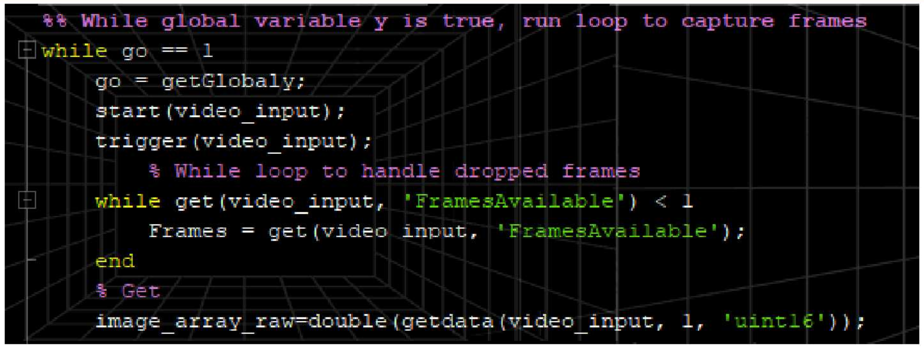
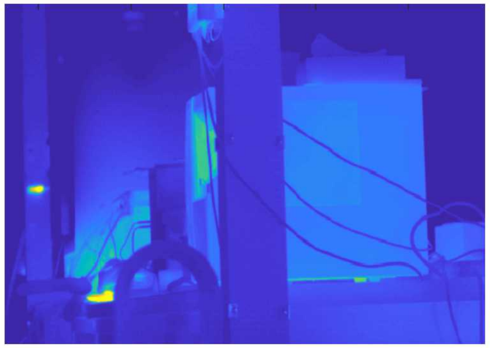
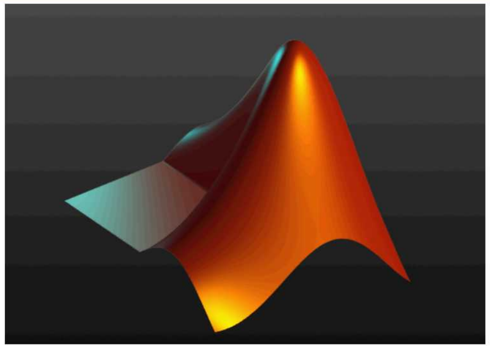
SAND2019-3702PE



```
%% While global variable y is true, run loop to capture frames
while go == 1
    go = getGlobaly;
    start(video_input);
    trigger(video_input);
        % While loop to handle dropped frames
    while get(video_input, 'FramesAvailable') < 1
        Frames = get(video_input, 'FramesAvailable');
    end
    % Get
    image_array_raw=double(getdata(video_input, 1, 'uint16'));
```





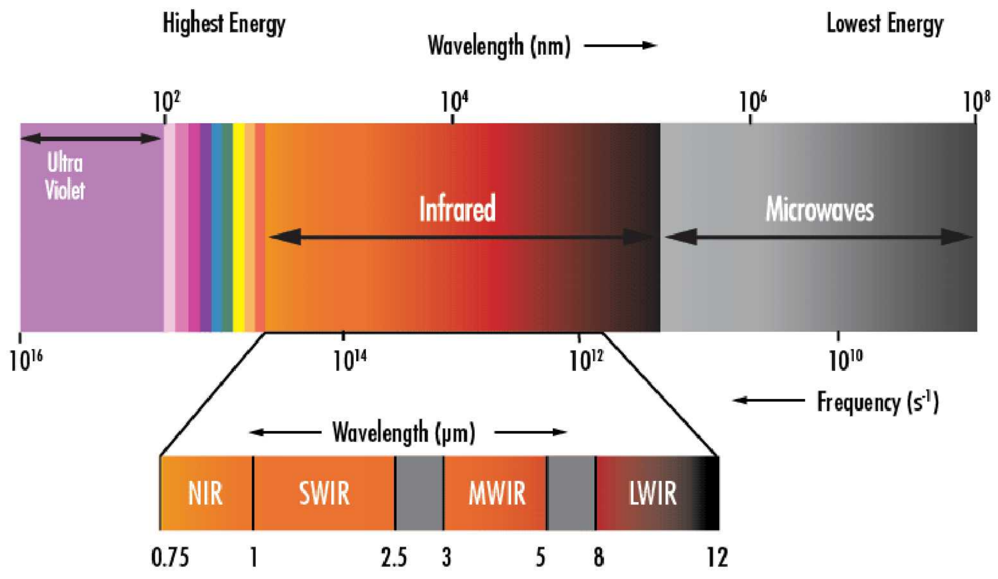
MATLAB® Infrared Camera Integration

Joshua Stanford

Primary Standards Lab, Sandia National Laboratories

Background

- Infrared Radiation (IR) is a form of electromagnetic radiation at wavelengths larger than that of visible light.
- Mid-Wavelength IR (MWIR) has a wavelength range of 3µm to 5µm.



