operating manual

hyperion
colorimeter

admesy
ADVANCED MEASUREMENT SYSTEMS

admesy
ADVANCED MEASUREMENT SYSTEMS

Contents

1 Introduction

1.1 Hyperion

The Hyperion series colorimeter offers a unique combination of high speed and accurate colour measurement capabilities packed in a robust package.

The improvement compared to the previous colorimeters is that the Hyperion has a significant improvement on filter characteristics and an incredible speed upgrade. It is actually 4x times faster than our previous models making accurate colour measurements possible in 50ms at 0.3cd/m^2.

The Hyperion colorimeter is available with a 10mm spot size. A fibre version is also available with several optics, custom optics can be applied on request. Added to the filter characteristics the high sensitivity, ultra-low noise electronics and a huge dynamic range make it the ideal device for display measurements even at low levels.

1.2 Hyperion highlights

- Highly accurate colour measurement according to human eye (CIE1931)
- Fast colour measurement even at low luminance level
- Flicker luminance (Y) function: 2000 samples/second.
- Auto-range function
- Powerful MCU enables internal JEITA flicker calculation
- Mechanical shutter
- USBMTC standard compliant
- Windows, Linux and MAC OSX compatible
- Directly supported in Labview, Labwindows, Visual Studio via VISA library

1.3 Standards

The Hyperion is compliant to the USBTMC standard and can be used in combination with external provided USBTMC compliant drivers. Currently it has been tested on Windows, Linux and Apple OSX using NI VISA (www.ni.com/visa) and using the open source drivers on Linux (i686, x86_64 and ARM). Refer to the Admesy support site for a more detailed description and free source code or contact us.

2 Interfaces

2.1 USB interface

The USB-B connector is used to connect the Hyperion to a PC/Laptop. The Hyperion uses the USBTMC class protocol and can therefore be used directly with third party provided VISA compliant libraries like NI-VISA. Hyperion can be used USB powered in case the host provides enough current. Normally every USB 2.0 host should be able to drive 500mA. Non-powered USB-HUB's mostly do not supply enough current. It is therefore recommended to use powered USB-HUB's only.

2.2 RS232 interface

RS232 is provided to connect the Hyperion Colorimeter to any host that doesn't provide USB or for which no USBTMC drivers exist. Using RS232, the high speed options of the colorimeter are still available, only transfer of data to the host is reduced in speed. It is recommended to use USB in case the high speed sampling options are needed.

Baud rate	Data bits	Parity	Stop bits	Flow control	Termination character
115200[1]	8	None	1	None	LF='\n'

Table 1 RS 232 port configuration.

1 Baud rate can be changed.

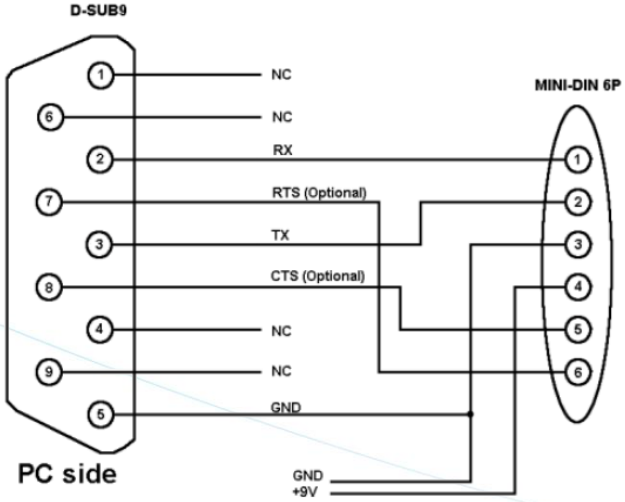


Fig 1 Hyperion RS232 connection.

Fig 2 Rear side mini-DIN-6 pin connector.

2.3 Trigger in/out

The Hyperion colorimeter has two trigger connections. One trigger output and one trigger input. The type of connectors is SMA. When triggering is enabled, the trigger output line will be set to a high level once the measurement has finished and the measurement result is available. It will stay at a high level until the next command is carried out, but has a minimum high level of 5µs. A trigger will carry out the last send command and send the result to the host via the selected interface. Supplied code examples show how to use this feature in an application. The trigger output line is used to indicate that the measurement is ready. Trigger signals should comply with the following Timing (Fig 3).

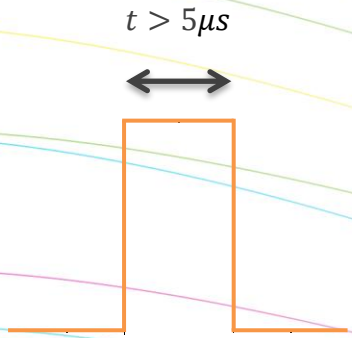


Fig 3 Trigger-in timing.

Trigger pulses arriving faster than the Hyperion can measure will be ignored, but it may slowdown overall performance. Trigger pulses should not arrive faster than the measurement takes to complete. The best way is to use the trigger output to make sure measurement was finished.

The trigger out port provides a continuous high signal when the device is in use. Its signal turns low when the device is not executing any commands.

2.4 Power connections

The Hyperion should be connected to USB with enough supply or using a 9V DC power supply to the RS232 connector. When using RS232 the colorimeter needs to be powered via an external adapter. This can be done through the mini-DIN connector as illustrated in the RS232 connections. The unit shall be powered by a 9V DC voltage or via a standard USB PC-port, reinforced separated from Mains, with a limited energy of < 150VA and < 8A.

