

NSP32 SDK
PRISM GUI Application

User Manual
ver 1.7

nanoLambda

IMPORTANT NOTICE

nanoLambda Korea and its affiliates (“nanoLambda”) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to nanoLambda’s terms and conditions of sale supplied at the time of order acknowledgment. Customers are responsible for their products and applications using any nanoLambda products. nanoLambda does not warrant or represent that any license, either express or implied, is granted under any nanoLambda patent right, copyright, mask work right, or other nanoLambda intellectual property right relating to any combination, machine, or process in which nanoLambda products or services are used. Information published by nanoLambda regarding third-party products or services does not constitute a license from nanoLambda to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from nanoLambda under the patents or other intellectual property of nanoLambda. Reproduction of nanoLambda information in nanoLambda documents or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. nanoLambda is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions. Resale of nanoLambda products is not allowed without written agreement. Decompiling, disassembling, reverse engineering or attempt to reconstruct, identify or discover any source code, underlying ideas, techniques or algorithms are not allowed by any means. nanoLambda products are not authorized for use in safety-critical applications. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of nanoLambda products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by nanoLambda. Further, buyers must fully indemnify nanoLambda and its representatives against any damages arising out of the use of nanoLambda products in such safety-critical applications.

Table of Contents

- 1. Introduction4
 - 1.1. About Prism4
 - 1.2. Features4
 - 1.3. System Requirements5
- 2. NSP32 SDK Installation6
 - 2.1. Hardware installation6
 - 2.2. NSP32 SDK installation and Configuration7
 - 2.3. Run Prism GUI Application9
- 3. UI Components of Prism10
 - 3.1. GUI Layouts10
 - 3.2. UI Components11
- 4. Set shutter speed11
 - 4.1. Find optimal SS (Shutter speed)11**
- 4. Spectrum Data Acquisition13
 - 4.1. Intensity Level Validation13
 - 4.2. Shutter Speed (Exposure Time) Control14
 - 4.3. Find Optimal Shutter Speed with AE15
 - 4.4. Frame Averaging15
 - 4.5. Background Data Acquisition16
 - 4.6. Spectrum Data Acquisition16
- 5. Data Visualization18
- 6. Menus20
 - 6.1. 'File' Menu20
 - 6.2. 'View' Menu23
 - 6.3. 'Help' Menu24
- 7. Frequently Asked Questions25

1. Introduction

1.1. About Prism

Prism GUI is a stand-alone Qt-based GUI application from nanoLambda and implemented based on proprietary software libraries (NSP32 SDK) by nanoLambda. Prism GUI provides a convenient way for users to familiarize themselves with various functions of NSP32 spectral sensor.

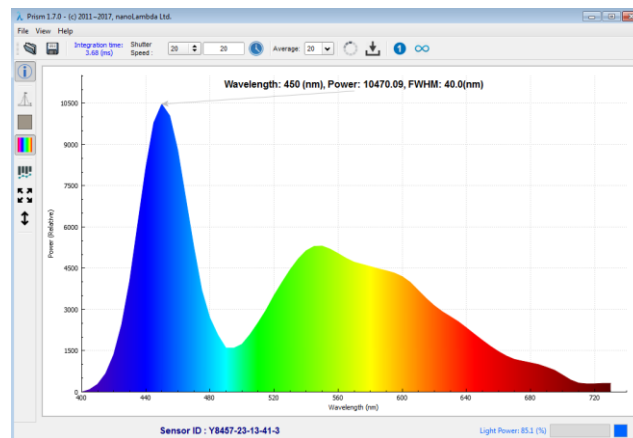


Fig 1-1 Prism GUI application

If you connects NSP32 ADK device to USB port on the user's computer with the provided USB cable and executes the Prism GUI application, Prism GUI will detect NSP32 ADK device automatically. For some platforms like Windows OS, it is necessary that the user installs the USB driver for the NSP32 spectral sensor before using it.

1.2. Features

The main features of Prism GUI s/w are :

- Spectrum acquisition function (single and continuous)
- Shutter speed (exposure time) change function
- Optimal shutter speed finding function
- Frame moving average function
- Spectrum visualization function (hollow, color-filled, and rainbow-filled spectrum graph)

1.3. System Requirements

Prism GUI application supports Windows, Linux(Ubuntu) and Mac OS operating systems. You can install Prism GUI application by unzipping NSP32 SDK package on your system . The path for Prism GUI application is '/nanolambda/NSP32_SDK/prism_gui/[platform]'. For example, you can find an executable (32-bit version) for Prism GUI application and related DLLs for Windows OS under '/nanolambda/NSP32_SDK/prism_gui/win32'.

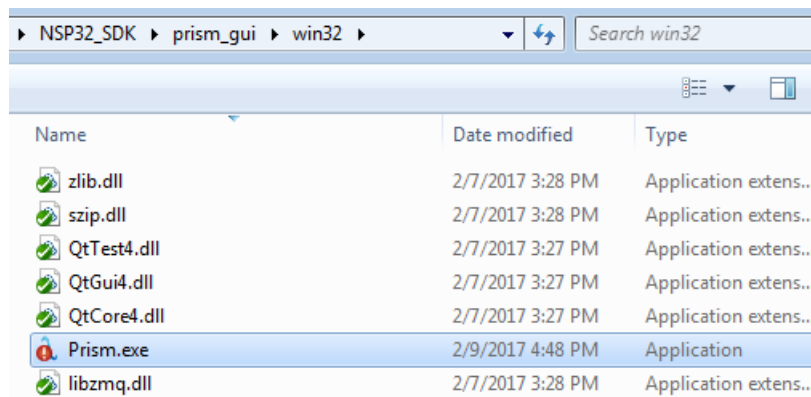


Fig 1-2 Prism GUI executable and related files

Table 1-1 Recommended System Configuration

OS	Windows: 7, 8, 8.1 and 10 (32 and 64-bit) Ubuntu: 14.04 or higher MacOS: OS X 10.10 (Yosemite) or later
CPU	Intel i3 or Higher with 1.5GHz or faster clock speed
Memory	>= 512 MB
Graphics	>= 1024x768 with 24-bit true colors
HDD	>= 500 MB free HDD space(an additional 100 MB is needed during installation only to accommodate the initial setup files)
Peripheral	1 USB port which can support USB 2.0 or higher
Other	

2. NSP32 SDK Installation

In order to acquire spectral data using the NSP32 spectral sensor and the Prism GUI application, the following two items must be checked in advance. The first relates to the installation/configuration of a communication channel to acquire data from the NSP32 spectral sensor (USB connection for NSP32 ADK device), and the second relates to the calibration data file of the NSP32 spectral sensor:

1. Is a USB driver for NSP32 spectral sensor installed on your computer?
2. Is the sensor calibration data file for your NSP32 spectral sensor in the './config' folder, a subfolder of the folder where the Prism.exe executable is located?

2.1. Hardware installation

A dedicated USB driver for NSP32 ADK device has to be installed on the your computer. About how to install USB driver for Window, please refer to "*NSP32 SDK USB Driver Installation Manual-v1.7.pdf*" document in 'doc' folder. For Mac OS and Ubuntu, you don't need to install USB driver because USB driver for NSP32 ADK device is installed by default on Mac OS and Ubuntu.

If your Windows machine does not have an appropriate USB driver for the NSP32 spectral sensor installed, the device property of your NSP32 spectral sensor will look like this. To see device property, Go to '**Start→Device and Printers**'.

