Integrating a Thorlabs WFS Wavefront Sensor in MATLAB

Introduction:

Thorlabs WFS wavefront sensors can be controlled in MATLAB using the provided driver DLLs. These can be loaded into MATLAB using the “loadlibrary” command. This example uses the 64-bit version of the CCS drivers. If you want to use the 32-bit versions, please make sure to change the file paths and names accordingly.

The driver documentation files are saved to this folder during installation: C:\Program Files (x86)\IVI Foundation\VISA\WinNT\WFS\Manual

Programming:

1) **Preparing “visatype.h”:**

At first, comment out the “__fastcall” and “signed” calling conventions in the header file “visatype.h” in this folder: C:\Program Files\IVI Foundation\VISA\Win64\Include

---

```
/*----------------------------------------------------------------------------*/
/* Distributed by IVI Foundation Inc.                                         */
/*                                                                            */
/* Do not modify the contents of this file.                                   */
/*----------------------------------------------------------------------------*/
/*                                                                            */
/* Title   : VISATYPE.H                                                       */
/* Date    : 05-12-2009                                                       */
```

```
/* Purpose : Fundamental VISA data types and macro definitions              */
/*                                                                          */
/*--------------------------------------------------------------------------*/

#ifndef __VISATYPE_HEADER__
#define __VISATYPE_HEADER__

#if defined(_WIN64)
#define _VI_FAR
#define _VI_FUNC            //__fastcall
#define _VI_FUNCC           //__fastcall
#define _VI_FUNCH           //__fastcall
#define _VI_SIGNED          //signed
#elif (defined(WIN32) || defined(_WIN32) || defined(__WIN32__) || defined(__NT__)) &&
!defined(_NI_mswin16_)
#define _VI_FAR
.
.
.
```

If these conventions aren't commented out, they normally cause error messages when you try to run the MATLAB code.

2) Sample code:

This sample code can be copied into MATLAB and can be executed once the change in 1) is made.

```
clc;
clear;
disp('Start WFS.');
disp(' ');

%   Loading the dll and header file into MATLAB
libname='C:\Program Files\IVI Foundation\VISA\Win64\Bin\WFS_64.dll';
hfile='C:\Program Files\IVI Foundation\VISA\Win64\Include\WFS.h';
if (~libisloaded('WFS_64'))
    loadlibrary(libname,hfile,'includepath','C:\Program Files\IVI
Foundation\VISA\Win64\Lib_x64\msc','includepath','C:\Program Files\IVI
Foundation\VISA\Win64\Include','addheader','C:\Program Files\IVI
Foundation\VISA\Win64\Include\visa.h','addheader','C:\Program Files\IVI
Foundation\VISA\Win64\Include\vpptype.h');
end;

%   Displays the functions in the library
%   Also gives the data types used in a command
%   - Not necessary for normal use -
libfunctionsview 'WFS_64';

%   Some dll functions use pointers
%   The 'libpointer' command has to be used in MATLAB for this

%   Get connected WFS sensors
length=libpointer('longPtr',0);
calllib('WFS_64', 'WFS_GetInstrumentListLen',0,length);
disp(['There are ', num2str(length.value), ' WFS sensors connected']);
disp(' ');
```

```
DevID=libpointer('longPtr',0);
InUse=libpointer('longPtr',0);
InstrName=libpointer('int8Ptr',int8(zeros(1,25)));
InstrSN=libpointer('int8Ptr',int8(zeros(1,25)));
ResourceName=libpointer('int8Ptr',int8(zeros(1,25)));

for i=0:(length.value-1)
    calllib('WFS_64',
'WFS_GetInstrumentListInfo',0,i,DevID,InUse,InstrName,InstrSN,ResourceName)
;
    disp(['Device ID: ', num2str(DevID.value)]);
    disp(char(InstrName.value));
    disp(['SN: ', char(InstrSN.value)]);
    disp(' ');
end;

%   Select one of the connected WFS sensors
UsedDeviceNum = input('Device ID of the WFS you want to use: ');

%   Initialize the WFS
UsedDeviceStr = ['USB::0x1313::0x0000::',num2str(UsedDeviceNum)];
res=libpointer('int8Ptr',int8(UsedDeviceStr));
hdl=libpointer('ulongPtr',0);
calllib('WFS_64', 'WFS_init',res,1,1,hdl);

%   Select microlens array 0 and configure camera
calllib('WFS_64','WFS_SelectMla',hdl.value,0);
spotsx=libpointer('int32Ptr',0);
spotsy=libpointer('int32Ptr',0);
calllib('WFS_64','WFS_ConfigureCam',hdl.value, 0, 0, spotsx, spotsy);
calllib('WFS_64','WFS_SetReferencePlane',hdl.value,0);
calllib('WFS_64','WFS_SetPupil',hdl.value, 0.0, 0.0, 5.0, 5.0);

for j=1:100
    %   Take spotfield image
    exposureTimeAct=libpointer('doublePtr',0.0);
    masterGainAct=libpointer('doublePtr',0.0);
    calllib('WFS_64','WFS_TakeSpotfieldImageAutoExpos',hdl.value,
exposureTimeAct, masterGainAct);

    imageBuf=libpointer('uint8Ptr',zeros(1,(1280*1024)));
    rows=libpointer('int32Ptr',0);
    cols=libpointer('int32Ptr',0);
    calllib('WFS_64','WFS_GetSpotfieldImageCopy',hdl.value, imageBuf, rows,
cols);

    %   Change buffer array and show image of spotfield
    pic=reshape(imageBuf.value,[1280,1024]);
    image(pic);
    pause(0.25);
end;

%   Closing the WFS driver session and unloading the dll
a=calllib('WFS_64','WFS_close', hdl.value);
unloadlibrary('WFS_64');
```

