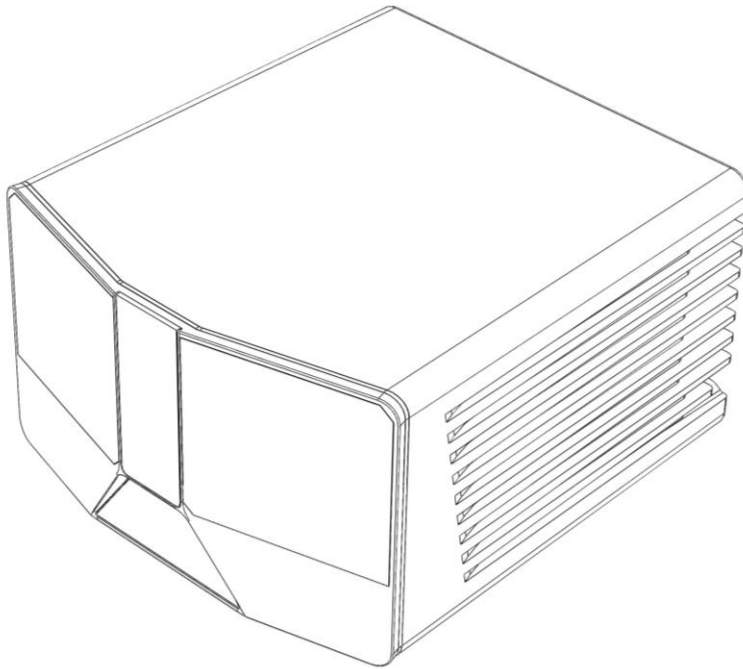




L3CAM USER MANUAL



PRODUCT MODEL	L3CAM
RELEASE	1.8

Version:	1.8	 BEAMAGINE L3CAM User manual	Page:	2/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

Disclaimer of Liability

The information disclosed to you hereunder (the “Materials”) is provided solely for the selection and use of Beamagine products. To the maximum extent permitted by applicable law: (1) Materials are made available “AS IS” and with all faults, Beamagine hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Beamagine shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Beamagine had been advised of the possibility of the same. Beamagine assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. The products are subject to the terms and conditions of Beamagine’s limited warranty.

Beamagine products are not designed to provide safety functions nor to be integrated as a part of a safety system. You assume sole risk and liability for use of Beamagine products in such critical applications.

Limited Warranty

Beamagine warrants the L3CAM sensor to be free of defects in materials and workmanship for 12 months from the date of shipment. The user should not attempt to service the L3CAM sensor product beyond the description in this manual. Any attempt by the user to repair any Beamagine product under warranty will void the warranty. If the sensor fails during the warranty period, return the laser freight prepaid to Beamagine with a failure description made by the user. Beamagine will, at its option, repair or replace the defective item and return it freight prepaid to the user. Beamagine also offers repair services for products beyond the warranty. The damaged product is then investigated and evaluated at Beamagine resulting in a repair service report and quotation for the customer to consider. When you return products to Beamagine, please ask for the RMA (Return Material Authorization) number. Beamagine does not accept any returns without an RMA number, and/or if the products are mechanically damaged, scratched, burned or insufficiently and inappropriately packaged and especially if opened by unauthorized persons.



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 L3CAM User manual	Page:	3/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

REVISION HISTORY

Date	Version	Revision
13/04/2022	1.0	Initial Release
13/06/2022	1.1	Protocol for UDP modified, the timestamp is now uint32_t. Added label for detections structure. Added UDP/RTSP port description for RGB and LWIR sensors.
18/07/2022	1.2	Protocol for RTSP updated to H264. Library functions list updated for RGB and thermal images. Thermal and RGB images added in RTSP samples. Structure definitions and error definitions updated. Fixed error with point cloud colormap images.
13/09/2022	1.3	Fixed typos in library API examples for RGB camera functions. Updated network configuration function to enable/disable DHCP
22/11/2022	1.4	Updated API examples and added TERMINATE function description.
06/03/2023	1.5	Updated API examples and added Allied Cameras functionalities. Updated streaming protocols and added sender and receiver pipelines for the allied cameras. Updated error code tables. Updated structure definitions.
13/11/2023	1.6	Updated API function lists. Windows and Linux network configuration rewritten. Updated API structure definitions. Added RAW and H264 pipelines for Polarimetric camera streaming. Added device Field of View description
05/02/2024	1.7	Added power supply configuration image and instructions.
13/03/2024	1.8	Added temperature array streaming information Added new thermal library functions Added new thermal pipelines definitions Added new thermal colormap definitions Modified sample code for reading Images and point clouds Added sample code for reading thermal data Added sample images for thermal image processing pipelines



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	4/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

TABLE OF CONTENTS


ACRONYMS.....	9
DEFINITIONS.....	10
REFERENCE DOCUMENTS	11
1 L3CAM DESCRIPTION	12
1.1 OVERVIEW	12
1.2 KEY FEATURES	13
1.2.1 Solid-state LIDAR	13
1.2.2 Imaging modes	15
1.2.3 Data fusion and embedded processing (optional features).....	17
2 GETTING STARTED	20
2.1 SAFETY WARNINGS AND CAUTIONS	20
2.2 LIST OF MATERIALS.....	20
2.3 SYSTEM SETUP	21
2.3.1 System connection	21
2.3.2 Windows host configuration.....	22
2.3.3 Linux host configuration.....	28
3 API USER GUIDE	31
3.1 LIBRARY OVERVIEW	31
3.2 UDP PROTOCOL AND DATA DESCRIPTION	32
3.2.1 Point cloud	32
3.2.2 Image.....	35
3.2.3 Thermal data.....	36
3.3 RTSP PROTOCOL AND DATA DESCRIPTION.....	38
3.3.1 Point cloud	39
3.3.2 Image Polarimetric.....	40
3.3.3 Image RGB.....	41
3.3.4 Thermal image.....	42
3.3.5 Image RGB Allied Wide.....	43
3.3.6 Image RGB Allied Narrow	44
3.4 API FUNCTIONS LIST	45
3.5 API FUNCTIONS DESCRIPTION	48
APPENDICES.....	145



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	5/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

APPENDIX 1	READING A POINT CLOUD	145
APPENDIX 2	READING AN IMAGE	147
APPENDIX 3	READING TEMPERATURES DATA	149
APPENDIX 4	POLARIMETRIC PIXEL FORMAT	150
APPENDIX 5	THERMAL IMAGE PROCESSING PIPELINES	151
APPENDIX 6	PRODUCT DESCRIPTION	152
APPENDIX 7	RETURN CODES	153
APPENDIX 8	STATUS.....	157
APPENDIX 9	DATA STRUCTURES	158
APPENDIX 10	L3CAM DIMENSIONS	164



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	6/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

LIST OF TABLES

Table 1. Acronyms.....	9
Table 2. Definitions.....	10
Table 3. Reference documents	11
Table 4. libL3Cam ports.	32
Table 5. Point cloud frame header.	33
Table 6. Data packet structure.	33
Table 7. Point structure.	34
Table 8. Point cloud frame footer.	34
Table 9. Image frame header.	35
Table 10. Perception packet.....	36
Table 11. Point cloud frame footer.	36
Table 12 Temperature header packet.....	37
Table 13. GStreamer pipeline.	38
Table 14. API functions libL3Cam.h.....	45
Table 15. API functions libL3Cam_allied.h.....	46
Table 16. API functions libL3Cam_econ.h	47
Table 17. API functions libL3Cam_polarimetric.h	47
Table 18. API functions libL3Cam_thermal.h.....	47
Table 19. Device temperatures pointer content	70
Table 20. Compatible sensors list	152
Table 21. Library return codes.	153
Table 22. LiDAR errors.....	154
Table 23. Econ RGB camera errors	154
Table 24. Polarimetric camera errors.	154
Table 25. Thermal camera errors.....	154
Table 26. Allied Cameras errors.....	155
Table 27. Status bits.....	157
Table 28. sensorTypes enum fields.	158
Table 29. pointCloudColor enum fields.	159
Table 30. streamingProtocols enum fields.	160
Table 31. sensor struct fields.....	160
Table 32. l3cam struct fields.....	161
Table 33. econResolutions enum fields.	161