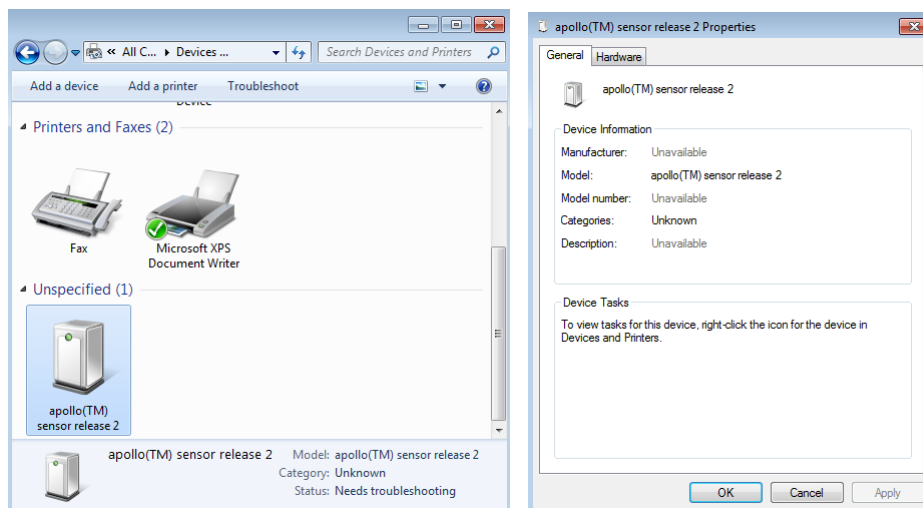
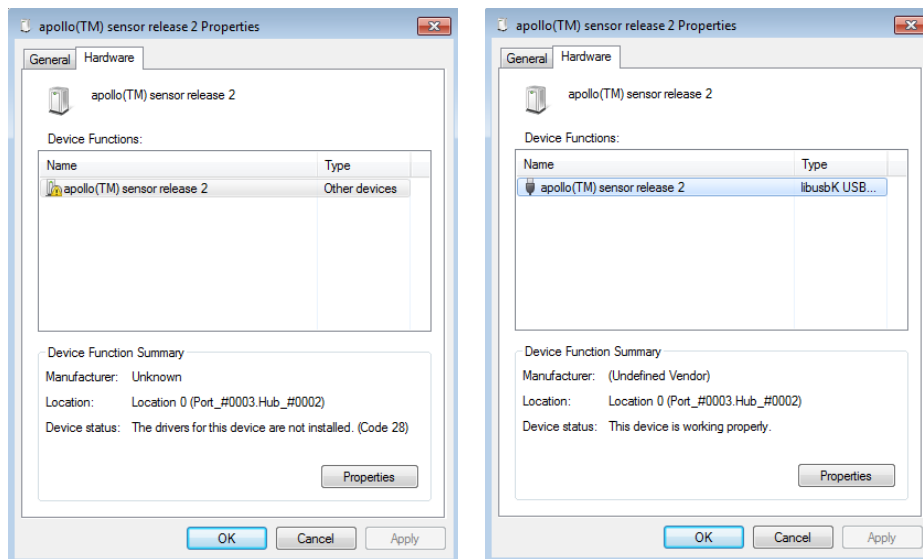


Fig 2.1 How to check device property of your NSP32 sensor

If you have installed a USB driver on your computer, you can check if the installation was successful or not as shown below. If the USB driver is properly installed, you will see the following figures (refer to '*NSP32 SDK USB Driver Installation Manual-v1.7.pdf*').



(a) USB driver is 'NOT' installed (b) USB driver installation is succeeded

Fig 2.2 USB driver installation state

If the driver installation status is different from Figure 2.2 (b), the installation of the USB driver must be failed and you have to find the cause of the failure. Please consult '*Chapter 7. Frequently Asked Questions*' to find any solution, or contact nanoLambda to have any support (support@nanolambda.net).

Note

NSP spectral sensor's device name on Windows system is used 'apollo (TM) sensor release 2' for EFM32 MCU-based or 'Presto sensor(TM)' for STM32 MCU'. However, for formal release, device name will be 'NSP32 spectral sensor'.

2.2. NSP32 SDK installation and Configuration

Installing Prism GUI application on Windows, Mac OS, and Ubuntu (Linux) platforms is very simple. The NSP32 SDK package is provided in compressed file format ("*NSP32_SDK_1_7_package.tar.gz*") so you can copy the NSP32 SDK package to your desired location and extract it.

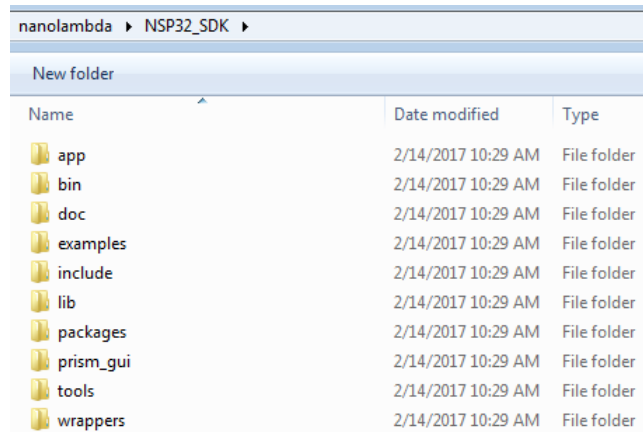


Fig 2.3 NSP32 SDK installation result

Prism GUI application is located under '*nanoLambda/NSP32_SDK/ prism_gui/[platforms]*' folder. Windows version is located in 'win32', Mac OS is 'macOS', Ubuntu version is located in sub folder named 'ubuntu'.

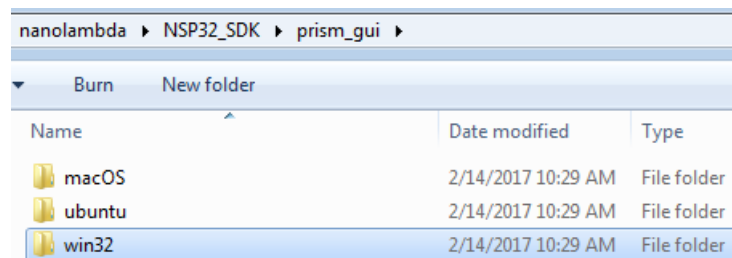


Figure 2.4 Sensor calibration data file

After installing the NSP32 SDK, you should copy the calibration data file of your NSP32 spectral sensor directly into the 'config' folder, which is a subfolder of the folder where the Prism.exe file is located. That is, if the ID of your NSP32 spectral sensor is 'Y8457-23-13-41-0', the file 'sensor_Y8457-23-13-41-0.dat' should be placed in the 'config' folder (see Figure 2.5). If you lost your calibration data file, please contact to nanoLambda (support@nanolambda.net).

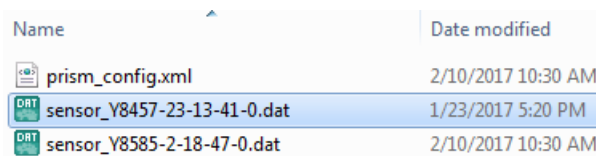


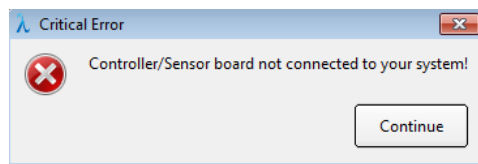
Figure 2.5 Sensor calibration data file

2.3. Run Prism GUI Application

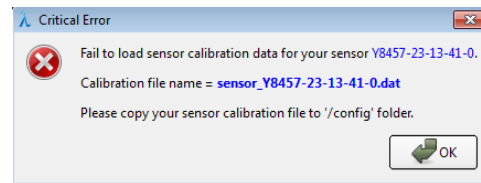
NSP32 ADK consists of a ADK device(NSP32 spectral sensor + a MCU board + F/W), a USB cable(type of USB micro B) and SDK package. NSP32 spectral sensor is connected with MCU through SPI and GPIO interfaces and MCU is connected with your computer through USB connection. When you executes Prism GUI application from your computer, Prism GUI application performs mainly two things below:

- testing NSP32 ADK device connectivity
- checking the existence of sensor calibration data file

If NSP32 ADK device is not found on any USB ports even though you installed USB driver successfully, then Prism GUI application will show a warning dialog to you and every UI components will be disabled (see Fig 2-6 (a)). If a NSP32 spectral sensor calibration data file not exists in './config' folder, then Prism GUI application will show a warning dialog to the user and every UI components will be disabled (see Fig 2-6 (b))



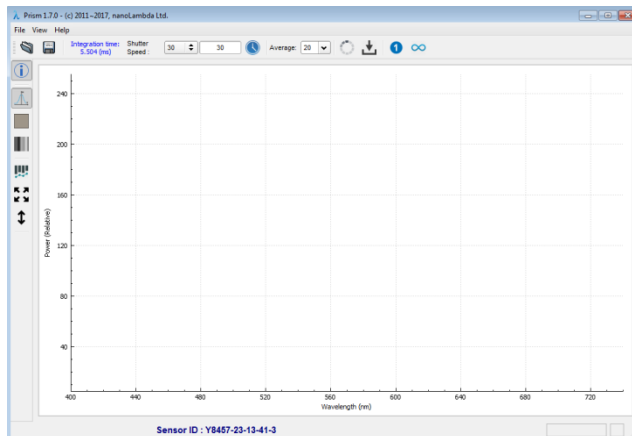
(a) Sensor connection failure happened



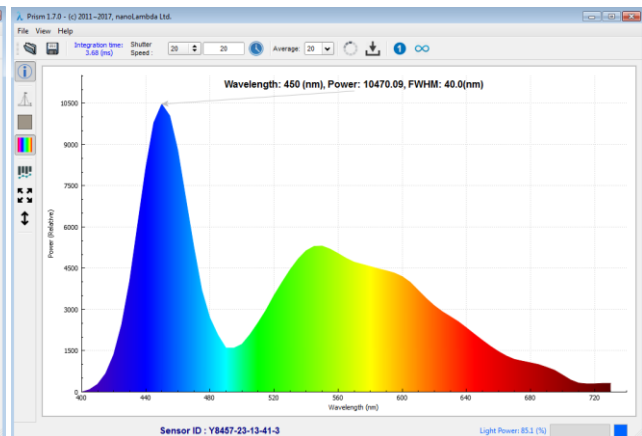
(b) Sensor calibration data file is not exist

Fig 2-6 Two warning dialog boxes for two abnormal cases

If the USB driver has been successfully installed and the sensor calibration data file for your NSP32 spectral sensor has been successfully copied to the 'config' folder, you can run the Prism GUI application without problems and you can get your first spectrum from NSP32 ADK device (see Fig 2-7).