Connection	Min. Voltage	Typ. voltage	Max. voltage	Max current
USB powered	4.75 V	5.00 V	5.25 V	300mA
DC powered	8.50 V	9.00 V	15.00 V	300mA

Table 2 Power supply levels.

3 Communications protocol

3.1 USB

The Hyperion colorimeter can be connected to any USB host. The colorimeter is a USBTMC compliant device which is a standard USB class device and device drivers for this class are available for most popular operating systems (also embedded). This makes the Hyperion colorimeter directly usable in popular programming languages like NI's Labview and Labwindows but also C++, Visual basic, C#, Java etc. The Hyperion colorimeter has two interfaces build in, which require a different device driver to be used.

- Admesy bootloader
  (USB RAW device driver Vendor ID : 0x23CF, Product ID 0x0109)
- Hyperion colorimeter
  (USBTMC device driver Vendor ID : 0x23CF, Product ID 0x1081)

When the Hyperion colorimeter is connected to the host, it will start the Hyperion colorimeter firmware. As soon as the firmware is idle to receive commands, the Power LED goes to the on state. The Admesy bootloader is a RAW USB device and in order to use this device in Windows, a driver must be installed which is supplied by Admesy. Besides upgrading to new firmware, it is also allowed to downgrade firmware in case this is required. Note that older firmware also may require the use of older software libraries and/or executable versions of software. The Hyperion colorimeter is USBTMC compliant and can be used with libraries that contain a USBTMC compliant driver like NI-VISA. The Hyperion colorimeter is a USB 2.0 High Speed device. In case a USB host is detected, it is assumed that the Hyperion colorimeter operates only via USB.

3.2 RS232

Hyperion commands are equal for all interfaces. Note that for high speed transfers it is best to use USB.

4 Device drivers

4.1 USB

The following table shows an overview of USB support on various operating systems.

OS	NI-VISA	Libusb	Native kernel	Agilent USBTMC
Windows XP [2]	▪	▪	Not available	Not tested[1]
Windows VISTA	▪	▪	Not available	Not tested[1]
Windows 7	▪	▪	Not available	Not tested[1]
Windows 8(.1)	▪	Not tested[1]	Not available	Not tested[1]
Windows 10	▪	Not tested[1]	Not available	Not tested[1]
Windows CE	▪	Not tested[1]	Not available	Not tested[1]
Apple OSX PPC	▪	Not tested[1]	Not available	Unknown
Apple OSX Intel	▪	Not tested[1]	Not available	Unknown
Linux i386 (32bit)	▪	▪	▪	▪
Linux i386 (64bit)	▪	▪	▪	▪
Linux ARM	Not available	▪	▪	▪
Linux other	Not available	▪	▪	▪

Table 3 Supported operating systems.

1 Not tested: Available, but not tested by Admesy, 2 Native Kernel: Driver included with OS.
2 Windows XP SP3 is supported: Windows official support has ended as of April 8 2014

Admesy supports all tested platforms but does not provide standard applications on all platforms. The table is provided to show the possible platforms for software development. Admesy does however provide software examples for most of the tested platforms. Most of these examples can be found on our support web page.

4.2 RS232

When no USB driver is available or the host system does not provide USB, RS232 can be used as it does not require additional drivers for the Hyperion.

5 Command set description

5.1 Commands

The functions of the Hyperion can be best described via the following categories.

- System commands
- Configuration commands
- Measurement commands
- User EEPROM commands

The Hyperion uses SCPI like commands for control and measurement. These are ASCII based commands and follow specific rules regarding syntax. Although the Hyperion uses SCPI like commands, they deviate from the SCPI standard.

5.2 Command structure

Every command starts with a colon ":", which identifies the root of the command tree. Each further keyword is also separated by a colon. In case parameters need to be specified, the last keyword and parameters are separated by a single space character. In case more than one parameter needs to be specified, the parameters need to be separated by a comma.

The command tables show commands in long and short format. The short format is specified by upper case characters. It is allowed to use long and short format or a mixed format. Optional keywords are shown between brackets [...]. Commands are not case sensitive, so it is allowed to use both or a mix of upper and lower case. The command structure is valid for all communication interfaces of the Hyperion. It is recommended to terminate a command by a newline character "\n".

Command table	Valid command syntax examples	Notes
:SENSe: INT 50000	:sens:int 500000 :sense:int 500000 :SENS:INT 500000 :SENSE:INT 500000	Sets the integration time of the Hyperion to 50ms
:MEASure:XYZ	:measure:XYZ :measure:xyz :meas:XYZ :MEASure:XYZ	The measure commands use the averaging and gain options
:SAMPle:Y	:sample:Y :sample:y :samp:Y :SAMPle:Y	the sample command, the Hyperion will perform fast sampling to its internal memory. Results are read from the memory after the measurement

Table 4 Example commands.

5.3 System commands

The following commands can be used to control the Hyperion or read back information.

Command syntax	Parameters	Purpose
:*CLS	None	Clear status
:*IDN?	None	Identification query
:*RST	None	Reset Command
:*STB?	None	Read Status Byte query
:*TST	None	Self-test query
:*FWD?	None	Firmware date query
:*FWT?	None	Firmware time query
:SYSTem:VERSion?	None	Get system version information
:SYSTem:ERRor?	None	Retrieve the last occurred error
:SYSTem:ERRor:NEXT?	None	Retrieve previous errors

Table 5 System commands.