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	7/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

LIST OF FIGURES

Figure 1. Components and functions of L3CAM.	12
Figure 2. General scheme of the pulsed solid-state MEMS-based L3CAM LIDAR.	13
Figure 3. MEMS raster scan and its associated FOV.	13
Figure 4. Scheme of the internal L3CAM arrangement and covered FOV.	14
Figure 5. Multiple back-reflected hits example whilst scanning a bulky object.	15
Figure 6. Bayer filter mounted on an imaging sensor.	16
Figure 7. Polarimetric imaging examples.	16
Figure 8. Scheme of the microscopic filter structure of the Sony polarsens sensors.	17
Figure 9. Examples of outdoor LWIR thermal images.	17
Figure 10. Multimodal fusion between the L3CAM LIDAR and different imaging modes.	18
Figure 11. Enhanced AI pedestrian perception of an outdoor scene thanks to data fusion.	19
Figure 12. System connection.	21
Figure 13. L3CAM side view.	21
Figure 14 Power Supply Connection.	22
Figure 15. Windows control panel.	22
Figure 16. Windows Network and Internet menu.	23
Figure 17. Windows Network and Sharing Centre.	23
Figure 18. Windows Network interface configuration.	24
Figure 19. Windows, (a) Network interface configuration, (b) IP address configuration.	25
Figure 20. Windows, Access advanced network interface properties.	26
Figure 21. Modify jumbo packet size in Windows.	26
Figure 22. Change receive buffers settings in Windows.	27
Figure 23. Ubuntu 20 Network settings menu.	28
Figure 24. Ubuntu 20 Network interface configuration.	28
Figure 25. Ubuntu 20 IP address configuration.	29
Figure 26. Ubuntu 20 MTU configuration.	29
Figure 27. Ubuntu 20 Network restart (a) power off (b) power on.	30
Figure 28. Configuration of the receiving buffers size.	30
Figure 29. Categories of the functions of the libL3Cam API.	31
Figure 30. L3CAM initialization sequence.	31
Figure 31. Operation sequence.	31
Figure 32. Transmission sequence of point cloud data.	33
Figure 33. Point cloud reference coordinate system.	34
Figure 34. Colour scale for distance (a) and intensity (b).	34
Figure 35. Transmission sequence of image data with results from perception functions.	35
Figure 36 Transmission sequence of temperatures.	37
Figure 37. Point cloud and cropping area.	39
Figure 38. Polarimetric Bayer image. See image format in APPENDIX 4.	40
Figure 39. RGB image from econ camera.	41
Figure 40. Thermal image using TYRIAN colormap.	42
Figure 41. Allied 1800 U-508c with KOWA-LM6JC.	43
Figure 42. Allied 1800 U-319c with M2514-MP2.	44
Figure 43. Colour filter array of Lucid PHX050S1.	150
Figure 44 Lite Thermal Image.	151
Figure 45 Legacy Thermal Image.	151



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	8/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

Figure 46 SeekVision Thermal Image	151
Figure 47. L3CAM front side dimensions.	164
Figure 48. L3CAM mount holes dimensions.	165



Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	9/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

ACRONYMS

Table 1. Acronyms

ACRONYM	DESCRIPTION
AC/DC	A lternating C urrent / D irect C urrent
AI	A rtificial I ntelligence
API	A pplication P rogramming I nterface
CCD	C harge- C oupled D evice
CMOS	C omplementary M etal- O xide S emiconductor
DMD	D igital M icromirror D evice
DOLP	D egree O f L inear P olarization
FOV	F ield O f V iew
IP	I nternet P rotocol
IPV4	I nternet P rotocol V ersion 4
IR	I nfra R ed
LIDAR	L ight D etection A nd R anging
LWIR	L ong- W ave I nfra R ed
MEMS	M icro E lectro M echanical S ystem / S ensor
MTU	M aximum T ransmission U nit
NIC	N etwork I nterface C ard
NIR	N ear I nfra R ed
RGB	R ed- G reen- B lue
RTSP	R ea L T ime S treaming P rotocol
SNR	S ignal-to- N oise R atio
SW	S oft W are
TCP	T ransmission C ontrol P rotocol
TOF	T ime O f F light
TOT	T ime O ver T hreshold
UDP	U ser D atagram P rotocol
VIS	V ISible



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	10/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

DEFINITIONS

Table 2. Definitions

TERM	DESCRIPTION
Data fusion	Data fusion is the process of integrating multiple data sources to produce more consistent, accurate, and useful information than that provided by any individual data source.
DOLP	Measure of the amount of linearly polarized light. The larger the DOLP, the clearer the linear polarization state.
Hit	A hit is a point at which the laser beam pulse reaches an object that reflects it to the LIDAR.
Perception	Machine perception is the capability of a system to interpret data similarly to humans.
TOF	Elapsed time between light events. In the case of L3CAM, TOF is the elapsed time a pulse travels from the laser source, backscatters on a target and comes back to the detector. Thus, it is a direct measure of the distance to the target.
TOT	The elapsed time that the voltage signal caused by an incoming pulse on the photodetector is above a threshold level. Thus, it is a measure of the total energy received since the larger the TOT, the higher the intensity.

Version:	1.8		Page:	11/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024


L3CAM User manual

REFERENCE DOCUMENTS

Table 3. Reference documents

	DOCUMENT	TITLE	VERSION	DATE
RD 1	L3CAM datasheet	L3CAM Multimode imaging LIDAR datasheet	1.0	06/04/2022
RD 2	GeniCam standards	European machine vision association web page.	NA	



Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	12/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

1 L3CAM DESCRIPTION

1.1 OVERVIEW

The L3CAM is a multimodal sensor composed essentially of a solid-state LIDAR sensor and up to 3 additional imaging modes (RGB, thermal and/or polarimetric).

The LIDAR system is a patented MEMS-based scanning technology that combines high-resolution 3D imaging, real-time frame rate, and long-range. The most suitable combination for applications related to autonomous vehicles, security, object detection, situational awareness and mapping.

Critical applications, however, require more than a single “eye” to achieve high-reliability levels once the data is processed. This is where the additional imaging modes play a significant role.

The data from the different sensors can be integrated through data fusion algorithms to provide more consistent, accurate and useful information about the environment.

On top of the processing chain, at the higher abstraction level, L3CAM integrates state-of-the-art perception algorithms to perform objects recognition.

L3CAM offers the all in a single, compact and cost-effective device.