(a) initial state of Prism GUI



(b) acquired first spectrum

Fig 2-7 Prism GUI application

3. UI Components of Prism

3.1. GUI Layouts

Prism GUI application provides simple and easy to use graphical user interface (see Fig 3-1).

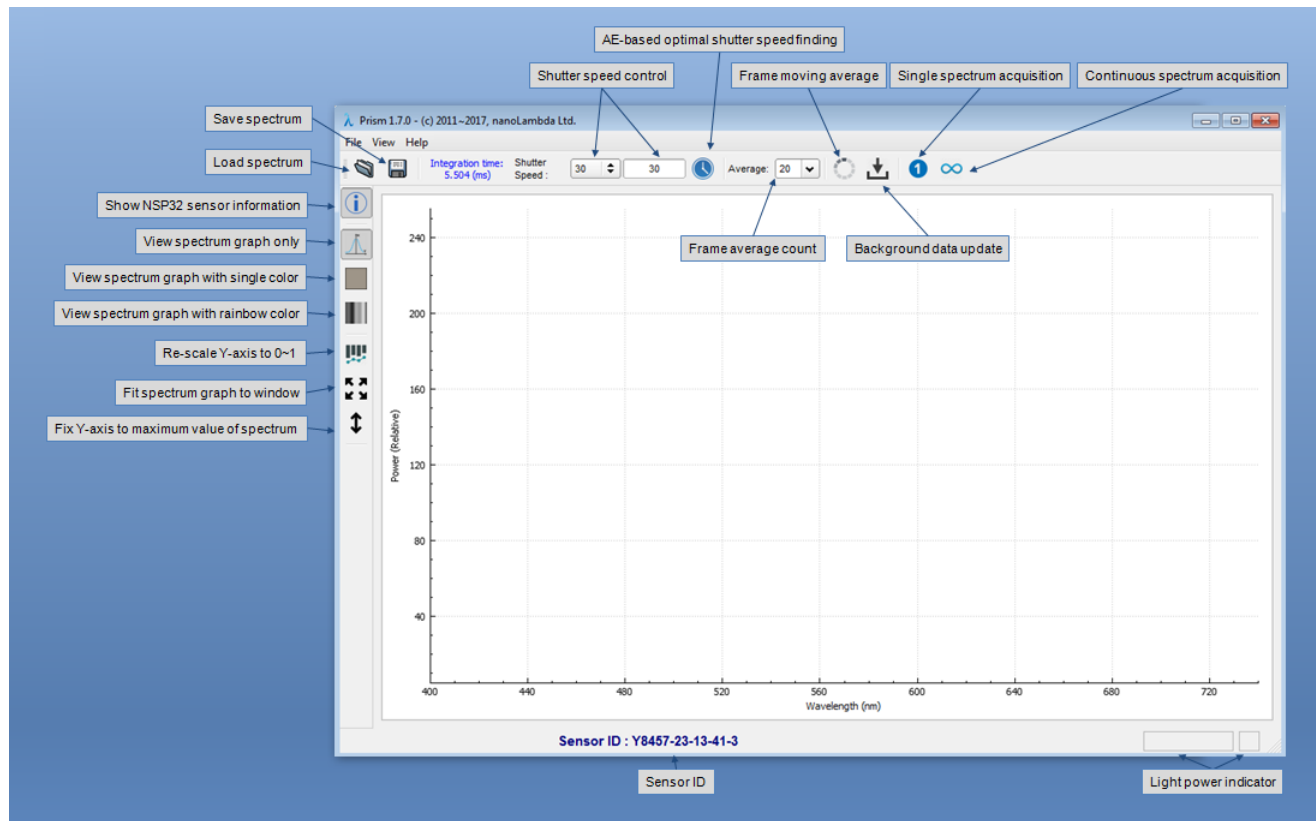


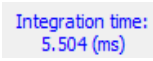
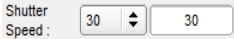

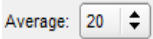






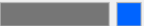
Fig 3-1 Overall layout of Prism GUI application








Initially, Prism GUI application gives a very simplified look and feel to the user with the default setting. The main GUI layout includes very basic functions such as :

- shutter speed setting and average count setting for NSP32 spectral sensor control
- spectrum data acquisition
- background data update
- spectrum data serialization
- various visualization options.

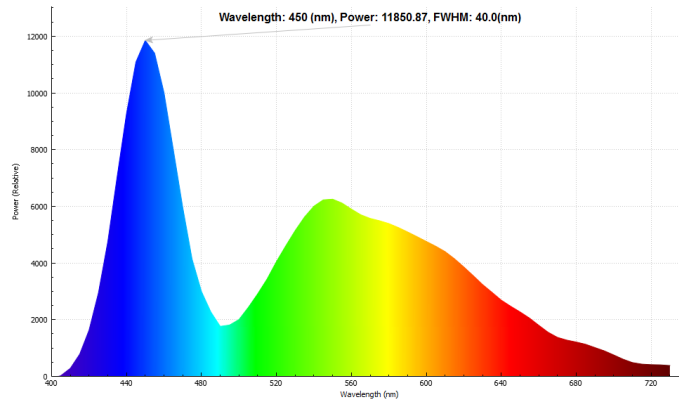
3.2. UI Components

Prism GUI application consists of 6 UI components:

Icon	Meaning	Description
NSP32 Spectral Sensor Control		
	Integration time (read only)	Calculated when SS is set
	Set shutter speed <ul style="list-style-type: none"> ● via Combo Box widget (select one shutter speed from pre-defined list) ● via Edit window (manual input) 	
	Find optimal SS (Shutter speed)	
	Set the number of frames used for averaging	
	Do frame moving average	
	Update background data	
Spectrum Data Serialization		
	Load spectrum data from file	File type: TXT, CSV
	Save spectrum data to file	File type: PDF, PNG, BMP, TXT, CSV
Spectrum Data Serialization		
	Acquire single spectrum data	
	Continuous acquisition of spectrum data	
Sensor ID and Light Condition Indicator		
Sensor ID : Y8457-23-13-41-3		Light Power: 45.9 (%) 
Spectrum Data Visualization Options		

	Show spectrum information(peak wavelength, peak power, and FWHM) on graph	
	Show spectrum graph only	
	Show single color under spectrum graph	
	Show rainbow color under spectrum graph	
	Scale spectrum to 0~1 range	
	Fit graph to window	
	Fix graph to maximum power	

Spectrum Graph Visualization



4. Spectrum Data Acquisition

4.1. Intensity Level Validation

The accuracy of spectrum data from Prism GUI application heavily depends on the sensor response to the light source. If sensor response to the light source is too weak or strong, that means, the power of the light source is too weak or too strong or the light source is too far away or too close to the sensor, in which the spectrum will not be accurate and reliable. Therefore, Prism GUI application checks the intensity level of input light source at run-time whenever a new sensor data is acquired, and notifies the user with 3 different states : ‘Low’, ‘Normal’, and ‘Saturated’.

For normal case, the power indicator’s color will be maintained with the ‘BLUE’ color (see Fig 4-1(a) and (b)). If the sensor data intensity is too low, then the power indicator’s color will change to ‘BLACK’ and spectrum graph will not be displayed (Fig 4-1(c)). If the sensor data intensity is saturated, the power indicator at the right-bottom side of main window of Prism GUI application will change to ‘RED’ and spectrum graph will not displayed (Fig 4-1(d)).