5.4 Configuration commands

Configuration commands are used to set parameters of the Hyperion colorimeter that are used by the measurement functions. The settings are used globally by other measurement functions. The integration time setting can be varied from 0.5ms to 1s. It is specified in µs. Results from the Hyperion include a clip and noise indication, which indicate whether the measured light is too bright (clip) or too low (noise). When clipping is detected, the resulting colour will not be correct and a lower integration time should be chosen. When noise is detected, a larger integration time should be chosen.

Command syntax	Parameters	Range	Purpose
:SENSe:INT	Int time	500 – 1000000	Set integration time (µs) of the colorimeter
:SENSe:INT?	None		Returns the current integration time setting of the colorimeter
:SENSe: AVERage	Averaging	1 – 200	Set the averaging
:SENSe: AVERage?	None		Returns the averaging setting
:SENSe:GAIN	Gain	1 – 3	Set the gain for measure commands is auto-range is not used
:SENSe:GAIN?	None		Returns the current gain setting
:SENSe:SBW	Calibration matrix	“off” “factory” “user1” – “user 30”	Set calibration matrix for the colorimeter
:SENSe:SBW?	None		Query selected calibration matrix
:SENSe: AUTORANGE	Auto-range	0 – 1	Set auto-range mode
:SENSe: AUTORANGE?	None		Returns current setting
:SENSe: SHUTter	State	0 – 1	Set shutter open/close 0 = open, 1 = close
:SENSe: SHUTter?	None		Returns shutter state
:SENSe: REALINT?	None		Returns the real integration time used in autoranging mode
:SENSe:WHITE	String[4]	“D65”, “A” etc (see table: 8)	Used for Dominant wavelength
:SENSe:WHITE?	None		Gets the currently configured white point
:SENSe:TRIG	Trigger	0 – 1	Set trigger mode (note, this one is not stored in EEPROM)
:SENSe:TRIG?	None		Returns the current setting

Table 7 Sense commands.

5.5 White point references

Reference white	Y	X	y
" A"	100.0000	0.447587	0.407454
"D65"	100.0000	0.31271	0.329022
"C"	100.0000	0.310038	0.316106
"D50"	100.0000	0.345662	0.358506
"D55"	100.0000	0.33242	0.347438
"D75"	100.0000	0.299023	0.314871
"FL1"	100.0000	0.185922	0.273141
"FL2"	100.0000	0.184704	0.337311
"FL3"	100.0000	0.187339	0.379794
"FL4"	100.0000	0.19409	0.421538
"FL5"	100.0000	0.180196	0.264471
"FL6"	100.0000	0.176995	0.331871
"FL7"	100.0000	0.251018	0.301727
"FL8"	100.0000	0.287456	0.333245
"FL9"	100.0000	0.284426	0.357334
"FL10"	100.0000	0.184757	0.305641
"FL11"	100.0000	0.190064	0.340407
"FL12"	100.0000	0.197444	0.391261
"HP1"	100.0000	0.177508	0.584345
"HP2"	100.0000	0.376282	0.432582
"HP3"	100.0000	0.254452	0.379496
"HP4"	100.0000	0.252627	0.339056
"HP5"	100.0000	0.282823	0.353062
"E"	100.0000	0.333333	0.333333

Table 8 White point references.

5.6 Measurement commands

Command syntax	Parameters	Range	Purpose
:MEASure:XYZ	None		Measure XYZ
:MEASure:Yxy	None		Measure Y and x,y colour point
:MEASure:Yuv	None		Measure Y and u',v' colour point
:MEASure:Y	None		Measure Y (luminance)
:MEASure:DWL	None		Measure dominant wavelength, purity, Y and x,y colour point
:SAMPle:XYZ	Samples	0 – 4000	Sample XYZ
:SAMPle:Y	Samples Delay	0 – 10000 0 – 255	Sample Y
:MEASure: TEMPerature	None		Measure temperature of sensor head and CPU
:MEASure: FLICker	Method, sample count, sample delay	Method 0 – 5 Sample count 1 – 10000 Delay 0 – 255	Measure flicker Method: 0: Contrast max/min 1: Contrast RMS 2: JEITA 3: VESA 4: Flicker Percentage 5: Flicker Index

Table 9 Measurement commands.

Note: The delay time is set in sample times, meaning a delay of one will skip one sample.

Note: When using high sample amount or long integration times, make sure timeout values in the application software are set accordingly.

Note: The :SAMPLE functions do not use auto-ranging. They use the fixed integration time. Since :SAMPLE functions are often used to measure switching signals, auto ranging is not useful.

Table 9 shows the measurement commands of the Hyperion colorimeter. Regarding colour/luminance measurement, there are two kind of commands (MEASure/SAMPle). The MEASure commands measure the requested values using the set averaging and integration time and returns the result in a single structure of three single precision floating point values. The SAMPle commands measure the requested parameters using a sample count and delay time and return an array of data. The array contains single floating point data. Each sample count equals one complete structure, for example one XYZ structure of data. The delay time is set in sample times, meaning a delay of 1 equals skipping 1 sample.