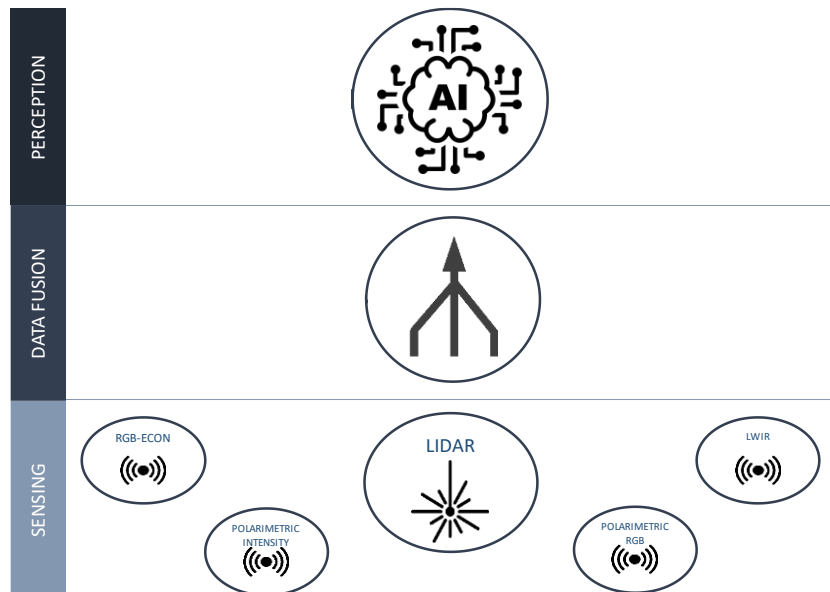


Figure 1. Components and functions of L3CAM.

Version:	1.8	 L3CAM User manual	Page:	13/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

1.2 KEY FEATURES

1.2.1 Solid-state LIDAR

The L3CAM LIDAR sensor is a solid-state device without mechanical moving parts for scanning the environment or large embodiments. Its optomechanical design is based on a MEMS scanner which steers a pulsed IR fibre laser beam towards targets within the FOV of the system and a receptor that gathers the backscattered light to resolve the distance to them from its TOF – time the pulse is travelling from the LIDAR source to the object and backscatters to the detector. This subdivision of the sensor can be appreciated in the following figure.

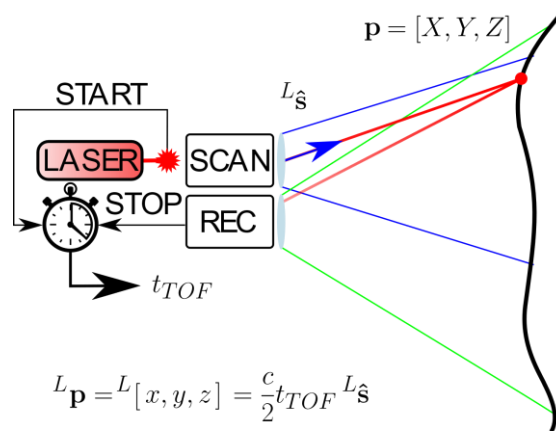


Figure 2. General scheme of the pulsed solid-state MEMS-based L3CAM LIDAR.

Using a MEMS scanner enables reaching high output frame rates (~10 Hz) as well as high-resolution Point Clouds with a large point density. The L3CAM LIDAR offers a rectangular FOV, similar to conventional cameras, by using a raster scan that provides great precision, accuracy and angular resolution in both horizontal and vertical scanning directions (<0.1°). The FOV is cropped to avoid non-linearities in the changes of lines.

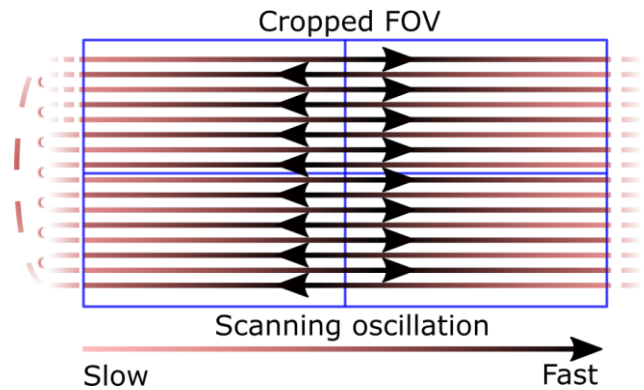


Figure 3. MEMS raster scan and its associated FOV.

Together with the solid-state design, the L3CAM LIDAR sensor offers a patented solar background suppression that enhances its detectivity, thus enlarging its measurement range, by improving the SNR.



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	14/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

It does not only become L3CAM LIDAR robust to sunlight but also self-immune to other devices situated simultaneously together in the same environment. Hence, it is possible to use an array of L3CAM devices for covering a wider FOV without the risk of cross-talking in the union.

In fact, the L3CAM LIDAR consists of two LIDAR modules, understanding a module as a pair of scanner-receiver units. They are horizontally separated and tilted with a given angle to cover a wider FOV. Whilst each unit cover a $30^{\circ} \times 20^{\circ}$ (HxV) FOV, the L3CAM LIDAR reaches a total $60^{\circ} \times 20^{\circ}$ FOV thanks to this arrangement of the modules, as the image below graphically explains:

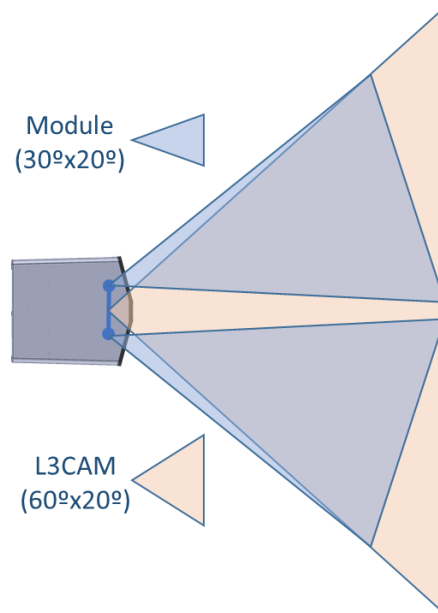


Figure 4. Scheme of the internal L3CAM arrangement and covered FOV.

It is important to notice that the modules' FOV have a minimum adjustable overlapping area in the central part of the L3CAM FOV and that the design FOV of $60^{\circ} \times 20^{\circ}$ is guaranteed by calibration at long ranges.

Moreover, the L3CAM LIDAR sensor is capable of gathering up to 4 different back-reflected pulses in a single scanning direction because the laser beam slightly broadens as long as it propagates through space due to its divergence ($\sim 0.1^{\circ}$). Consequently, a portion of the total emitted energy might back-reflect from either a tiny object or at the contour of a bulky one whereas the remaining one might keep travelling towards further targets producing multiple reflections with different back-reflected intensities measured with the TOT of the measured peak, as Figure 5 demonstrates below.

Version:	1.8		Page:	15/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

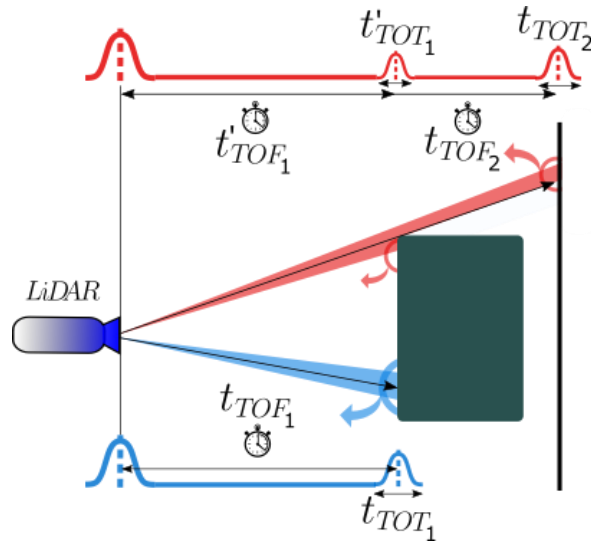


Figure 5. Multiple back-reflected hits example whilst scanning a bulky object.

To summarize, the L3CAM LIDAR sensor offers a high-end imaging performance thanks to the high resolution provided by the MEMS scanner and the enhanced range detectivity by the patented background suppression, resulting in dense output Point Clouds with up to 4 hits.