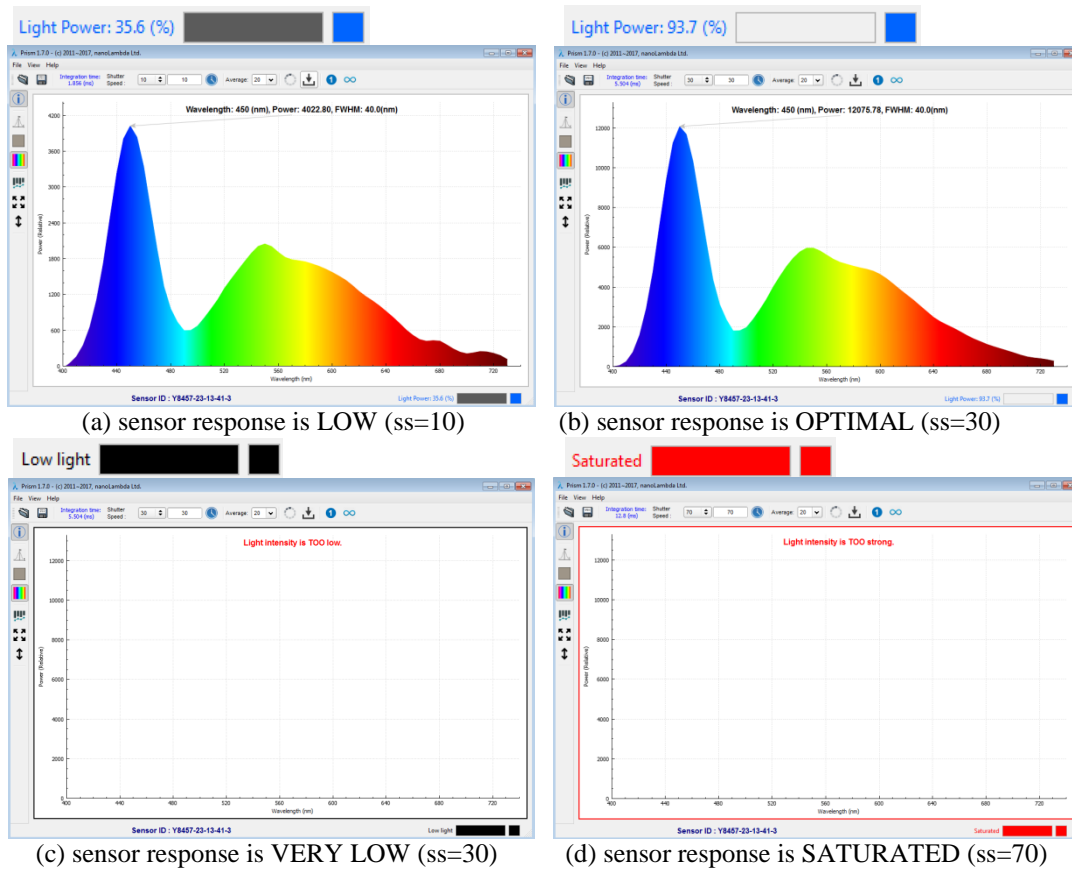


Fig 4-1 Different behaviors of Prism GUI for different light source status

nanoLambda recommends that the response level (i.e., input power level) of the NSP32 spectral sensor be maintained at **min 35% ~ max 99%** for accurate spectral acquisition. You can see the current intensity level of your NSP32 from 'power indicator' at the bottom-right side of Prism GUI application.

4.2. Shutter Speed (Exposure Time) Control

NSP32 spectral sensor provides similar functions to the image sensor of a camera such as controlling the sensitivity to an input light source. Basically, Prism GUI application controls the exposure time of NSP32 spectral sensor by shutter speed value. A specific shutter speed value can be transferred to exposure time (millisecond unit).

The user can get spectrum data with different photon counts of the light source by changing the shutter speed. There are two ways to change shutter speed in Prism GUI application. User can select a specific shutter speed value from list widget and use it for spectrum data acquisition and processing (see Fig 4-2). And user can also enter a specific shutter speed value to edit box widget and apply it to current shutter speed setting (). If shutter speed value is increased, then the integration time also increased and the time interval between mouse click for spectrum data acquisition and acquired data display on graph also increased.

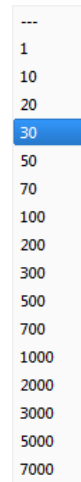



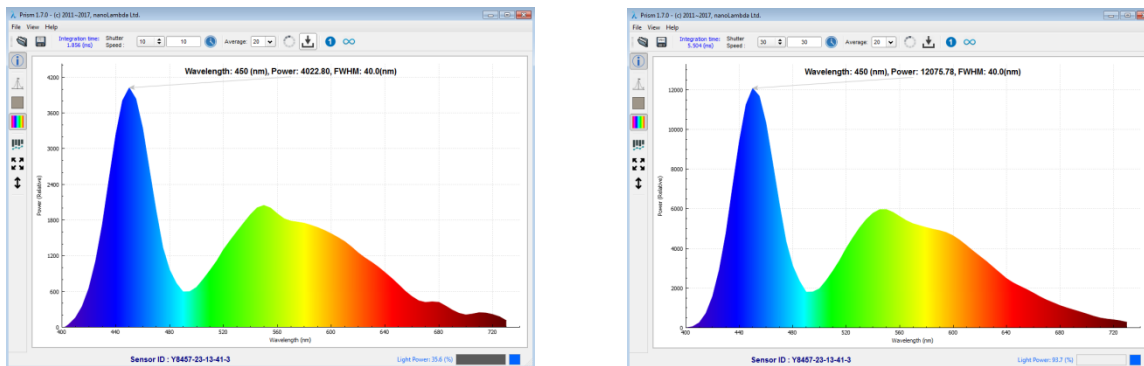
Fig 4-2 Pre-defined shutter speeds

Table 4-1 Transformation table from shutter speed value to integration time

Shutter Speed	Integration Time (ms)	Shutter Speed	Integration Time (ms)	Shutter Speed	Integration Time (ms)	Shutter Speed	Integration Time (ms)
1	0.2112	50	8.992	300	53.792	2000	358.432
10	1.824	70	12.576	500	89.632	3000	537.632
20	3.616	100	17.952	700	125.472	5000	896.032
30	5.408	200	35.872	1000	179.232	7000	1254.43
.....

4.3. Find Optimal Shutter Speed with AE

Prism GUI application provides the 'Optimal shutter speed finding' function to find optimal shutter speed for NSP32 spectral sensor. This function is similar one with auto-exposure function of camera. Optimal shutter speed is guarantee the highest SNR that NSP32 spectral sensor can give to the user . If shutter speed is too low or too high, then spectrum from NSP32 spectral sensor and Prism GUI application will not be reliable. If user clicks  button on the main toolbar, then Prism GUI application start to scan the optimal shutter speed which gives maximum SNR under the current illumination condition and changes current shutter speed with the found optimal speed. Fig 4-3 shows the benefit to find optimal shutter speed by AE. Shutter speed was 10 before the user runs the AE function(power level is 35.6%). After the user runs the AE function, the found optimal Shutter speed is 30 (power level is 93.7%). If it is necessary, user can tune this new shutter speed value manually.

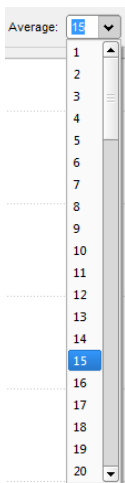


(a)Before AE (SS=10, power=35.6%)

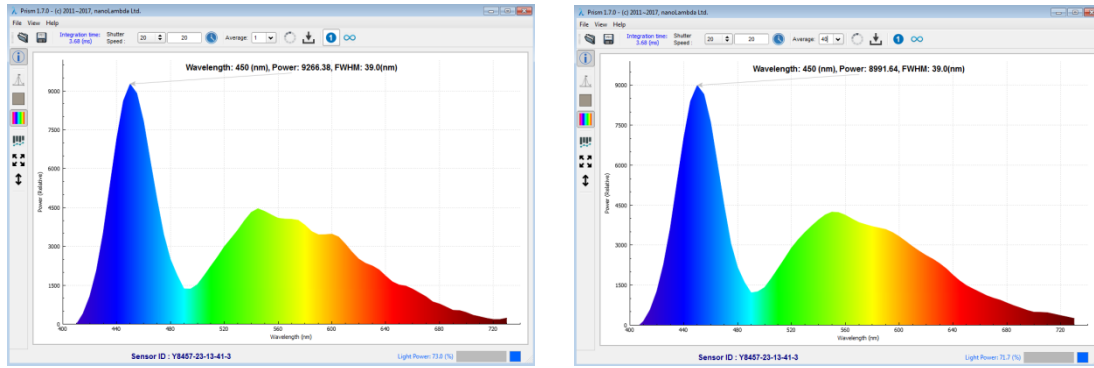
(b) After AE (SS=30, power=93.7%)

Fig 4-3 Auto-Exposure control to find optimal shutter speed

4.4. Frame Averaging



Prism GUI application provides a frame averaging function to reduce the effect of electrical noises like random shot noises and to minimize inter-frame variation of NSP32 spectral sensor. Frame averaging gives the enhanced SNR to user by reducing frame-to-frame variations due to frame fluctuations, random shot noise, and other. User can select the number of frame average from a combo-box widget on the main toolbar. The available frame average ranges from 1 to 100. If the number of frame average is increased, then the overall frame rate will be decreased accordingly. The recommended minimum number of frame averaging is 30~50, but this number will depend on your applications and user must select the appropriate number for frame averaging very carefully.