- IR cameras are sensors capable of detecting IR and creating images based on the detections. These IR cameras are valuable to many different industries as well as a wide variety of research and development.

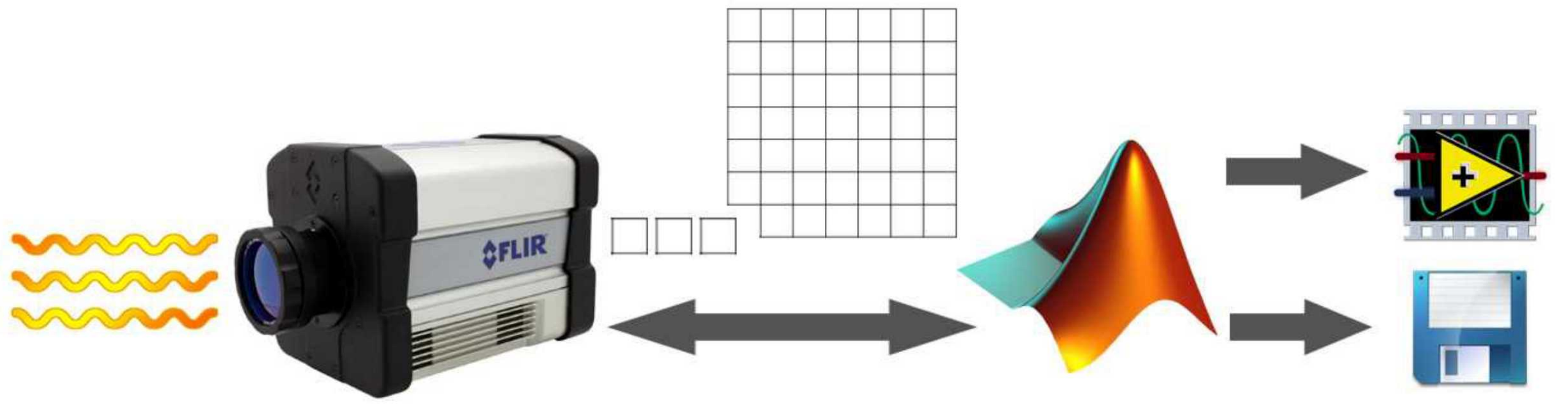
FLIR® Research IR Camera

- Fastest commercial infrared camera in the world.
- Operates in the MWIR (3μm- 5μm) spectrum.
- 200 Megapixels per second for 565 FPS @ 640x512 pixels.
- Gigabit Ethernet (GigE) digital video output.

- Camera has proprietary FLIR® software but no integration with MATLAB® or LabVIEW® Integrated Development Environments (IDEs). Proprietary software does not save in desired output of comma delimited spreadsheet (*.CSV).

Project Goals

- Communicate with a FLIR® MWIR camera via a MATLAB® script.
- Create a MATLAB® Graphic User Interface (GUI) for camera control.
- Save IR image and pixel intensity array to local drive.
- Integrate MATLAB® script with LabVIEW® virtual instrument.



*Certain commercial equipment, instruments, software, or materials are identified in this paper in order to adequately describe the experimental procedure. Such identification does not imply recommendation or endorsement by the authors, Sandia National Laboratories, nor does it imply that the materials or equipment identified are the only or best available for the purpose.

IDE Requirements

MATLAB® IDE Version

R2018a

MATLAB® Addons

Image Acquisition Toolbox Support Package for GigE Vision Hardware version 18.1.0

