs: {[1]}
           DeviceInfo: [1x1 struct]

>> imaqhwinfo('gige',1)

ans =

              DefaultFormat: 'Mono8'
       DeviceFileSupported: 0
                DeviceName: 'C4_1280S6_GigE'
                  DeviceID: 1
     VideoInputConstructor: 'videoinput('gige', 1)'
    VideoDeviceConstructor: 'imaq.VideoDevice('gige', 1)'
          SupportedFormats: {'Mono16'  'Mono8'}
```

Listing 2: Information about the connected device

If no hardware is connected the output of 'imaqhwinfo('gige')' or another adaptor will look like listing 3. There you can see that the 'DeviceIDs' and the 'DeviceInfo' are empty.

```
>> imaqhwinfo('gige')

ans =

       AdaptorDllName: 'C:\MATLAB\SupportPackages
    AdaptorDllVersion: '4.7 (R2014a Prerelease)'
          AdaptorName: 'gige'
            DeviceIDs: {1x0 cell}
           DeviceInfo: [1x0 struct]
```

Listing 3: Output for 'imaqhwinfo('gige')' if no device connected

Video input and video source object

To change the parameters you first have to create an video input object and an video source object.

```
32      % create the video input and video source object
33 -    vid = videoinput(adaptorname,DevID, format);    % video input object
34 -    src = getselectedsource(vid);                   % video source object
```

Listing 4 Create video input and video source object

The 'vid' object and the 'src' object are now in the workspace. To get more information just double-click on them in the workspace. Figure 14 shows the video source object and figure 15 the video input object.

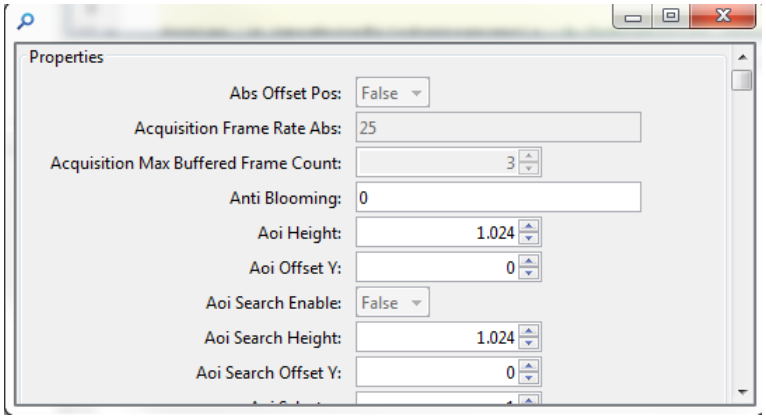

Figure 14: Video source object

With the video source object you can change or get parameters like the 'Aoi Height' (Listing 5).

```
>> src.AoiHeight

ans =

        1024

>> src.AoiHeight = 500;
Warning: Resetting the ROI as the max width and height has been updated.
>>
```

Listing 5: Get and set parameter

The video input object allows you to start/stop preview take frames change frame rate and so on.

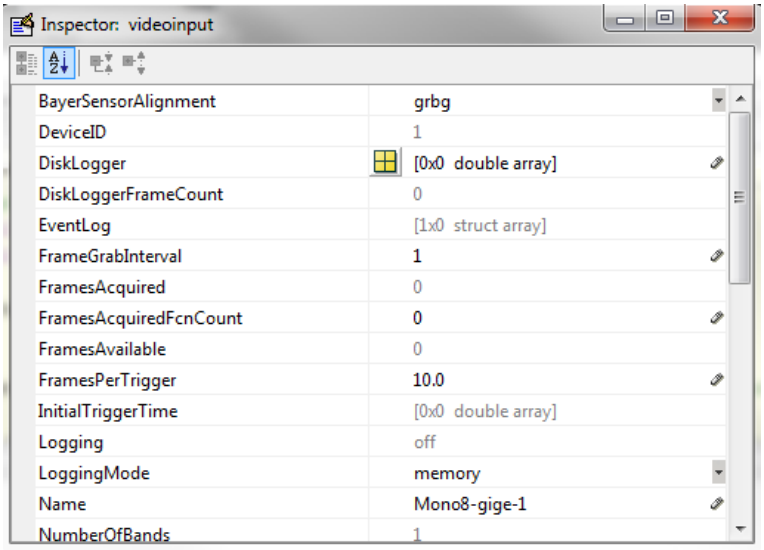

Figure 15: Video input object

Take frames via functions

After connecting the device and creating the objects you are ready to take frames save the data and so on. Listing 6 shows an example for taking 10 frames ('vid.FramesPerTrigger=10') with the default trigger settings ('vid.TriggerRepeat = 0').