(a) Average count = 1

(b) Average count = 40

Fig 4-4 High average count can reduce spectrum variation by random noise

4.5. Background Data Acquisition

Depending on the application, the user needs to correct(remove) background signal to obtain a pure spectrum of the target signal source like LED illuminant. For background correction, Prism GUI application acquires background filter data(signal) at the initialization stage and keeps it in the internal memory buffer and uses it at run-time. But, if your measurement conditions are changed (e.g., temperature is changed significantly), then you must update your background data to acquire the correct spectrum. Prism GUI application provides a simple and easy way to update background filter data. To do this update, you need to just click 'Update background data' button() on the main tool bar. If you click that button, then one dialog box will be displayed to confirm to update the current background data with new one (see Fig 4-5). By selecting 'Cancel', you can cancel background data update.

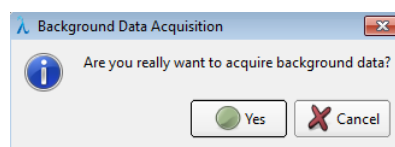


Fig 4-5 Displays a confirmation dialog box for background data update

4.6. Spectrum Data Acquisition

There are two UI components for spectrum data acquisition at the main toolbar of Prism GUI application.



Fig 4-6 UI components for spectrum data management

The user can acquire single spectrum data by clicking the 'Acquire spectrum(Single)' button(1) and can also use the 'Acquire spectrum (Continuous)' button (∞) to acquire consecutive spectrum data continuously. Of course, only one spectrum data will be displayed on the graph at one time.

If the user checks the 'Show spectrum information on the graph' button(i), the more detailed spectrum information including 'peak wavelength', 'peak power', and 'FWHM' will be displayed on the top side of the graph window. For example, you can see the detailed spectral information of white LED from Fig 4.7 that peak wavelength is 450nm, peak power is 11515.60, and FWHM is 39.0 nm.

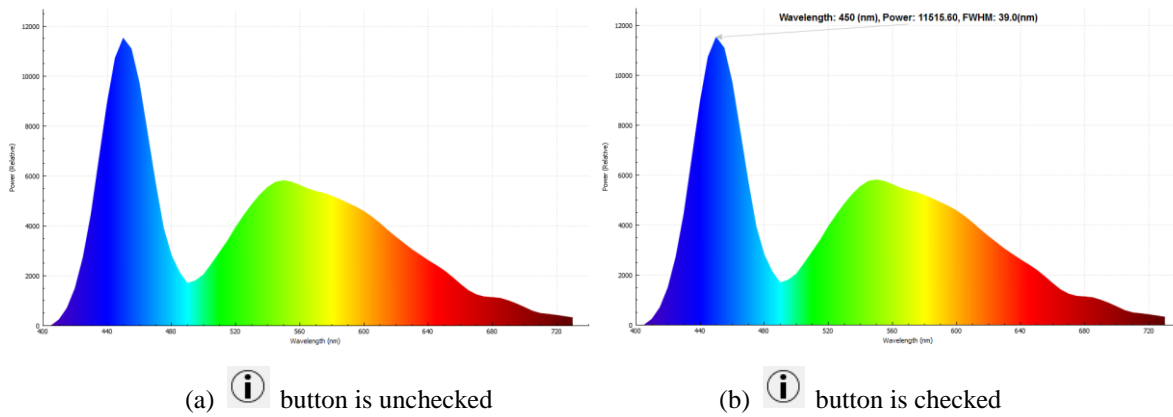
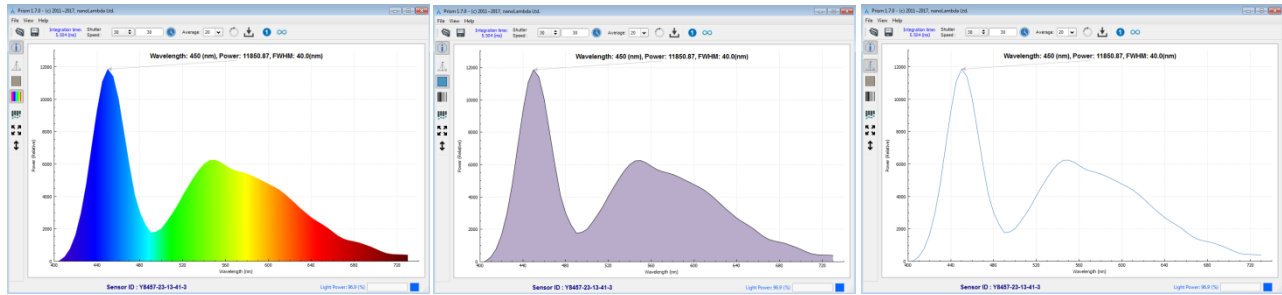


Fig 4-7 Display the detailed information of a spectrum

If you want to change sensor parameters such as shutter speed or frame average count, or to change options related to the visualization of spectrum data during continuous acquisition of the spectrum, you must first stop the spectral continuous acquisition. If you start the spectral continuous acquisition again, the spectra obtained since then will be acquired/calculated and visualized on the graph window based on the changed parameters.

5. Data Visualization

Prism GUI applications provides several convenient options for your spectrum visualization. An acquired spectrum can be visualized with 3 different modes: 1) Spectrum graph with face color (rainbow), 2) Spectrum graph with face color (solid), and 3) Spectrum graph with face color (hollow) (See Fig 5-1).



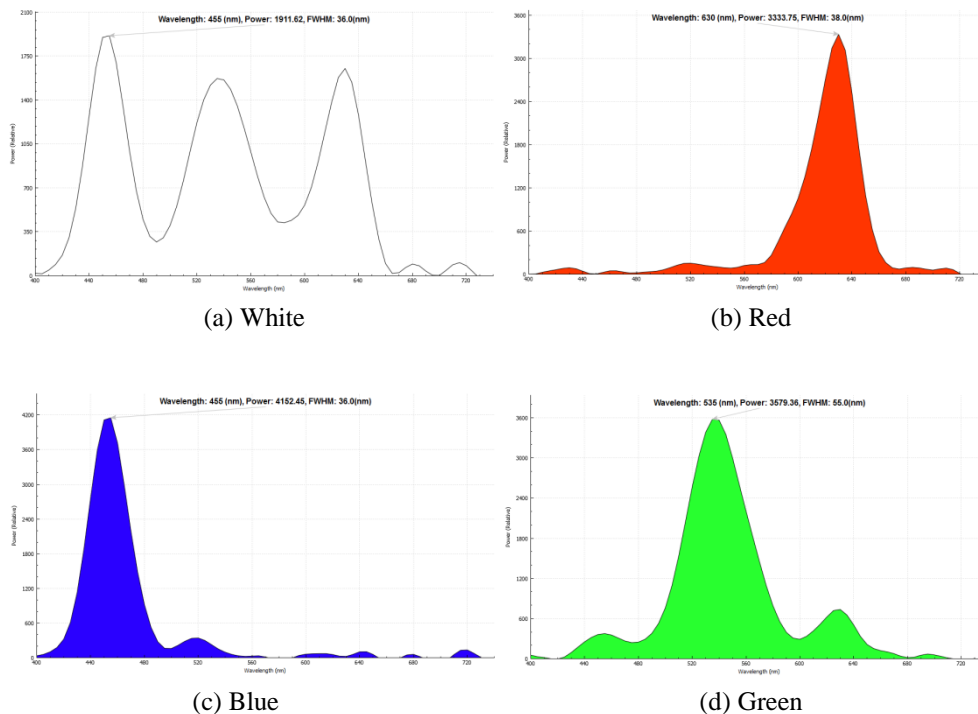
(a) Face color (rainbow)

(b) Face color (solid)

(c) Face color (hollow)

Fig 5-1 Visualization mode of spectra

For your information, if you select 2'nd mode('Spectrum graph with solid color'), then the face of spectrum will be filled with a RGB color which is calculated with current spectrum. Fig 5-2 shows two spectrum graphs of RED and BLUE color screens of Apple iPhone7 with 'solid face color' option.