5.7 User EEPROM commands

Table 10 shows the commands which can be used to store values in the user EEPROM space. It is advised to reboot the Hyperion after writing new values to the EEPROM.

Command syntax	Parameters	Range	Purpose
:EEPROM: STARTUP:READ	None		Copies startup conditions from EEPROM to internal variables. Values can then be read.
:EEPROM: STARTUP:WRITE	None		Copies internal variables to EEPROM
:EEPROM:CONFigure:MODE	Mode	0 = USB 1 = RS232	Configures the mode. This is currently not used.
:EEPROM:CONFigure:MODE?	None		Reads the mode as set in the EEPROM
:EEPROM: CONFigure:BAUDRATE	Baudrate	0 = 9600 1 = 19200 2 = 38400 3 = 57600 4 = 115200 5 = 230400	Configures the RS232 baudrate in the EEPROM
:EEPROM: CONFigure:BAUDRATE?	None		Read the current RS232 baudrate from EEPROM
:EEPROM: CONFigure:TRIG	Trigger	0 = off 1 = on	Configures triggering
:EEPROM: CONFigure:TRIG?	None		Reads from EEPROM value for the trigger variable.
:EEPROM: CONFigure:TRIGDELAY	Delay	0 – 30000000	Configures a delay before triggering the Hyperion in microseconds
:EEPROM: CONFigure:TRIGDELAY?	None		Reads the configured delay before triggering the Hyperion.
:EEPROM:CONFigure:AUTO:FREQ	Frequency (Hz)	1 – 255	Auto-range parameter: frame frequency of the source (display) to measured
:EEPROM:CONFigure:AUTO:FREQ?	None		Reads the set frequency
:EEPROM:CONFigure:AUTO:FRAMES	Frames	1 – 255	Auto-range parameter: The number of frames to measure
:EEPROM:CONFigure:AUTO:FRAMES?	None		Reads the set Frames
:EEPROM:CONFigure:AUTO:ADJMIN	Adjmin (%)	1 – 100	Auto-range parameter: The minimum level to adjust to
:EEPROM:CONFigure:AUTO:ADJMIN?	None		Reads the set Adjmin
:EEPROM:CONFigure:MAXINT	Max int	1000 – 1000000	Maximum integration time. If time is too short, measuring dark level may be difficult
:EEPROM:CONFigure:MAXINT?	None		Query the maximum integration time
:EEPROM:CONFigure:INT	Int time	500 – 1000000	Configures default Hyperion integration time
:EEPROM:CONFigure:INT?	None		Reads the default integration time
:EEPROM:CONFigure:AVG	Averaging	1 – 200	Configures default Hyperion averaging
:EEPROM:CONFigure:AVG?	None		Reads the default averaging
:EEPROM:CONFigure:AUTORANGE	Auto-range	0 = off 1 = on	Configures auto-ranging

:EEPROM:CONFigure:AUTORANGE?	None		Reads the auto-range setting
:EEPROM:CONFigure:WHITE	String[4]	“D65”, “A” etc (see table: 8)	Used for Dominant wavelength
:EEPROM:CONFigure:WHITE?	None		Gets the currently configured white point
:EEPROM:CONFigure:GAIN	Gain	1 – 3	Configures default Hyperion Gain
:EEPROM:CONFigure:GAIN?	None		Reads the default gain
:EEPROM:SBW:READ	User matrix Index 1 Index 2	0 – 29 0 – 2 0 – 2	Read user calibration matrix
:EEPROM:SBW:WRITE	User matrix Index 1 Index 2 Value	0 – 29 0 – 2 0 – 2 Float	Write user calibration matrix
:EEPROM:READ:SN	None		Read serial number
:EEPROM:READ:SBWNAME	Matrix number	0 – 29	Read name of matrix
:EEPROM:READ:SBWNAMEFACT	Matrix number	0	Read name of factory matrix
:EEPROM:READ:USERCAL	None		Reads the user calibration values from EEPROM to memory
:EEPROM:WRITE:USERCAL	None		Write the user calibration values form memory to EEPROM. This step fixes all values for a next restart of the instrument
:EEPROM:READ:SN	None		Read the device serial number
:EEPROM:READ:UNIT			Reads the output unit (luminance, illuminance)
:EEPROM:READ:ID			Read device ID
:EEPROM:WRITE			Write all settings to EEPROM

Table 10 User EEPROM commands.

5.8 Returned results

:MEASure command return their result in ASCII formatted floating point as shown below where X,Y,Z can be substituted for Y,u,v or other colour spaces.

(X,Y,Z,clip,noise) → %f,%f,%f,%d,%d\n

:SAMPle command return all measurement data also in ASCII format, except the :Sample:Y function. The first three values indicate the delta time between samples and the clip and noise values.

MEAS command	
dt	%f\n
Clip	%f\n
Noise	%f\n
Value1	%f\n
Value2	%f\n
Value3	%f\n

Table 11 Return result MEAS command.

Exceptions to the above are the :MEASure:TEMPerature, :MEAS:Y and :SAMPle:Y commands.

:MEASure:TEMPerature → (Sensor temperature) → %f\n

:SAMPle:Y returns its data in Single precision floating point format.

Sample Y command	
dt	%f\n
Clip	%f\n
Noise	%f\n
Value1	%f\n
Value2	%f\n
Value3	%f\n

Table 12 Return result SAMPLE Y command.