The command “`libfunctionsview 'WFS_64'`” will open a window like this:

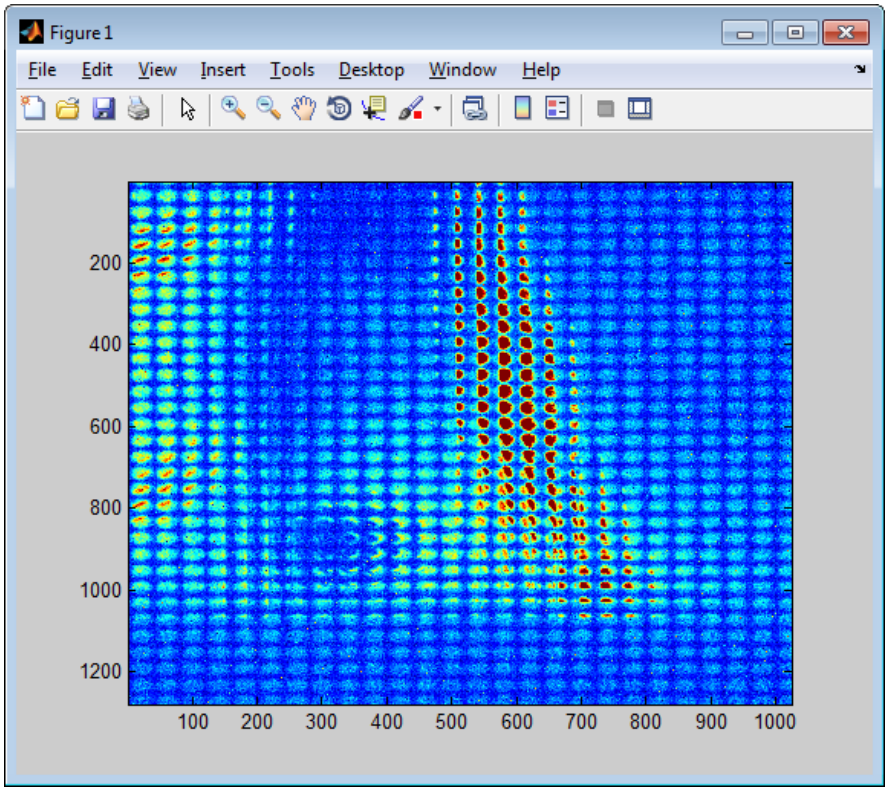
Functions in library WFS_64

Return Type	Name	Arguments
[long, longPtr]	WFS_AverageImage	(ulong, long, longPtr)
long	WFS_AverageImageRolling	(ulong, long, long)
[long, doublePtr, doublePtr, doublePtr, doublePtr]	WFS_CalcBeamCentroidDia	(ulong, doublePtr, doublePtr, doublePtr, doublePtr)
[long, doublePtr, doublePtr, doublePtr, doublePtr, doublePtr, dou...	WFS_CalcFourierOptometric	(ulong, long, long, doublePtr, doublePtr, doublePtr, doublePtr, doublePtr, doublePtr)
[long, longPtr, longPtr, doublePtr]	WFS_CalcImageMinMax	(ulong, longPtr, longPtr, doublePtr)
[long, doublePtr, doublePtr]	WFS_CalcMeanRmsNoise	(ulong, doublePtr, doublePtr)
[long, longPtr, doublePtr, doublePtr]	WFS_CalcReconstrDeviations	(ulong, long, longPtr, long, doublePtr, doublePtr)
long	WFS_CalcSpotToReferenceDeviati...	(ulong, long)
long	WFS_CalcSpotsCentrDiaIntens	(ulong, long, long)
[long, singlePtr]	WFS_CalcWavefront	(ulong, long, long, singlePtr)
[long, doublePtr, doublePtr, doublePtr, doublePtr, doublePtr, dou...	WFS_CalcWavefrontStatistics	(ulong, doublePtr, doublePtr, doublePtr, doublePtr, doublePtr, doublePtr)
long	WFS_CheckHighspeedCentroids	(ulong)
[long, longPtr, longPtr]	WFS_ConfigureCam	(ulong, long, long, longPtr, longPtr)
[long, singlePtr, singlePtr]	WFS_ConvertWavefrontWaves	(ulong, double, singlePtr, singlePtr)
long	WFS_CreateDefaultUserReference	(ulong)