(a) White

(b) Red

(c) Blue

(d) Green

Fig 5-2 Visualization of spectra of 4 different colors

Prism GUI application gives you relative spectrum power. If you want to see your spectrum with a normalized range like 0~1, you can visualize your spectrum by enabling one button ('Scale spectrum to [0~1] range') on tool bar at left side (→). Fig 5-3 shows a normalized spectrum graph of white LED light source.

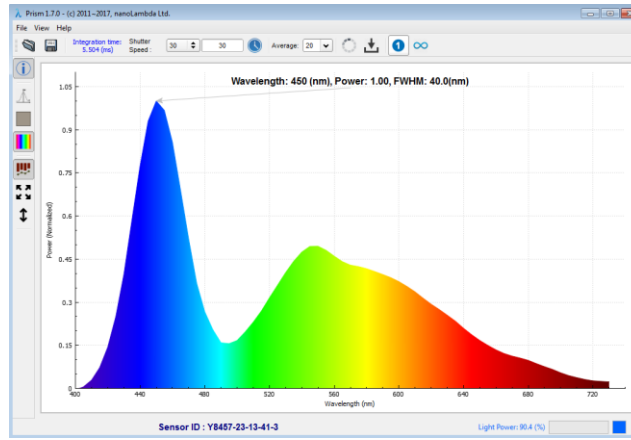


Fig 5-3 Y-axis is normalized to [0, 1] range

By default, the Prism GUI application tries to fill the spectrum into the graph window as much as possible. If the power of the spectrum drops, the spectrum will still fill the graph window. This approach can make it hard for you to notice the relative differences in the spectrum. The Prism GUI application provides an option to fix the Y-axis representing the power of the spectrum. If you enable a toolbar button ('Fit graph to window vertically') on the left side (→), the spectrum graph is fixed to the maximum value of the spectrum. If the maximum power value of the newly acquired spectrum is less than the maximum power value of the previous spectrum, the Y-axis is fixed to the maximum power value of the previous spectrum. If the maximum power value of the newly acquired spectrum is greater than the maximum power value of the previous spectrum, the Y-axis is fixed to the maximum power value of the new spectrum. This allows you to visually and easily observe the difference in power between successive spectra (see Fig 5-4).

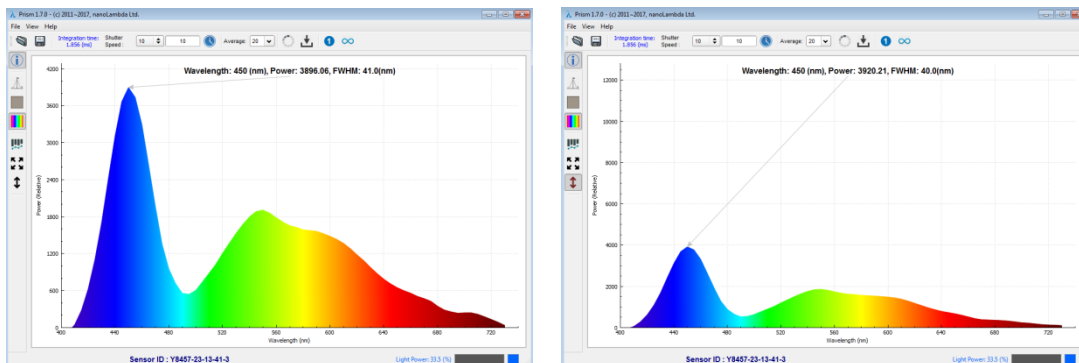
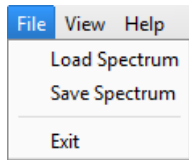


Fig 5-4 Y-axis control: (LEFT) variable mode, (RIGHT) fixed mode to maximum value


6. Menus

6.1. 'File' Menu



The menu of the Prism GUI application is very simple. There are only two menus, 'File' and 'View'. The 'File' menu has two menus related to serialization of spectral data: 'Load spectrum data' and 'Save spectrum data', and a menu ('Exit') for terminating the Prism GUI application. The 'View' menu contains a menu ('Show NSP32 sensor info.') that enables the docking widget to show details of the NSP32 spectral sensor and another menu('Show spectrum table') that displays spectrum data in table format .

Save Spectrum To File

You can save spectrum data by clicking the button() and specifying file name using any of the 5 different file formats (PDF, PNG, BMP, TXT, and CSV).

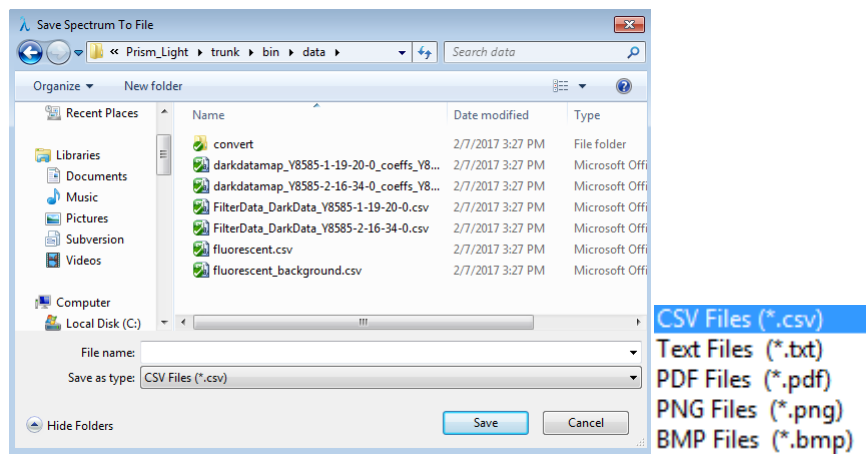


Fig 6-1 'Save Spectrum To File' dialog window

In the case of PDF, PNG and BMP format files, the graph area in Prism GUI application will be captured and saved into PDF format or image format files. More specifically, the detailed information about the spectrum acquisition conditions like sensor ID, shutter speed, frame average count, and time stamp doesn't saved into PDF,

PNG and BMP files. So, if you want to save the detailed information and use it later, CSV or TXT file formats are well fit for your purpose.

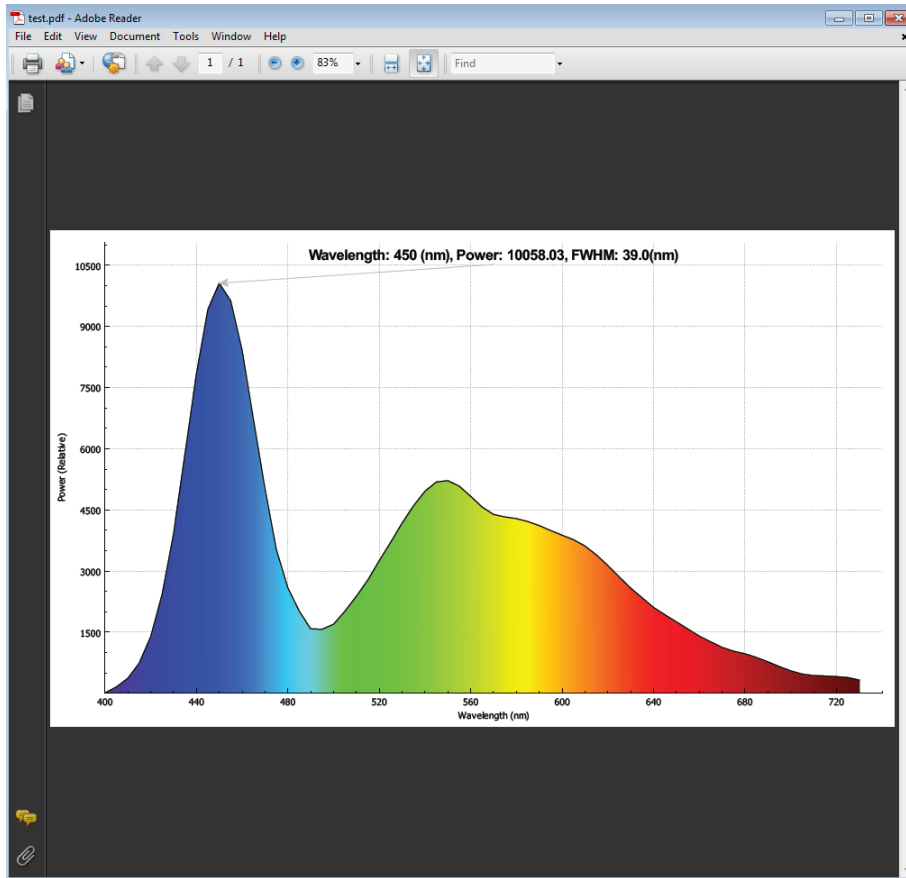


Fig 6-2 Spectrum graph in PDF file

Note


Users using the latest Windows operating systems, such as Windows 7 or 10, may fail to save spectral data to a file using the Prism GUI application. This is a problem related to the execution rights of the application, so nanoLambda recommends that you always run Prism GUI application with Administrator privileges. You can save spectral data as a file without problems with administrator privileges.