Image Acquisition Toolbox Support Package for GenICam Interface version 18.1.0

Image Acquisition Toolbox version 5.4

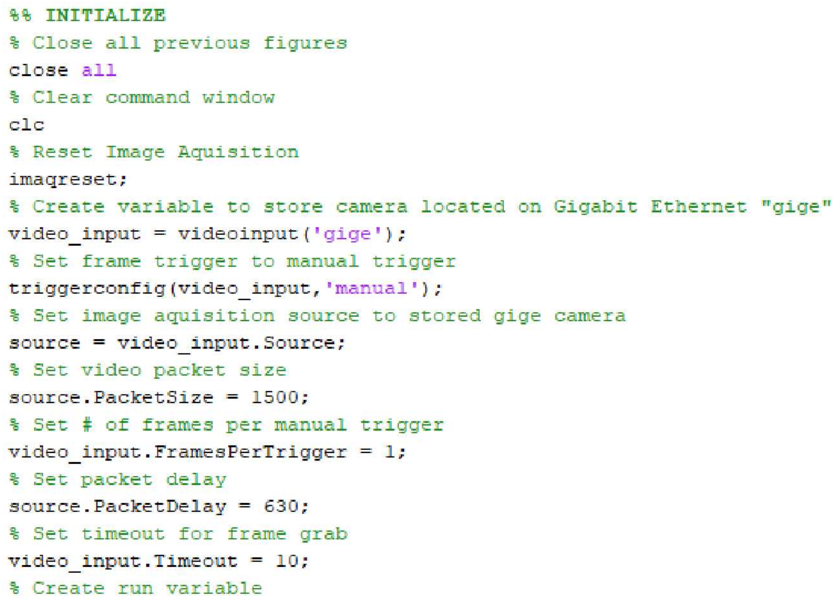
LabVIEW® IDE Version

NATIONAL INSTRUMENTS LabVIEW™ 2015

MATLAB® Camera Communication

The MATLAB script establishes communication with the camera via functions that are included in the Gige, GenIcam, and Image acquisition addons.

The MATLAB script must also specify how the camera will operate. This includes determining if the camera is manually triggered or continuous, the packet size, packet delay, and video timeout.

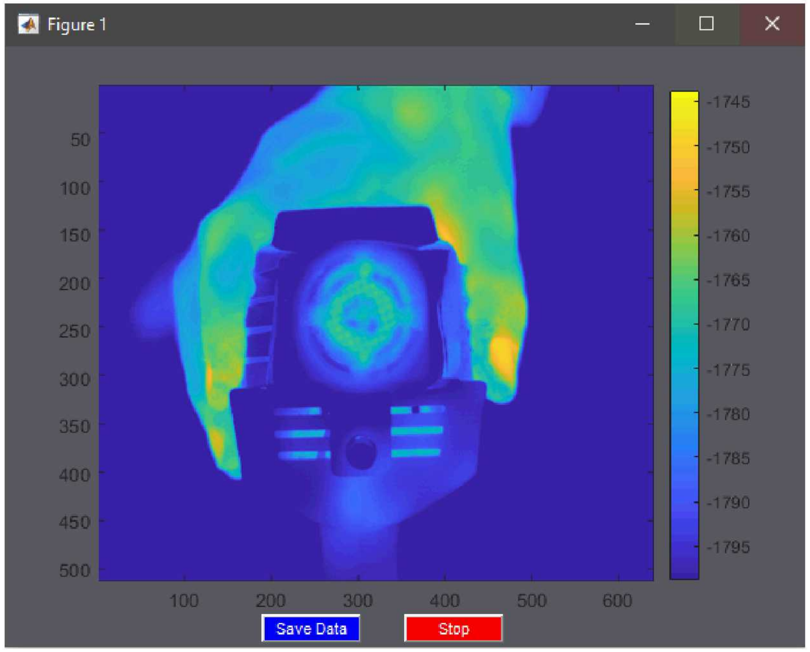


```matlab
%% INITIALIZE
% Close all previous figures
close all
% Clear command window
clc
% Reset Image Aquisition
imaqreset;
% Create variable to store camera located on Gigabit Ethernet "gige"
video_input = videoinput('gige');
% Set frame trigger to manual trigger
triggerconfig(video_input,'manual');
% Set image aquisition source to stored gige camera
source = video_input.Source;
% Set video packet size
source.PacketSize = 1500;
% Set # of frames per manual trigger
video_input.FramesPerTrigger = 1;
% Set packet delay
source.PacketDelay = 630;
% Set timeout for frame grab
video_input.Timeout = 10;
% Create run variable
```

MATLAB® GUI

Figure Modifications

The menu bar was removed to create a simpler look. The background color was changed to a darker color to lessen the harsh contrast from the IR image.



Save Data Button

A save data button was added to allow the user to save the desired outputs. The text, color, and position were customized. When clicked, the save button calls a separate function to perform the save.

Stop Button

A stop button was added to allow the user to save the desired outputs. The text, color, and position were customized. When clicked, the stop button calls a separate function to end the script.