1.2.2 Imaging modes

Apart from the LIDAR sensor, the L3CAM device provides additional imaging modes that are complementary between them. In such a way, the L3CAM device offers information from different natures and working principles which are referred to as imaging modes. Obtaining information from different imaging modes and sensors is crucial for improving the overall performance and reliability of L3CAM.

Up to date, L3CAM versions can be mounted with up to 3 complementary passive imaging modes:

1.2.2.1 RGB or colour

The conventional colour camera senses the colour information of the environment using a CCD or a CMOS sensor with a Bayer filter (Figure 6). This imaging mode senses the amount of light, thus intensity, in the VIS domain (400 – 700 nm) of the spectrum.

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	16/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

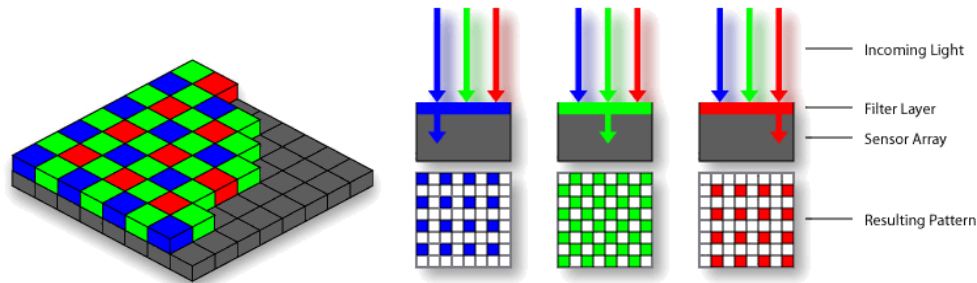


Figure 6. Bayer filter mounted on an imaging sensor.
Image from <https://commons.wikimedia.org/wiki/File:BayerPatternFiltration.png>

The high-resolution colour images provided by this imaging mode mimic human eye perception, hence the importance of this mode for AI perception and assisted computer visual applications.

1.2.2.2 Polarimetric mono

Polarization is the vectorial property of light related to the geometrical orientation of the transverse wave's oscillations, which is perpendicular to its direction of motion. Light can be unpolarized (radiation from many light sources like the sun) or either linearly, circularly or elliptically polarized. Polarizers are materials that let a certain polarization state pass through, thus they act as filters. Hence, polarimetric cameras sense the polarization state of light using combinations of orthogonal polarizers. In particular, the L3CAM's polarimetric mono camera uses an array of micro-polarizers set as a Bayer filter for sensing four different polarization states in the VIS and slightly in the NIR (>780 nm) domain at once, as Figure 7 presents.

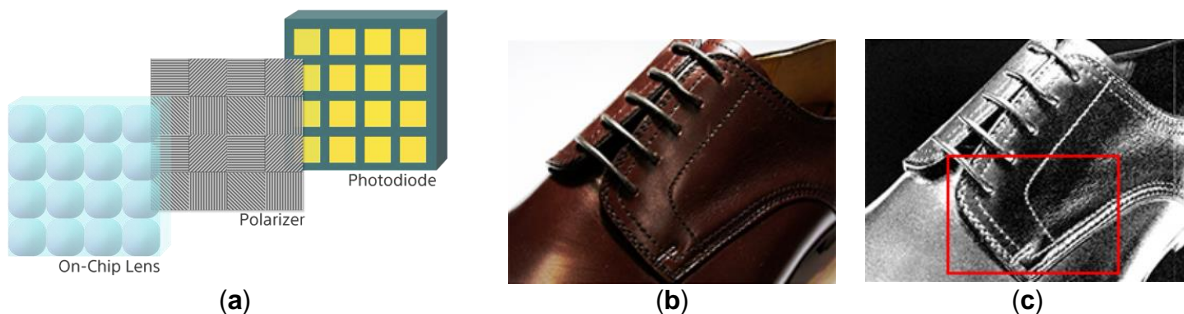


Figure 7. Polarimetric imaging examples.
(a) Microscopic structure of the Sony IMX264MZR polarsens sensor. (b) Conventional colour image.
(c) Polarimetric image representing the Degree Of Linear Polarization (DOLP) that enhances contrast.
Images from <https://www.sony-semicon.co.jp/e/products/IS/industry/technology/polarization.html>.

Polarization is imperceptible for human vision but many animals are sensitive to this light's property because it helps to distinguish between materials. Usually, metals and some artificial materials that keep polarization or polarize in a certain state, clearly differentiate from natural objects that depolarize light. Thus, this imaging mode provides additional contrast of the environment in high resolution as well.

1.2.2.3 Polarimetric colour

Whereas the previous imaging mode senses the total amount of light with a certain polarization state in the whole VIS domain, polarimetric colour cameras are capable of also splitting such information in colours. They combine the micro-polarizers array with a Bayer filter for obtaining the polarization state at the three-primary colour RGB wavelengths as Figure 8 depicts below.



Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	17/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

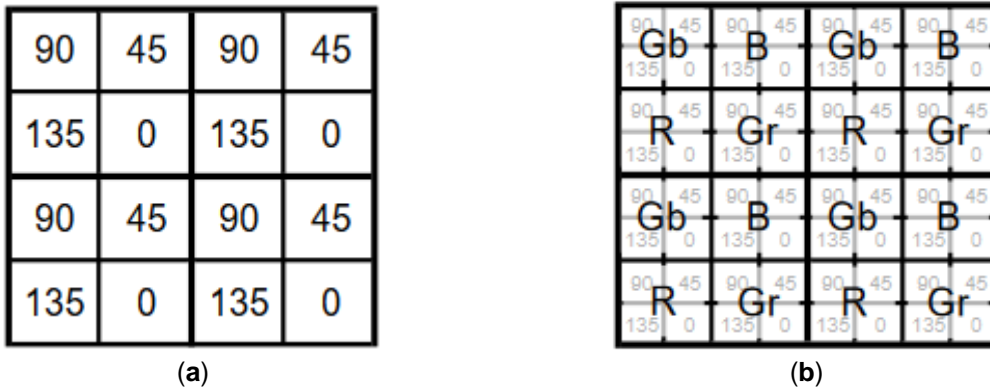


Figure 8. Scheme of the microscopic filter structure of the Sony polarsens sensors.
(a) Polarimetric mono IMX264MZR. (b) Polarimetric colour IMX264MYR.

In line with the above imaging mode, this one not only provides polarization information to the overall perception but also colour information at once from a unique sensor.

1.2.2.4 LWIR or thermal

Contrary to all the previous imaging modes that work in the VIS and NIR domain, LWIR imaging mode is sensitive to the long-wave IR wavelengths (7 – 14 μm). Consequently, this imaging mode provides thermal information about the environment. Temperature, besides being complementary, offers robustness to lighting conditions and to challenging environments like fog, smoke, rain and dust for example. Firstly, temperature variations in the LWIR range are extremely slow compared to varying illumination conditions and it does not require external illumination since objects always radiate in the LWIR domain. Secondly, scattering media responsible for reducing visibility interacts differently with the LWIR wavelengths than with the VIS and NIR ones due to particle size.




Figure 9. Examples of outdoor LWIR thermal images.