The saved file includes the additional information as well as spectrum data (see Fig 6-3).

#date	2017-02-14-1656									
#sensor id	Y8457-23-13-41-0									
#shutter speed	30									
#exposure time	5.504									
#wavelength(nm)	400	401	402	403	404	405	406	407	408	
#power	0	0	0	0	0	12.8482	57.8407	102.833	147.826	
#filterdata	111.35	111.35	112.95	112.6	115.75	104.65	103.25	110.25	105.65	

Fig 6-3 File structure of the save spectrum

Load Spectrum From File

Also the user can load spectrum data from file ((CSV and TXT formats) by clicking the button () and selecting a file from the dialog window ('Load Spectrum From File') to visualize it as a graph. Fig 6-4 and Fig 6-5 shows one spectrum data in 'table_stand_spectrum.csv' file is loaded and displayed on Prism GUI application.

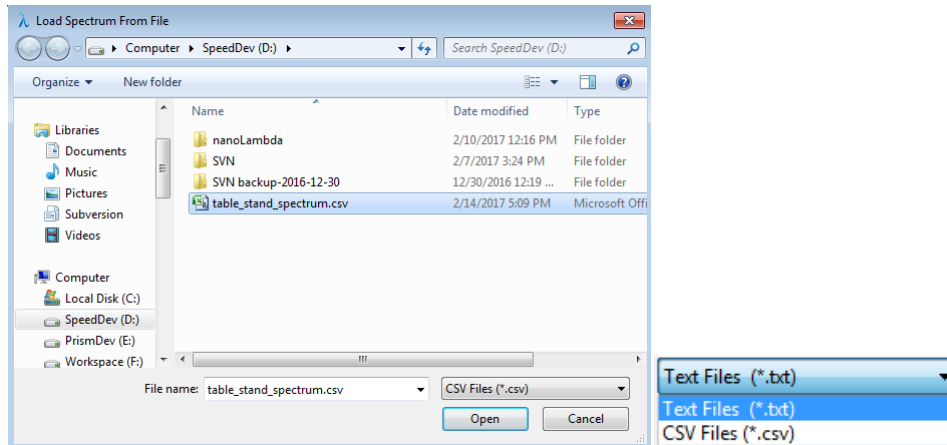


Fig 6-4 'Load Spectrum From File' dialog window

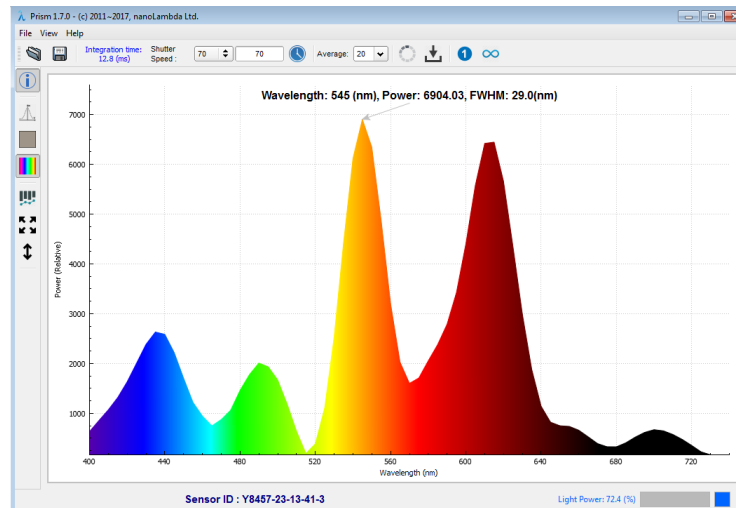


Fig 6-5 Loaded and displayed spectrum data file('table_stand_spectrum.csv')

Exit From Prism

If you select 'Exit' sub-menu from 'File' menu, then you'll get a dialog window like Fig 6-6. If you click 'Yes' button on a dialog window, Prism GUI application will be terminated. If you click 'Cancel', you'll return to Prism GUI again.

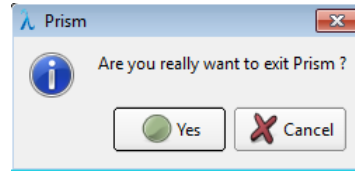
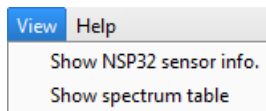


Fig 6-6 A dialog window for 'Exit' conformation

6.2. 'View' Menu



The 'View' menu contains a menu (*Show NSP32 sensor info.*) that enables the docking window to show details of the NSP32 spectral sensor and another menu (*Show spectrum table*) that displays spectrum data in table format. If you select *Show NSP32 sensor info.* menu, then one docking window will be activated at the left region of Prism GUI application (See Fig 6-7). You can see more detailed information about your NSP32 spectral sensor from that docking window (sensor ID, wavelength range, and current sensor settings like shutter speed, integration time and frame average count). If you change shutter speed and/or frame average count from main tool bar at run-time, then those values in the docking window also will be updated accordingly.

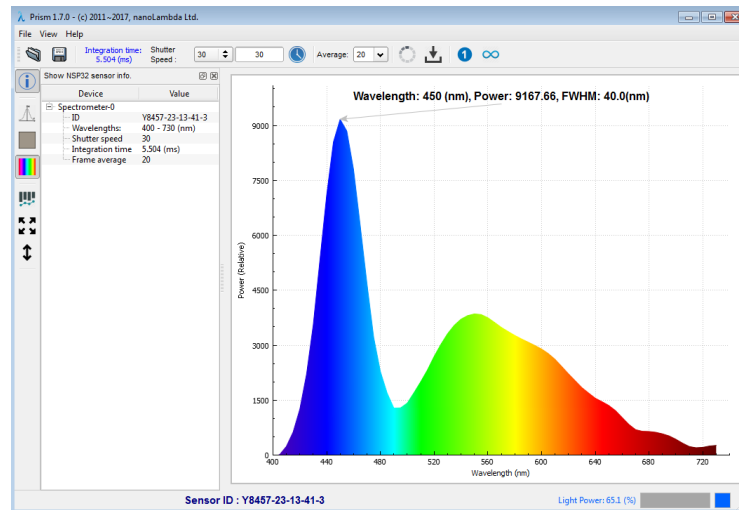


Fig 6-7 Docking window for NSP32 spectral sensor information

If you select 'Show spectrum table' menu, then another docking window will be displayed at the right side of Prism GUI application (see Fig 6-8). You can see two numerical values of a current spectrum in the table: one is wavelength and another is relative power value. From this docking window, you can save current spectrum data in the table to file if it necessary.

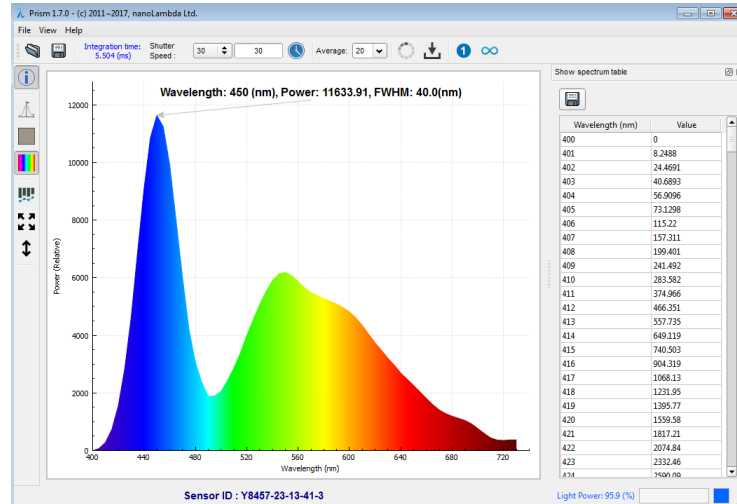
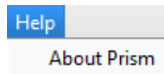


Fig 6-8 Docking window for spectrum data table

6.3. 'Help' Menu



When you clicks the 'About Prism' sub-menu from the 'Help' menu, then the 'About Prism' dialog box will pop up with a brief description of Prism GUI application.