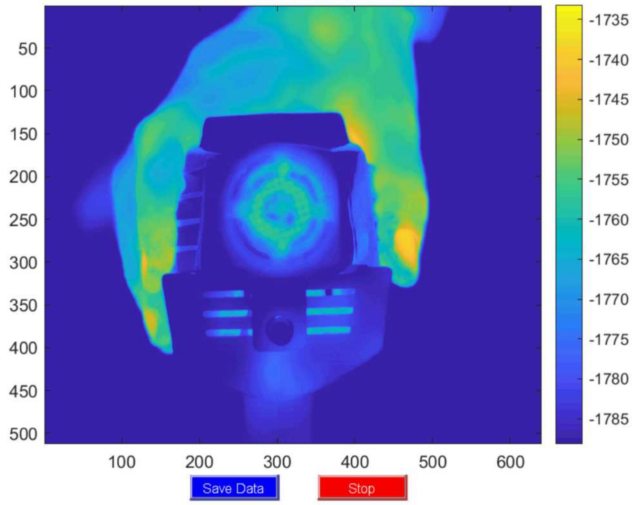
Color Bar

A color bar was added to the figure to show how each color represents the corresponding value in the image array. (Note: the value does not represent a temperature).

MATLAB® Save

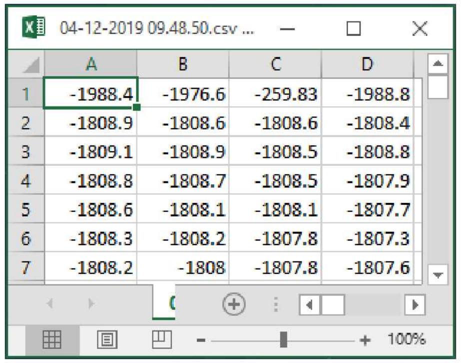
Image Save

The MATLAB Script allows the user to save and image of the figure as a Portable Network Graphic (PNG) using the saveas() function and a timestamp naming convention.



CSV Save

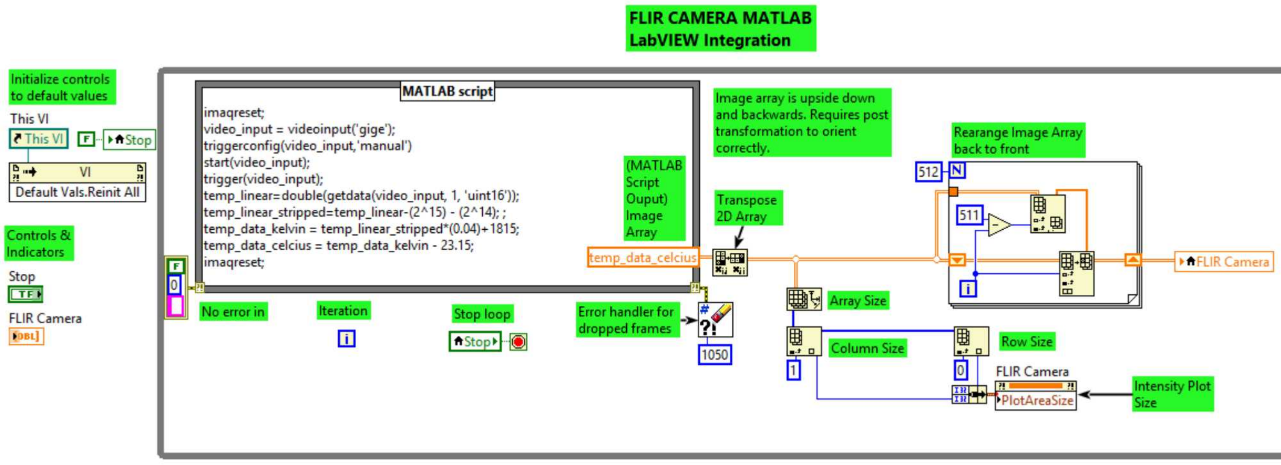
A comma delimited spreadsheet is also saved with the same using the csvwrite() function and a timestamp naming convention. Each cell in the spreadsheet represents one pixel of the 640x512 pixel image.


04-12-2019 09.48.50.csv ...