Note: In RS232 mode the SAMPLE command separate the values using a TAB (\t) and the last value is terminated using an end of line constant (\n).

6 Measurement example

The Hyperion uses default settings when the device is started. These can be programmed by the end user so that the device starts with the same settings each time it is connected.

Although it's possible to program all Hyperion devices in production environment to start with equal settings, it is recommended to set the averaging, integration time and SBW values in the initialization routine of the host software.

A typical measurement example of XYZ would include the following commands as shown on the right.

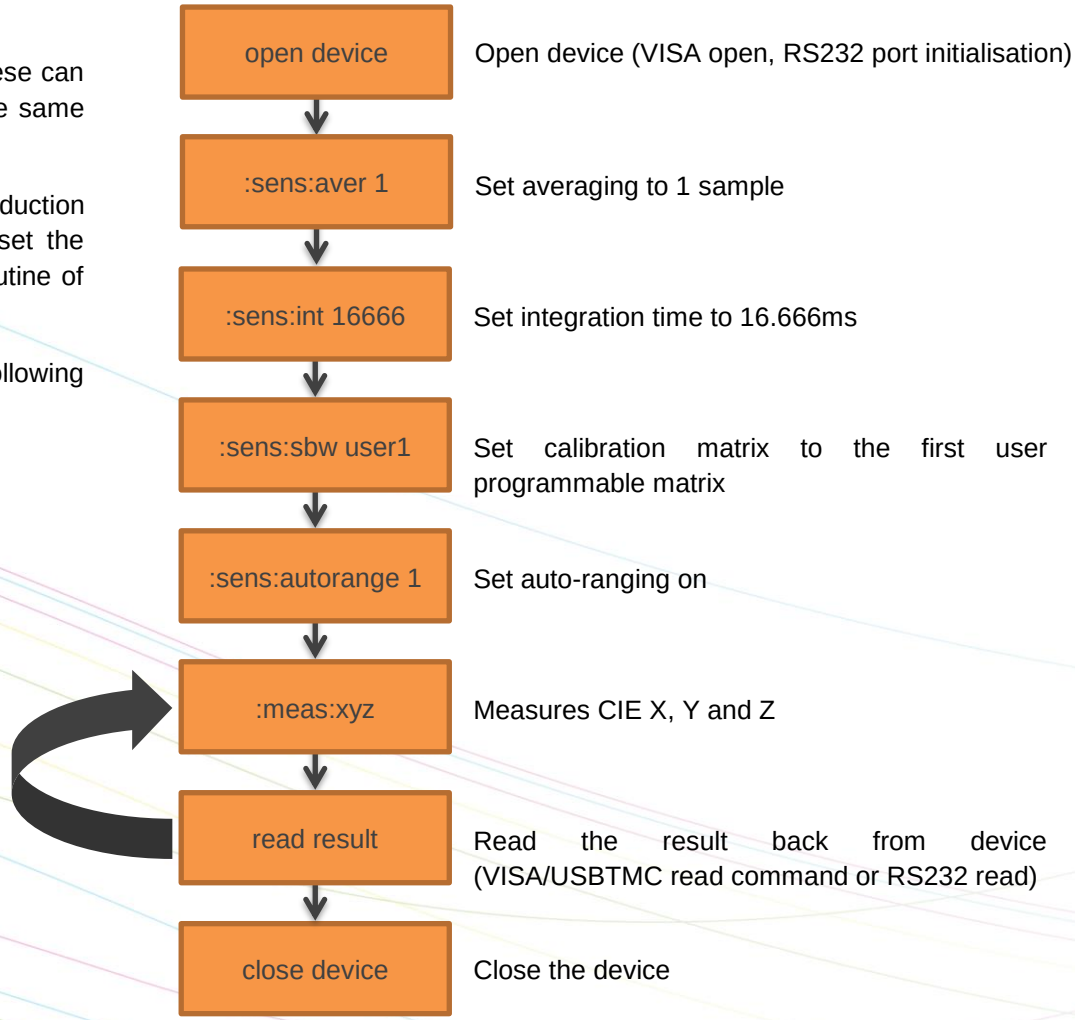


Fig 4 Measurement example.

7 Auto-ranging

7.1 Introduction

The Hyperion includes an auto range function. This function is useful in case the measured object shows an unknown luminance value. In this case, the Hyperion will try to find the optimum setting which is a trade-off between speed and the stability of the instrument. The auto-ranging function can also be fine-tuned to reach better stability levels by setting a few parameters. Auto-ranging can be controlled by 3 parameters.

- Frequency : supposed to be frame frequency of the source (display) that is measured
- Frames : The minimum number of frames to measure
- Adjmin : The minimum level to adjust to

Adjmin can vary between 1 and 100. Reasonable results will be achieved when set to 10 regarding the speed and the stability. If high stability is needed, this setting must be increased (but measurements will become slower). If a faster measurement is needed and the stability level may be lowered a little, the value of adjmin can be decreased. When the Hyperion measures in auto range mode, it can happen that the found integration time is very low. For example, when measuring white it may be just 5ms. When this happens, the Hyperion will automatically increase the averaging so that the total measurement time equals >= (1/frequency) · frames.

7.2 How auto-ranging works

The auto-ranging works, by first setting a default integration time. If this already meets the criteria for a good measurement, the measurement will be done using that integration time. It should be clear that this is the fastest because no adjustment will be done. In case the measured result is either to low or too high (clip), than the Hyperion will adjust to a better level. The following graph shows how levels inside the Hyperion work.