long	WFS_CutImageNoiseFloor	(ulong, long)
long	WFS_DoSphericalRef	(ulong)
[long, singlePtr, singlePtr]	WFS_Flip2DArray	(ulong, singlePtr, singlePtr)
[long, doublePtr, doublePtr, doublePtr, doublePtr]	WFS_GetAoi	(ulong, doublePtr, doublePtr, doublePtr, doublePtr)
[long, longPtr]	WFS_GetBlackLevelOffset	(ulong, longPtr)
[long, doublePtr]	WFS_GetExposureTime	(ulong, doublePtr)
[long, doublePtr, doublePtr, doublePtr]	WFS_GetExposureTimeRange	(ulong, doublePtr, doublePtr, doublePtr)
[long, longPtr, longPtr, longPtr, longPtr, longPtr, longPtr]	WFS_GetHighspeedWindows	(ulong, longPtr, longPtr, longPtr, longPtr, longPtr, longPtr)
[long, int8Ptr, int8Ptr, int8Ptr, int8Ptr]	WFS_GetInstrumentInfo	(ulong, int8Ptr, int8Ptr, int8Ptr, int8Ptr)
[long, longPtr, longPtr, int8Ptr, int8Ptr, int8Ptr]	WFS_GetInstrumentListInfo	(ulong, long, longPtr, longPtr, int8Ptr, int8Ptr, int8Ptr)
[long, longPtr]	WFS_GetInstrumentListLen	(ulong, longPtr)

It shows the commands in the driver dll file which can be used with the WFS sensor. It is useful for the programming in MATLAB because it also shows which data types are expected as input / output variables by MATLAB.

Once the programming is finished, this window will be unnecessary in most cases and can be commented out.

The measurement part of the program takes 100 spotfield images and displays them. You can see an example of the spotfield display window below:



For further assistance please contact us on: europe@thorlabs.com / +49 (0) 8131-5956-2