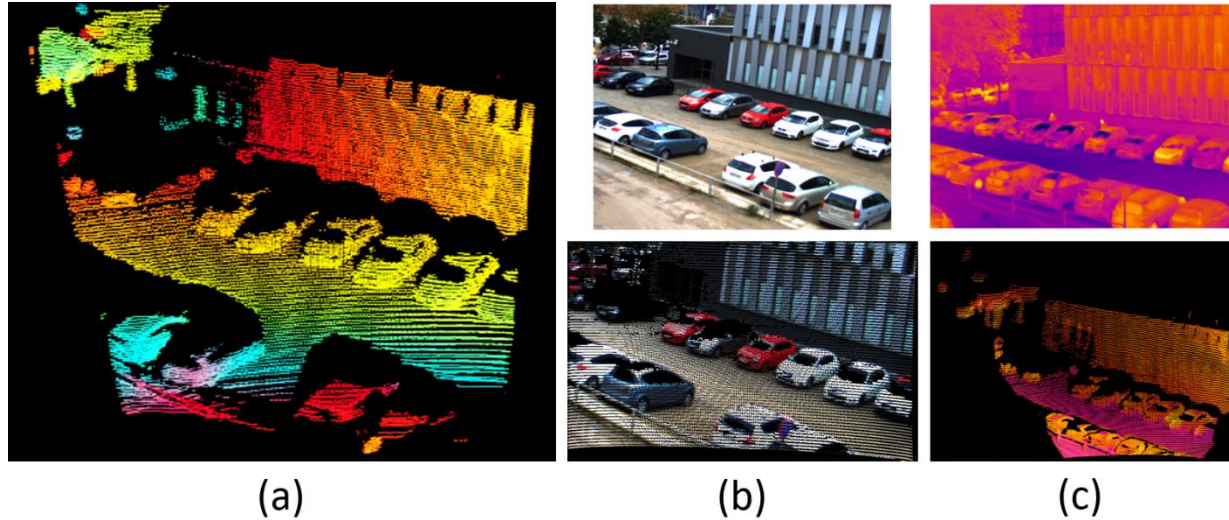
Hence, this imaging mode is robust to external conditions, it does not even need external illumination, and to hazardous conditions where VIS and NIR sensors might fail to work properly.

1.2.3 Data fusion and embedded processing (optional features)

The L3CAM device optionally offers data from all its available sensors precisely and accurately fused, what is called congruent data fusion, according to the acquired product version. This is thanks to an industry-leading multisensory intrinsic calibration and the mechanical robustness and embeddedness of

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	18/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

the device. This novel multimodal calibration process yields low parallax error that outperforms the state-of-the-art data fusion devices for all the available imaging modes.



*Figure 10. Multimodal fusion between the L3CAM LIDAR and different imaging modes.
(a) Original Point Cloud coloured with distance. (b) Colour and (c) Thermal image with the corresponding fused Point Cloud.*

Thanks to its powerful embedded processing unit, the L3CAM device is capable of offering this congruent data fusion in real-time although it can also be computed offline. This unit synchronously gathers data from all sensors with 10 ns high-resolution timestamps and short time discrepancy. Moreover, depending on the product version, the L3CAM software incorporates advanced and accelerated AI perception algorithms for real-time object detection and tracking that benefit from the congruent data fusion for improving its overall performance and reducing the rate of false alarms.

Version:	1.8		Page:	19/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

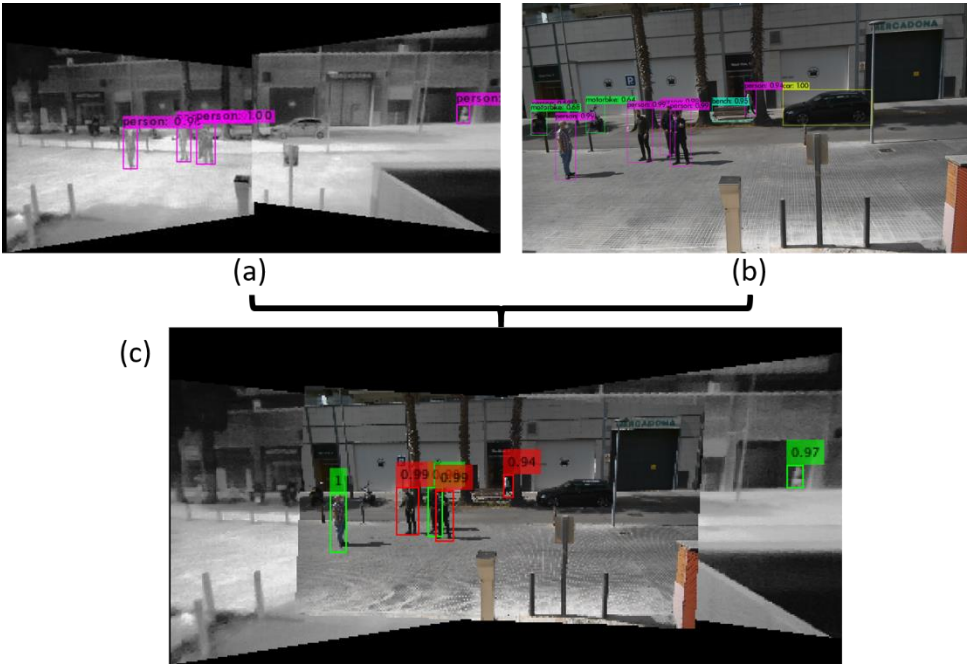


Figure 11. Enhanced AI pedestrian perception of an outdoor scene thanks to data fusion. (a) Perception on a panoramic thermal image. (b) Perception on a colour image. (c) Perception on a colour + thermal image where the colour of the bounding boxes determines which sensor provides greater detection confidence: (green) thermal detections and (red) colour detections.



Version:	1.8	 BEAM\GINE	Page:	20/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

2 GETTING STARTED

2.1 SAFETY WARNINGS AND CAUTIONS

The L3CAM product fulfills the class 1 classification according to the requirements of IEC 60825-1: 2014 / 2007 (Safety of Laser Products).



WARNING:

"DANGER - Laser Radiation - Do Not Stare Into Beam or View Directly With Optical Instruments, for example, eye loupes, magnifiers, microscopes or binoculars".

The light source from this sensor emits invisible laser radiation. Avoid direct exposure to the laser light source.

2.2 LIST OF MATERIALS

The following items are delivered with the purchase of L3CAM:

- L3CAM multimodal sensor.
- Connection cable.
- AC/DC converter.
- SW integration library.
- User guide.

The configuration of the L3CAM includes the sensors specified in the test report document provided with each unit following the product description as shown in APPENDIX 6.



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	21/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

2.3 SYSTEM SETUP

Hereinafter we describe the steps required to set up L3CAM, connect it to the system host and configure the network.

2.3.1 System connection

The L3CAM cable to connect the multimodal sensor to the host has a custom connector on the L3CAM side and an RJ45 connector and a power connector on the host side.

The connection of the system is described in the following figures.

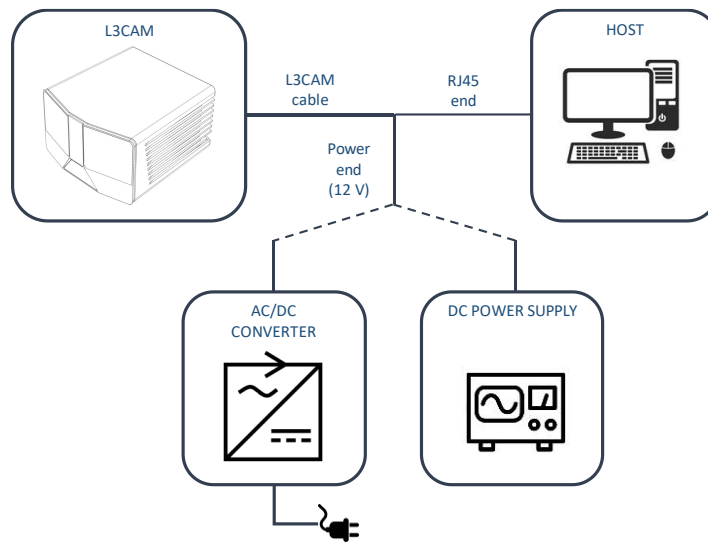


Figure 12. System connection.