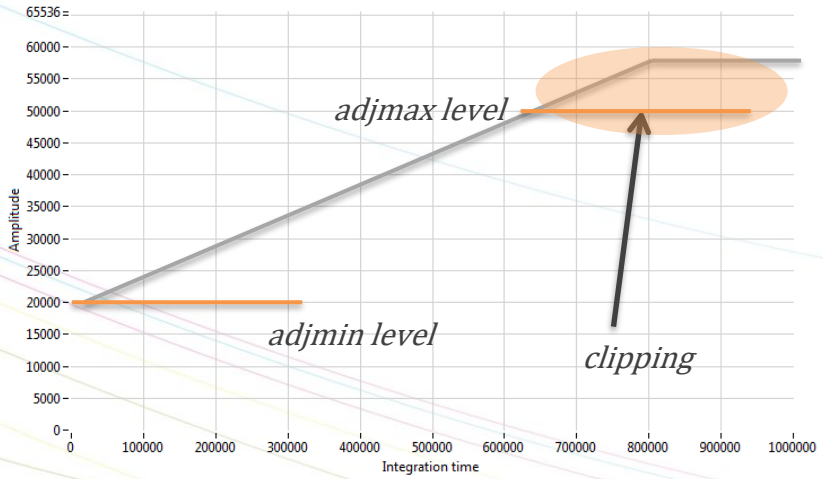


Fig 5 Auto-range levels

When auto-ranging is set, the Hyperion will accept any level between “Adjmin” and “Adjmax” as a good signal. The Adjmin level can be set by the user through a software command, allowing some fine tuning of the stability and the speed of the instrument.

7.3 Auto-range parameters

Auto-range controls the following parameters:

- **Frequency:** This setting must be set to the frame frequency of the measured sample. The range is 1 to 255 Hz. If a sample has a higher frequency than that, set it such way that 1/frequency covers multiple frames of the sample.

- **Frames:** The number of frames that at least should be measured. It means auto ranging will have a minimum measurement time of (1/frequency) · frames. Example: In case frequency is 60Hz and frames is 3, the Hyperion will measure at least 3 frames at an integration time of 16.66ms. This would be equal to measuring in non-auto-range with a fixed integration time of 16.66ms and averaging of 3.

- **Adjmin:** The adjmin parameter is very important. The higher this level is set, the more stable the measurement becomes, but it also slows down the measurement if set too high. That means, there is a trade-off between speed and stability. It mainly affects the lower grey scale levels. If these levels appear not stable, adjmin needs to be increased. Adjmin is set from 1-100, which means 1-100% of the Adjmax level (Adjmax is fixed by Admesy).

- **Maximum integration time:** Although not really part of the auto-ranging algorithm, this parameter is used when the auto-ranging result exceeds this setting of integration time. When the grey scales are low (grey scale 0 for example), the maximum integration time setting can be used to allow only a maximum measurement time. Of course this affects stability (longer measurement is always more stable), but again this is a trade-off for production environments to save on test time.

7.4 Auto-range in practice

When auto-ranging is set, the measurement settings (integration time, gain and averaging) are automatically adjusted. An initial measurement is done to determine the final settings of averaging and integration time, based on the settings of frequency, frames and adjmin.

The set integration time (1/frequency as set in the EEprom auto-range frequency parameter) of the instrument will always be used as the first value to try. If this setting results in a value between adjmin and adjmax, no further actions are necessary and the speed is optimal.

When measuring a display we can measure random patterns, but mostly known patterns are measured. Setting an integration time that is nearly right, 1/frequency (Hz) is preferred, as it optimizes the speed for adjusting the auto ranging settings and thus provides optimal results.

Example: a 23” TFT display (standard desktop LCD) with white at 290 cd/m² and a frame rate of 60Hz. Settings for auto ranging are:

- Frequency = 60
- Frames = 1
- Adjmin = 10
- Max int = 200000

These settings will provide a stable measurement on white and black. Auto-range algorithm will adjust to measure black 0.3cm/m² with 200ms integration time, resulting in ~225ms tact time.

7.5 Programming Hyperion for auto-ranging

Hyperion has start-up settings. This means that when the Hyperion is connected to a PC or pattern generator, it only needs the ":meas:Yxy" command to get luminance and colour data. All other settings can be saved as preset and are loaded when the instrument is started. However, it requires the right settings to be set in advance. This can be done though the Admesy Iliad application and select Device→Start-up settings→Hyperion. These setting may be optimized per display type. There are a few categories we can define. Those are shown in the next chapters.

- **Display without PWM:** This can be measured using almost any settings, but the settings may be optimized for high speed. This means, we can use a short integration time for bright images (white for example) and decreased the maximum integration of black to a level that gives stable results and is still fast. In this case the frequency setting is not critical.

- **Display with PWM:** Since the luminance will show PWM, a measurement must at least cover 1 full frame. Taking more frames will result in better stability. In this case the frequency setting is apparently more critical to get a good stability.

7.6 Auto-ranging recommendations

It is recommended to apply auto ranging in case process variation is high or in case various grey scales are measured. In cases where luminance setting is known, it may be desirable to turn off auto-ranging to save additional measurement / test time.