	A	B	C	D
1	-1988.4	-1976.6	-259.83	-1988.8
2	-1808.9	-1808.6	-1808.6	-1808.4
3	-1809.1	-1808.9	-1808.5	-1808.8
4	-1808.8	-1808.7	-1808.5	-1807.9
5	-1808.6	-1808.1	-1808.1	-1807.7
6	-1808.3	-1808.2	-1807.8	-1807.3
7	-1808.2	-1808	-1807.8	-1807.6

LabVIEW® Integration

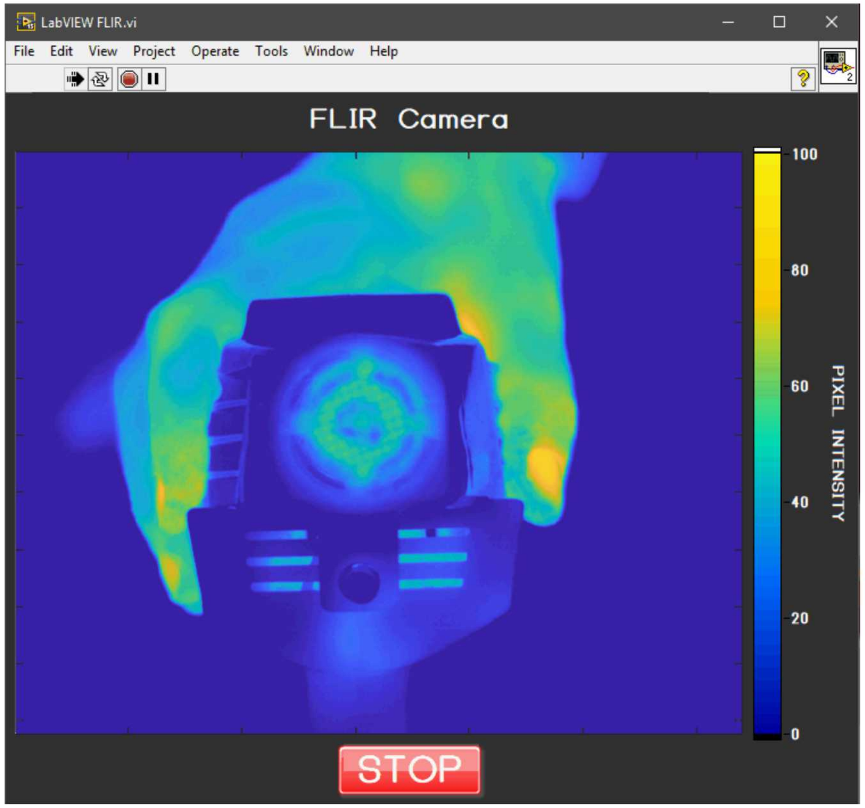
To integrate the camera operation and image acquisition into LabVIEW®, a MATLAB® Script function is used in the virtual instrument (VI) block diagram. The actual MATLAB® code used in the script is a much smaller version of the previously discussed script and contains camera communication. The obtained image array is not oriented correctly and the array must be transformed several times to display meaningful information.


FLIR CAMERA MATLAB LabVIEW Integration
Initialize controls to default values
This VI
This VI
F
Stop
VI
Default Vals.Reinit All
Controls & Indicators
Stop
FLIR Camera
DBL
MATLAB script
(MATLAB Script Ouput) Image Array
temp_data_celcius
No error in
Iteration
i
Stop loop
Stop
Error handler for dropped frames
1050
Image array is upside down and backwards. Requires post transformation to orient correctly.
Transpose 2D Array
Array Size
Column Size
1
Row Size
0
Rearange Image Array back to front
512
N
511
i
FLIR Camera
FLIR Camera
PlotAreaSize
Intensity Plot Size

```
imaqreset;
video_input = videoinput('gige');
triggerconfig(video_input,'manual')
start(video_input);
trigger(video_input);
temp_linear=double(getdata(video_input, 1, 'uint16'));
temp_linear_stripped=temp_linear-(2^15) - (2^14); ;
temp_data_kelvin = temp_linear_stripped*(0.04)+1815;
temp_data_celcius = temp_data_kelvin - 23.15;
imaqreset;
```

LabVIEW® GUI

The LabVIEW VI GUI is similar to the MATLAB GUI minus the save button. This feature will be added in future revisions of the LabVIEW VI. Also, the color bar reflects pixel intensity as opposed to the values displayed in the MATLAB image arrays.



Conclusion

- Created MATLAB® script to communicate with camera, GUI to allow for saving and camera control, and integration into the LabVIEW® environment.

- Enhanced author's MATLAB® knowledge; including image acquisition, saving data, modifying the GUI, and MATLAB® to LabVIEW® integration.

- Continued work: adding save functionality to the LabVIEW® VI, faster image acquisition, more accurate temperature conversion from output pixel data.

Questions?