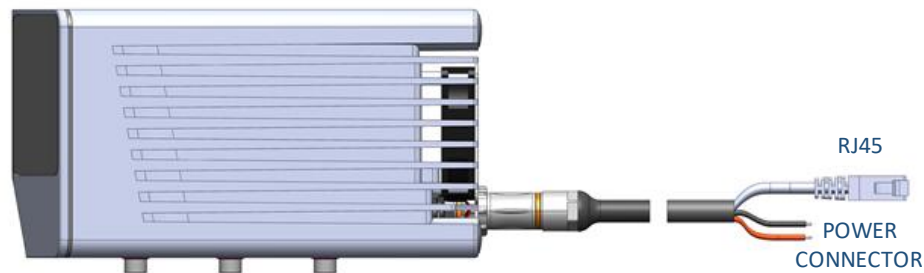


Figure 13. L3CAM side view.

Connect the device to a DC power supply with 12V and 5A, the wire with the red end in the positive and the wire with the black end in the negative as shown in the next figure.

Version:	1.8		Page:	22/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual



Figure 14 Power Supply Connection

2.3.2 Windows host configuration

To modify the network configuration when using Windows PC, use the control panel. First open the **control panel** menu.

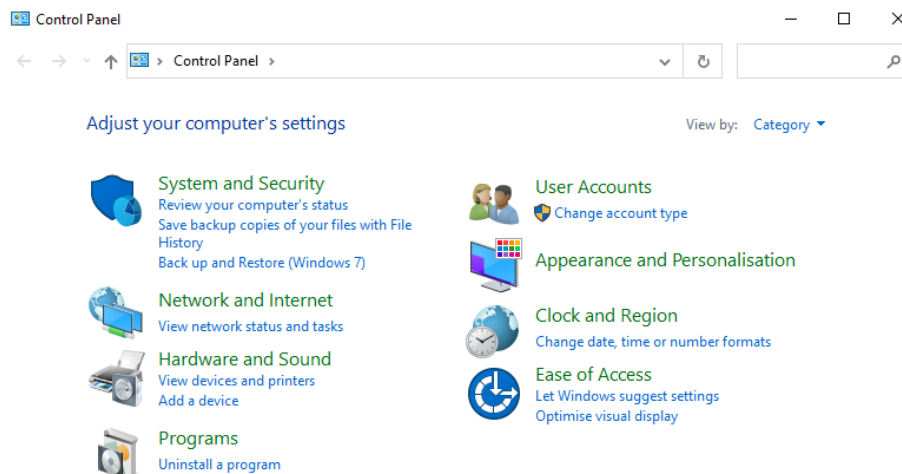


Figure 15. Windows control panel.



Version:	1.8	 BEAM\A\GINE L3CAM User manual	Page:	23/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

Click on the **Network and Internet** option.

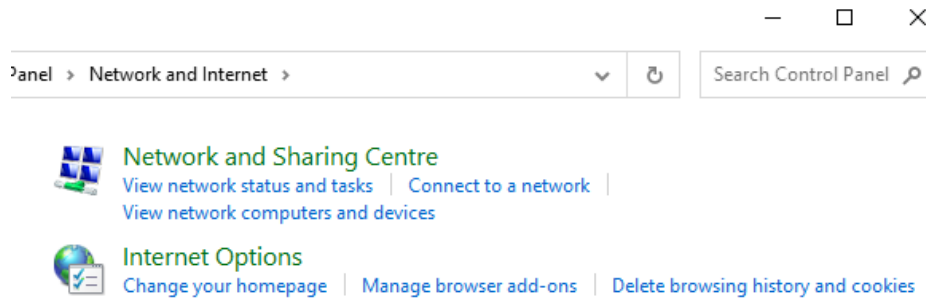


Figure 16. Windows Network and Internet menu.

Click on the **Network and Sharing Centre** option to open the menu with all the Ethernet devices in the Windows PC.

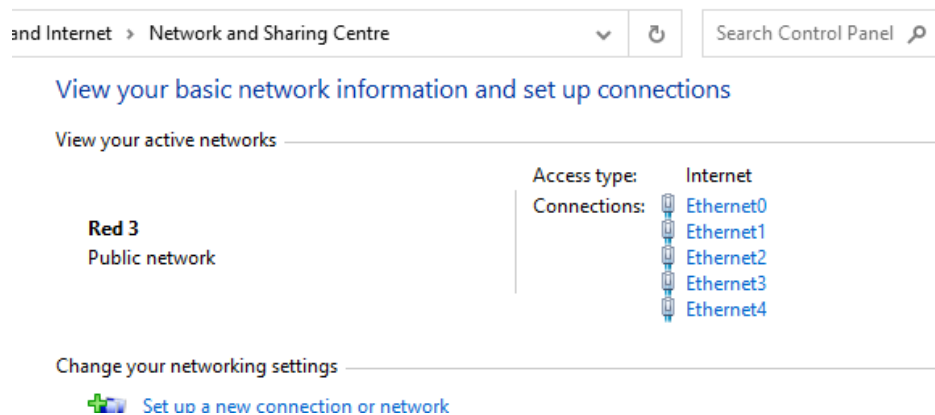


Figure 17. Windows Network and Sharing Centre.



Version:	1.8	 L3CAM User manual	Page:	24/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

Select the network interface where the L3CAM device is connected. **Remember to power on the device, otherwise the interface may not appear.**

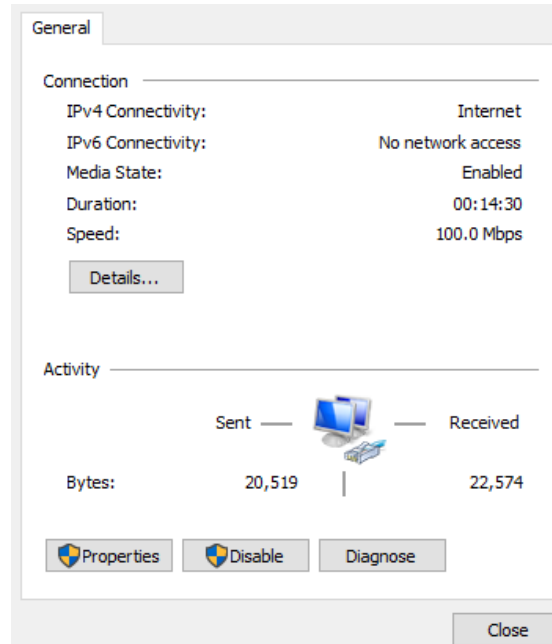


Figure 18. Windows Network interface configuration

Version:	1.8	 L3CAM User manual	Page:	25/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

In the new opened window, click on the **Properties** button to open the configuration options window.