Regarding the grey scales, like a gamma measurement is performed, speeds up the measurement significantly, when using auto-ranging in combination with the feedback function.

There are a lot of ways to fine tune the Hyperion for each application. Take this document into consideration and apply the comments correctly. By assuming that any default setting will do, it will lead to a non-optimal measurement results. Any measurement is application dependent. One should consider that auto-range settings exist for the sole purpose of optimizing each application.

8 Hyperion formulas

8.1 Formulas

The Hyperion colorimeter uses an XYZ sensor, meaning that other colour spaces are being converted from XYZ. The following sections show the mathematical conversions that are used by the Hyperion colorimeter to perform conversion from XYZ to other colour spaces.

Since the Hyperion internal processor is usually slower than the host system it is recommended to do colour space conversion on the host system for the fastest possible measurement.

8.2 XYZ to Yxy conversion

$$x = \frac{X}{X + Y + Z}$$

$$y = \frac{Y}{X + Y + Z}$$

8.3 XYZ to CIE 1976 UCS Yu'v'

u'v' is noted without the hyphen in the Hyperion commands. All Yuv commands perform CIE Yu'v' calculations.

$$u' = \frac{4X}{X + 15Y + 3Z}$$

$$v' = \frac{9Y}{X + 15Y + 3Z}$$

8.4 Flicker calculation

The Hyperion flicker measurement calculation is based on the ratio between the signal's AC and DC component. How these levels are calculated depends on the chosen standard. When signal filtering is needed, it is recommended to acquire the raw data from the Hyperion and perform the calculation externally as for example in the Admesy Iliad application. The Hyperion has six types of flicker measurement, each with different calculation method:

Flicker Min Max :MEASure:FLICker 0,samples,delay

This method is based on the minimum and maximum value of the AC component

$$Flicker = \frac{(\max - \min)}{((\max + \min) / 2)} \times 100\%$$

Flicker RMS :MEASure:FLICker 1,samples,delay

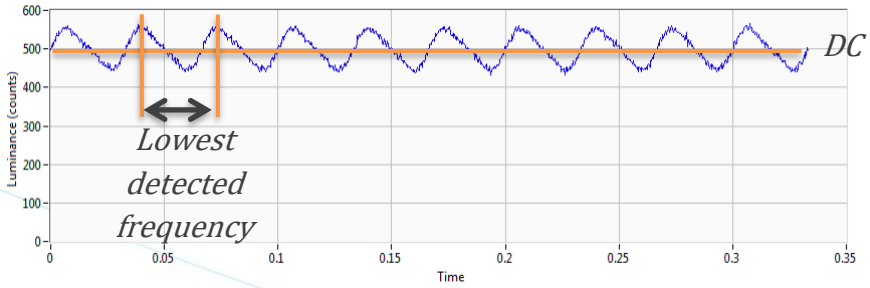
This method is based on the RMS of the AC component

$$Flicker = \frac{\sqrt{\frac{1}{n}\sum_{i=0}^{n-1}(x - xdc)^2}}{xdc} \times 100\ [\%]$$

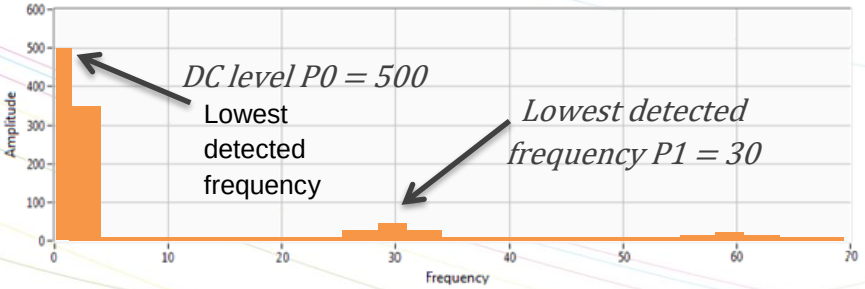
$$xdc: average\ value\ of\ the\ signal$$

Flicker Jeita :MEASure:FLICker 2,samples,delay

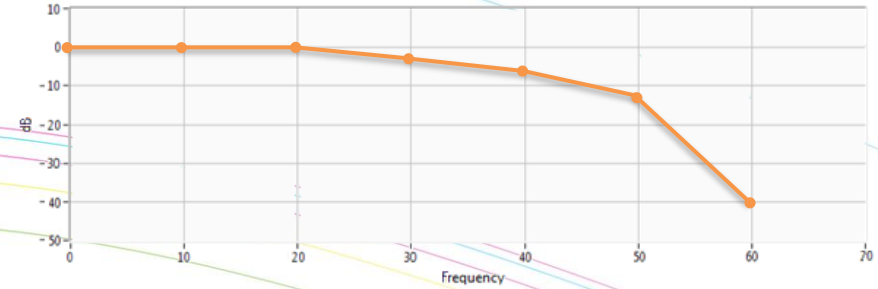
The JEITA method is based on frequency domain calculations. It uses an FFT to determine the AC and DC level of the measured signal.



It translates the signal into an FFT:



After calculating the FFT, a weighting factor is applied to compensate for human eye sensitivity to frequency. This has been defined as shown in the graph and table below.