Fig 6-8 'About Prism' dialog box

7. Frequently Asked Questions



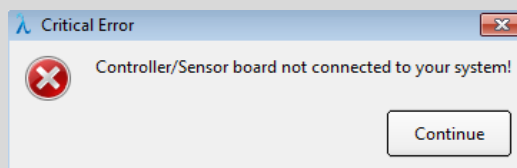
I want to get spectrum data for my application using Prism and NSP32 ADK. What is the recommended procedure?

Regardless of the type of application, the following procedure is recommended. It is assumed here that your application measures the LED spectrum.

1. **Light intensity control** - The NSP32 spectral sensor is very similar to the camera. That is, enough light is needed to get an accurate spectra (image). There are two ways to provide enough light for the NSP32. You can use both of these methods at the same time: 1) adjust the intensity of your LED light source, 2) adjust the distance between the LED light source and the NSP32 sensor.
2. **Adjusting shutter speed of the NSP32** - You can adjust shutter speed to exposing NSP32 to the incident photons. Increasing this shutter speed will increase the output level of the NSP32, and decreasing shutter speed will decrease the output level. You can set/adjust the shutter speed manually. However, this method is time-consuming to find the optimal shutter speed for your application. The more accurate the shutter speed of the NSP32 is, the more accurate the spectrum can be obtained. Therefore, it is recommended to use the '*Find optimal shutter speed*' function provided by Prism GUI application at the beginning of the experiment using NSP32 ADK device. The '*Find optimal shutter speed*' feature on the Prism toolbar automatically finds the optimal shutter speed for you. The NSP32 spectral sensor with optimal shutter speed guarantees 90 ~ 99% intensity level of the saturation level.
3. **Frame average count adjustment** - NSP32 spectral sensor acquires raw filter data by snapshot method instead of scanning method. Prism and the NSP32 SDK provide the ability to set the frame average count in order to minimize the effects due to the sensor's own electrical noise or external environment and to improve the signal-to-noise ratio. You can get a low noise spectrum by setting this frame average count to a sufficiently high value (recommended value is 40~50).



If I run Prism GUI application, there is no sensor connected. How can I fix this situation?



Make sure that you have installed the USB driver for the NSP32 ADK device for Windows machine or check if a USB connection to NSP32 ADK device is build or not. In Windows, you can see the following from the 'Start→Devices and Printers' menu. If you do not have a USB driver installed, you need to install the USB driver using the manual below.



The USB driver is installed but the NSP32 sensor is not detected.

There may be a problem with the USB connection. There are two possible reasons.

1. The product quality of USB cable (USB Micro B type) included in the NSP32 ADK could affect to the USB connection. For this case, try connecting the NSP32 ADK device to another USB cable to see if that resolves the problem.
2. If replacing the USB cable does not solve the problem, please check the power of USB port or hub. Especially for embedded computer like RaspberryPi, if you connect NSP32 ADK device to RaspberryPi directly, then power from Raspberry to NSP32 device would not enough. For this case, please use a USB hub, connect NSP32 device to USB hub, and use a dedicated power adapter for your USB hub.

Even with these trials, problem doesn't solved, it is possible that the NSP32 spectral sensor has failed. If this is the case, contact nanoLambda immediately (support@nanolambda.net).



Spectral data looks different with the desired one or distorted

Check the light intensity or shutter speed. You have to make sure that the power indicator shows a power level within 3~99% range. At the beginning, try to use AE to find optimal shutter speed. If the input power to NSP32 spectral sensor or NSP32 ADK device is too weak, it'll take several tenth seconds to find optimal shutter speed by Prism GUI application(actually, NSP32 SDK funtion) and will return very large number as an optimal shutter speed. For this case, it means that you need to adjust your setup of light source or optical source. For example, reduce the optical path between your signal source and NSP32. After then, you can try optimal shutter speed finding with AE again. If shutter speed value is less than 300, then you will acquire most promising spectrum for your application.



The acqired spectrum shows large variation(fluctuated).

The major reason of the variation in spectrum is electrical random noise in NSP32. Of course, if there is any flickering signal/optical source in your measurement/experiment environment (e.g.,

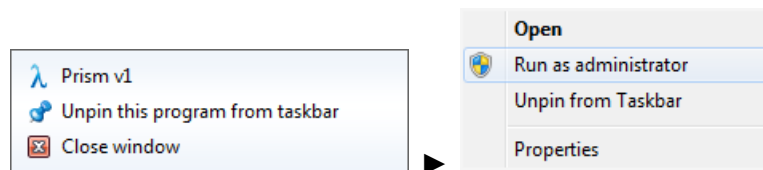
fluorescent lamp), then the temporal variation of spectrum is natural. So, to acquire stable spectrum data, you have two options (non-exclusive each other):

1. Block NSP32 from the ambient source as many as possible.
2. Increase frame average count (nanoLambda recommends to use 40~50 for frame averaging).

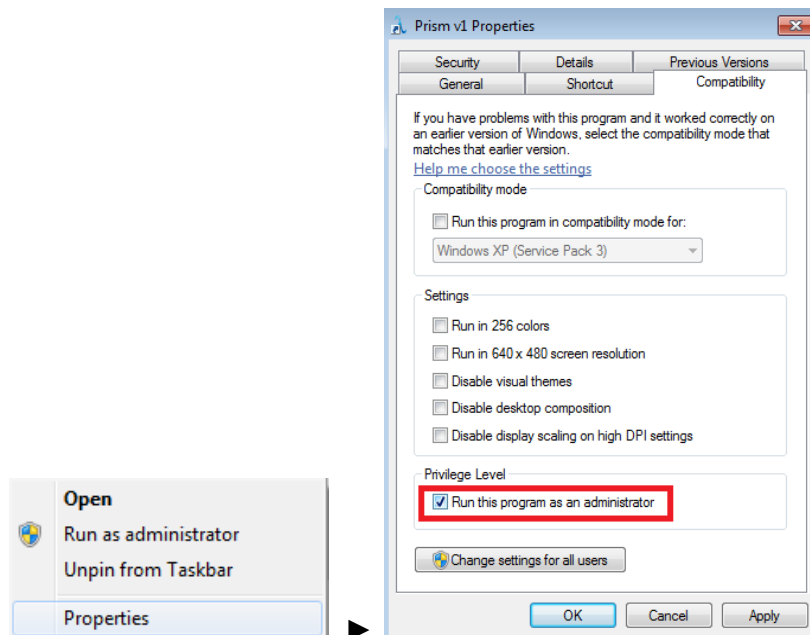


The spectrum is not saved to a file.

Sometimes, Windows 10 users have difficulty storing spectral data to file. This is caused by a problem with you's authority. To avoid this problem, you must run 'Prism.exe' with administrator privileges, or store the spectral data on a drive other than the drive on which Windows is installed.



You can always run Prism as an administrator with the following settings.



Too slow response of AE

It's depends on the intensity of your illuminant or optical signal source. Too low illuminant power(intensity), takes time to find optimal shutter speed and vice versa.



Is NSP32 calibrated?

Yes. Wavelength and power are calibrated to the NIST traceable radio-photometer. You need to note that the power have relative unit.



What's background filter data and when I need to use 'background filter data update' function?

Background filter data are the output of 'monitoring filters'. NSP32 SDK and Prism GUI application use this background filter data to compensate background signal before spectrum reconstruction. This is very important data for spectrum quality, but in normal situation, you don't need to care about this. There is a typical situation where you must update your background filter data by yourself. If the temperature in your test/measurement environment are changed or continuously changing, then you must update old background filter data with new one before acquire spectrum.

1. If your environment is thermally stable, then you need to update your background filter data just one time.
 - ▶ Update background filter data → acquire 1'st spectrum → acquire 2'nd spectrum → acquire N'th spectrum
2. If your environment is thermally unstable and temperature is changed randomly, then you must update your background filter data whenever you try to acquire one spectrum.
 - ▶ Update background filter data → acquire 1'st spectrum → update background filter data → acquire 2'nd spectrum → update background filter data → acquire N'th spectrum



Don't we need to cover NSP32 sensor before update background filter data with new one?

YES. The NSP32 acquires and compensates background data in a different way than an off-the-shelf commercial spectrometer. NSP32 spectral sensor has several monitoring filters what are sensitive to the thermal/temperature changes in the environment. Those filters doesn't affected with optical input. To use 'Background filter data update' function, you don't need to cover NSP32 sensor with something to make NSP32 totally dark.