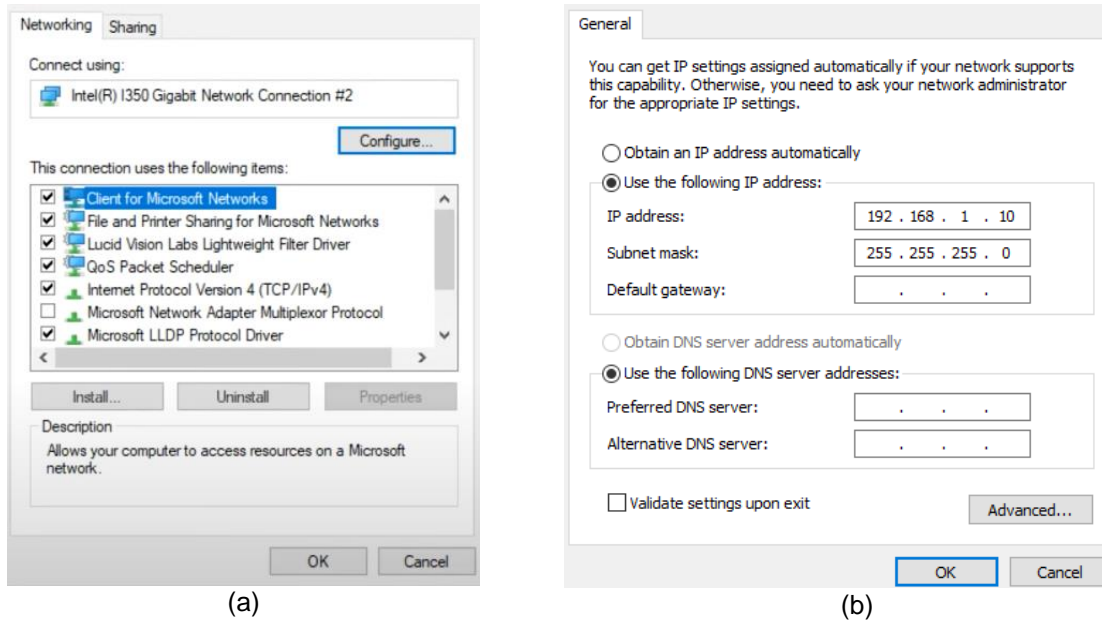


Figure 19. Windows, (a) Network interface configuration, (b) IP address configuration

Click on the **Internet Protocol Version 4 (TCP/IPv4)** option (a), this will open the address configuration window (b). In the address configuration window, select **Use the following IP address:** and configure the IP address and the Subnet mask as shown in the image.

Click on the **OK** button and don't close the (a) window, this will be used to configure the Jumbo Frames.

The device sends the point clouds and image information in UDP packets with a maximum size of 8004 Bytes, to be able to receive these packets, jumbo frames must be enabled. Make sure your Ethernet adapter supports a jumbo frame of size 9000 bytes.

Version:	1.8	 L3CAM User manual	Page:	26/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

To enable jumbo frames on the NIC, click on the **Configure...** button of the Ethernet interface properties to open the advanced settings.

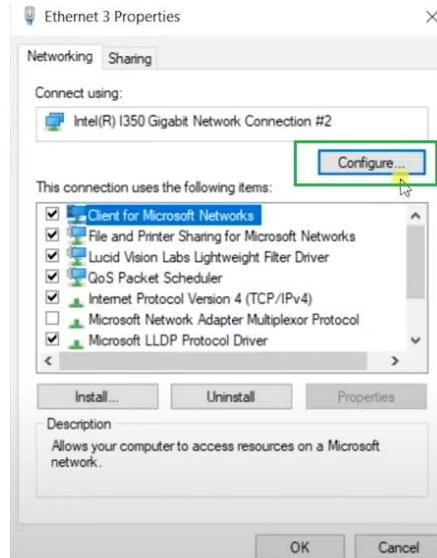


Figure 20. Windows, Access advanced network interface properties.

Switch to the **Advanced** tab, then search for the **Jumbo Packet** configuration, finally change the value to **9014 Bytes**.

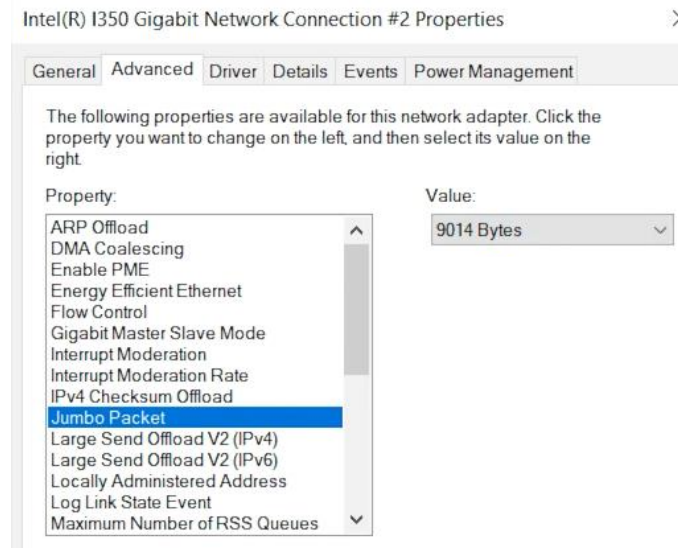


Figure 21. Modify jumbo packet size in Windows.

Version:	1.8	 L3CAM User manual	Page:	27/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

If available, modify also the **Receive Buffers** parameter, increasing the value to **2048**.

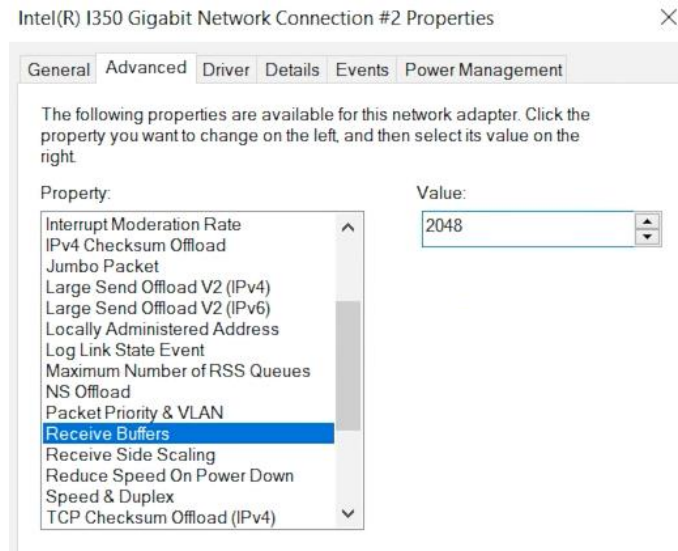


Figure 22. Change receive buffers settings in Windows.

Finally, click on **OK** button, and close all the windows to execute the changes. The Windows host will be ready to receive the data streams of the L3CAM.

Version:	1.8	 L3CAM User manual	Page:	28/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

2.3.3 Linux host configuration

The network configuration of the Linux PC can be done with the graphical tools.

First open the settings configuration and select the Network option to list the available network interfaces.

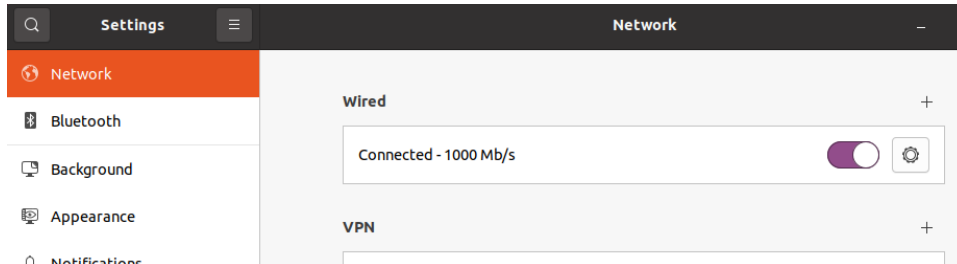


Figure 23. Ubuntu 20 Network settings menu.

Click on the gear button at the right, in the network interface where the device is connected, to open the configuration.