Frequency	Factor	
(Hz)	dB	Ratio
0	0	1.000
10	0	1.000
20	0	1.000
30	- 3	0.708
40	- 6	0.501
50	- 12	0.251
60	- 40	0.010

This means that in the signal we measured, P1 at 30Hz should be reduced by -3dB. The 60Hz component should be reduced by -40dB but since it is lower than the 30Hz component, it is not used in the final calculation anyway.

The amplitudes found in the previous page (P0 and P1) are reduced by the factors and the final result will then be:

Pr0 = P0 * 1

Pr1 = P1 * 0.708

$$Flicker_{JEITA} = 20\ Log_{10}\left(\frac{Pr1}{Pr0}\right) dB$$

Advantages:

- No need for an additional low pass filter

Disadvantages:

- Slowest method, on embedded systems sometimes too slow to be useful. Needs a fast CPU for FFT calculation.
- In case of too low number of samples, not always the most stable method.
  So, often in production areas not suitable because of speed.

Flicker Vesa :MEASure:FLICker 3,samples,delay

The VESA method is equal to the JEITA method with a small difference in the result calculation which results in a difference of approximately 3.01dB caused by squaring the amplitudes in the FFT:

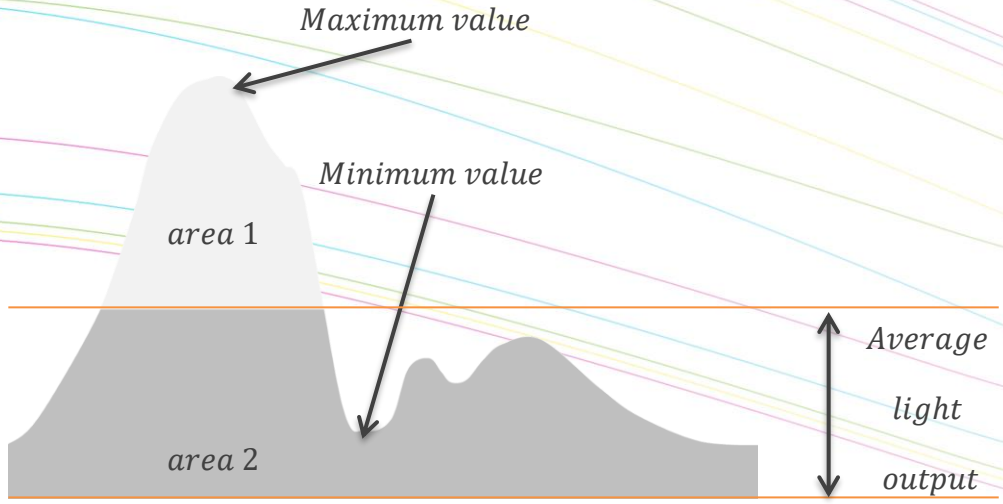
$$Flicker_{VESA} = (Flicker_{JEITA} + 20\ Log_{10}(\sqrt{2}))dB$$

Percent Flicker :MEASure:FLICker 4,samples,delay

$$Percent\ Flicker = 100\% \times \left(\frac{A - B}{A + B}\right)$$

Flicker Index :MEASure:FLICker 5,samples,delay

$$Flicker\ Index = \left(\frac{Area\ 1}{Area\ 1 + Area\ 2}\right)$$



8.5 Additional colour spaces

Admesy provides a colour library for its customers. This library is supplied in the form of a shared library and contains all of the above calculations and a lot of additional colour space conversions and handy routines for colour calculation.

9 Operating mode

The operating mode only matters in trigger mode. The mode is used to send back results by USB or RS232. The modes of the Hyperion are.

- USB mode
- RS232 mode

A trigger actives only one command, for example ":meas:xyz". This command needs to be set first by the host. After a trigger is received, the command will execute and the result will be presented on the selected interface as shown above. The Hyperion responds to a rising edge of the trigger signal.

10 Typical spectral sensitivity

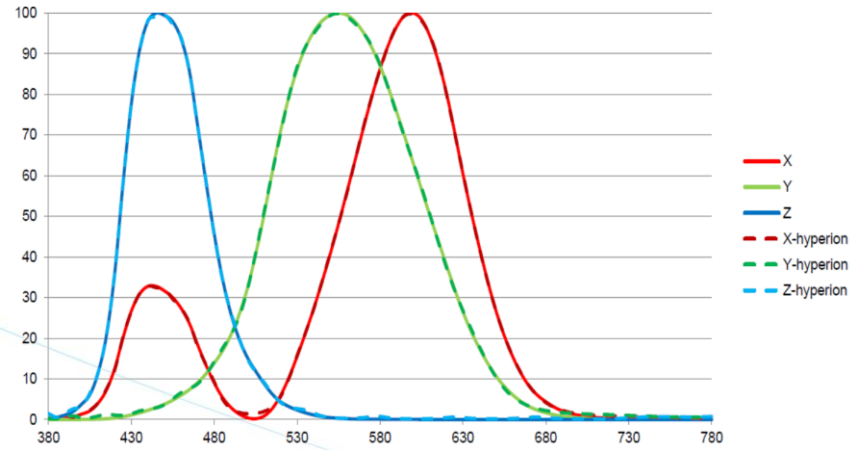


Fig 8 Spectral sensitivity of the colorimeter.

Admesy B.V.
Sleestraat 3
6014 CA Ittervoort
The Netherlands

T +31 (0)475 600 232
F +31 (0)475 600 316

www.admesy.com
info@admesy.com

Version 1.1.0 05/2019