```
% create the video input and video source object
vid = videoinput(adaptorname,DevID, format);    % video input object
src = getselectedsource(vid);                   % video source object

% information about the connected device
CamInfo(vid,src);

% set frames per trigger
vid.FramesPerTrigger = framespertrigger;

% set trigger repeats
vid.TriggerRepeat = trigrepeat;

% set the trigger type
triggerconfig(vid,trigtype);

% start the acquisition
start(vid);

%wait until acquisition ready
wait(vid);

% if the expected number of frames not equals the acquired frames
if (vid.TriggerRepeat + 1) * vid.FramesPerTrigger ~= vid.FramesAcquired
   error('The expected number of frames not equals to the acquired frames')
   quit;
end
% save data to an '.mat' file
FrameData = getdata(vid,vid.FramesAcquired);
save(SaveFile, 'FrameData');
clear FrameData;
```

Listing 6: Take ten frames per trigger

Listing 6 runs with the default settings shown in listing 7

```
4       % create default settings
5 -     adaptorname         = 'gige';
6 -     format              = 'Mono8';
7 -     trigtype            = 'immediate';
8 -     SaveFile            = 'ImaqData.mat';
9 -     framespertrigger    = 10;
10
11       % default trigger repeat = 0 so only 10 frames will be taken
12 -    trigrepeat          = 0 ;
```

Listing 7: Default settings

Figure 16 shows the output of the function 'CamInfo(vidObj, srcObj)' (Listing 6, line 43).

```
Device Information:

      Manufacturer: AT-Automation Technology GmbH
       Device Name: C4_1280S6_GigE
         Device ID: 20502269
  Firmware Version: 1.4.1
     Actual Format: Mono8
            Height: 1280
             Width: 1024
```

Figure 16: Output of CamInfo(vidObj, srcObj)

Source code

Connect camera and acquire ten frames

```
clear all
close all
% create default settings
adaptorname         = 'gige';
format              = 'Mono8';
trigtype            = 'immediate';
SaveFile            = 'ImaqData.mat';
framespertrigger    = 10;

 % default trigger repeat = 0 so only 10 frames will be taken
trigrepeat          = 0 ;

% Disconnect and delete all image acquisition objects
imaqreset;

% Information about available image acquisition hardware
device  = imaqhwinfo(adaptorname);

% if no hardware found
if ~cell2mat(device.DeviceIDs)
   error('No image acquisition hardware connected (DeviceIDs is a 1x0 cell array) ');
   quit;
else
    % One device is connected
    DevID   = device.DeviceIDs(1);

    % convert cell array to single array
    DevID   = cell2mat(DevID);
end

% create the video input and video source object
vid = videoinput(adaptorname,DevID, format);    % video input object
src = getselectedsource(vid);                   % video source object

% information about the connected device
CamInfo(vid,src);

% set frames per trigger
vid.FramesPerTrigger = framespertrigger;

% set trigger repeats
vid.TriggerRepeat = trigrepeat;

% set the trigger type
triggerconfig(vid,trigtype);

% start the acquisition
start(vid);

%wait until acquisition ready
wait(vid);
```

```
% if the expected number of frames not equals the acquired frames
if (vid.TriggerRepeat + 1) * vid.FramesPerTrigger ~= vid.FramesAcquired
   error('The expected number of frames not equals to the acquired frames')
   quit;
end
% save data to an '.mat' file
FrameData = getdata(vid,vid.FramesAcquired);
save(SaveFile, 'FrameData');
clear FrameData;
```

Show camera information

```
function [] = CamInfo( vidObj, srcObj )
%   This function prints device information
%   to the command line.
StringLength = 80;

res         = vidObj.Videoresolution;
height      = res(1);
width       = res(2);

formTex = sprintf('\\n%%%ds: %%s', floor(StringLength / 2));
fprintf(formTex,'Device Information');

fprintf('\n');
fprintf(formTex,'Manufacturer',srcObj.DeviceManufacturerInfo);
fprintf(formTex,'Device Name', srcObj.DeviceModelName); %#ok<*CTPCT>
fprintf(formTex,'Device ID', srcObj.DeviceID);
fprintf(formTex,'Firmware Version' , srcObj.DeviceFirmwareVersion);
fprintf(formTex,'Actual Format',vidObj.VideoFormat);
fprintf(formTex,'Height', num2str(height));
fprintf(formTex,'Width', num2str(width));
fprintf('\n\n');

end
```

Plot first frame from local ImaqData

```
clear all;
close all;
load('ImaqData');
[h,w,cmp,numel]=size(FrameData);

for i = 1:numel
    F(:,:,i)=FrameData(:,:,cmp,i);
end
imagesc(F(:,:,1))
colormap(gray)
```

Document Revision

Rev. Nr.	Date	Modification
1.0	10.02.2014	First draft