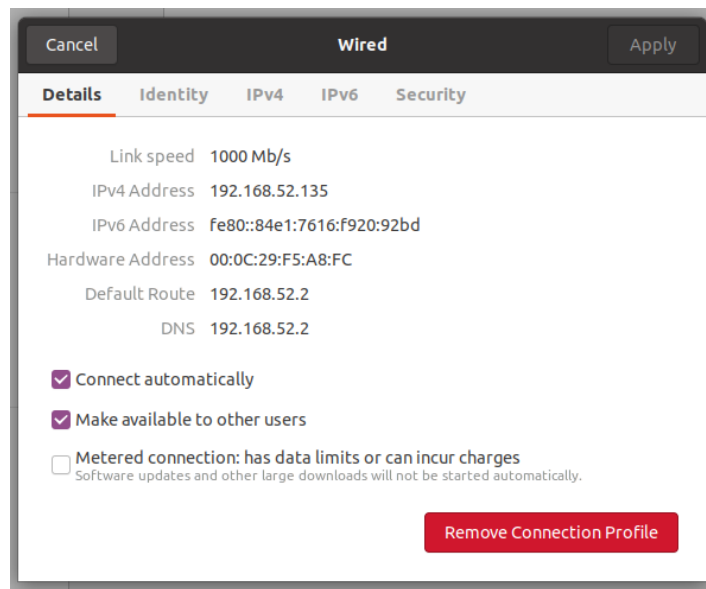
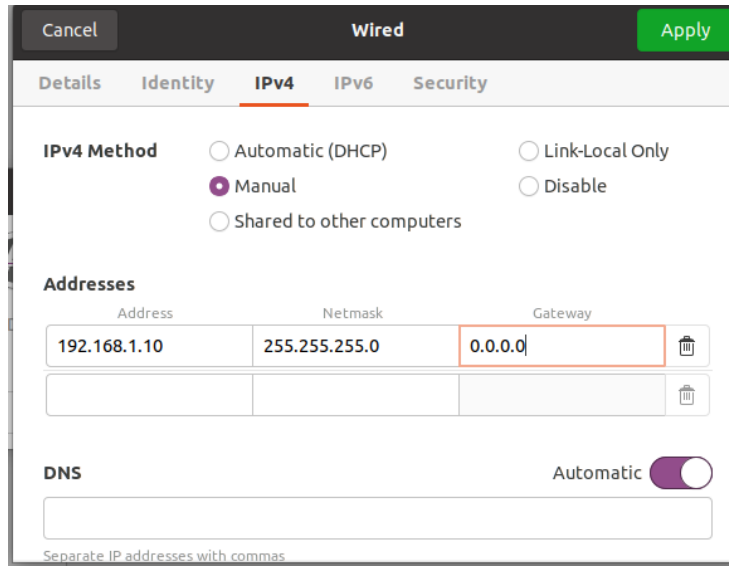


Figure 24. Ubuntu 20 Network interface configuration.

In the IPv4 tab, first change the IPv4 method from DHCP to Manual, then change the configuration to an IP address and netmask that match the subnet of the device, in the example the address 192.168.1.10 will be used.

Version:	1.8	 L3CAM User manual	Page:	29/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024



Cancel **Wired** Apply

Details Identity **IPv4** IPv6 Security

IPv4 Method

☐ Automatic (DHCP) ☐ Link-Local Only

☒ **Manual** ☐ Disable

☐ Shared to other computers

Addresses

Address	Netmask	Gateway
192.168.1.10	255.255.255.0	0.0.0.0

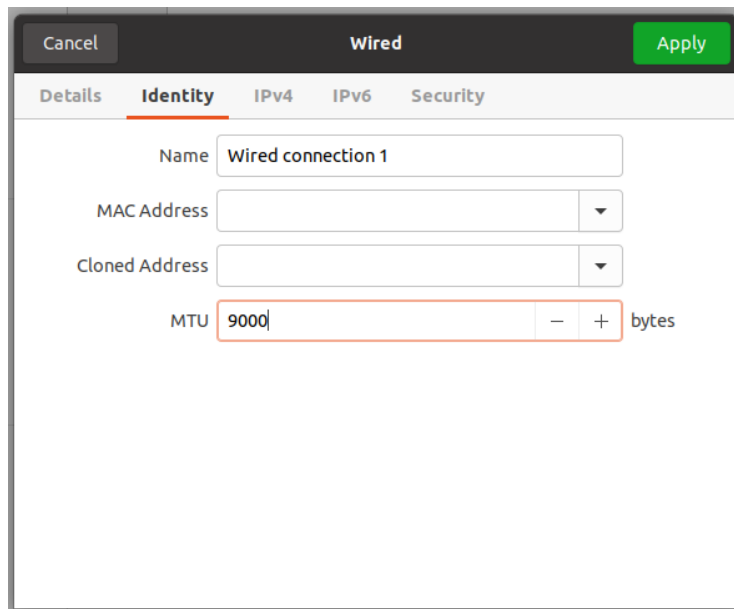
DNS Automatic ☒

Separate IP addresses with commas

Figure 25. Ubuntu 20 IP address configuration

The device sends the point clouds and image information in UDP packets with a maximum size of 8004 Bytes, to be able to receive these packets, jumbo frames must be enabled. Make sure your Ethernet adapter supports a jumbo frame of size 9000 bytes.

In the current network interface configuration window, select the tab identity, and modify the MTU field to 9000 bytes.



Cancel **Wired** Apply

Details **Identity** IPv4 IPv6 Security

Name: Wired connection 1

MAC Address: [dropdown]

Cloned Address: [dropdown]

MTU: 9000 bytes

Figure 26. Ubuntu 20 MTU configuration.



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	30/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

Click on the Apply button and the window will automatically close. To make the changes effective, restart the NIC, to do so, click on the toggle button twice.



Figure 27. Ubuntu 20 Network restart (a) power off (b) power on.

Finally, the receiving buffers size must be modified to **268435456**, this must be done using the terminal.

The terminal can be opened by pressing **ctrl + alt + t** or by pressing the windows key and searching for terminal.

Once the terminal is open paste the following commands:

```
sudo sh -c "echo 'net.core.rmem_default=268435456' >> /etc/sysctl.conf"
sudo sh -c "echo 'net.core.rmem_max=268435456' >> /etc/sysctl.conf"
sudo sysctl -p
```

```
beamagine@ubuntu: ~
beamagine@ubuntu:~$ sudo sh -c "echo 'net.core.rmem_default=268435456' >> /etc/sysctl.conf"
[sudo] password for beamagine:
beamagine@ubuntu:~$ sudo sh -c "echo 'net.core.rmem_max=268435456' >> /etc/sysctl.conf"
beamagine@ubuntu:~$ sudo sysctl -p
net.core.rmem_default = 268435456
net.core.rmem_max = 268435456
```

Figure 28. Configuration of the receiving buffers size.

Now the Linux PC is ready to communicate and receive the point clouds and images from the device.



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	31/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

3 API USER GUIDE

3.1 LIBRARY OVERVIEW

The library for the integration of L3CAM in the system host (libL3Cam) has been designed to encapsulate all the complexities of the system, automating many functions to simplify the use of the system. The interface exposed by the API has been carefully designed to facilitate its use.

Before proceeding with the details of the API let's establish the following set of categories of the functions:

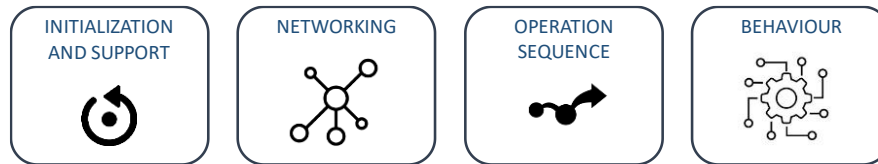


Figure 29. Categories of the functions of the libL3Cam API.

This set of functions is used to initialize and operate L3CAM.

The initialization sequence is described in the following figure.



Figure 30. L3CAM initialization sequence.

The operation sequence is as follows:

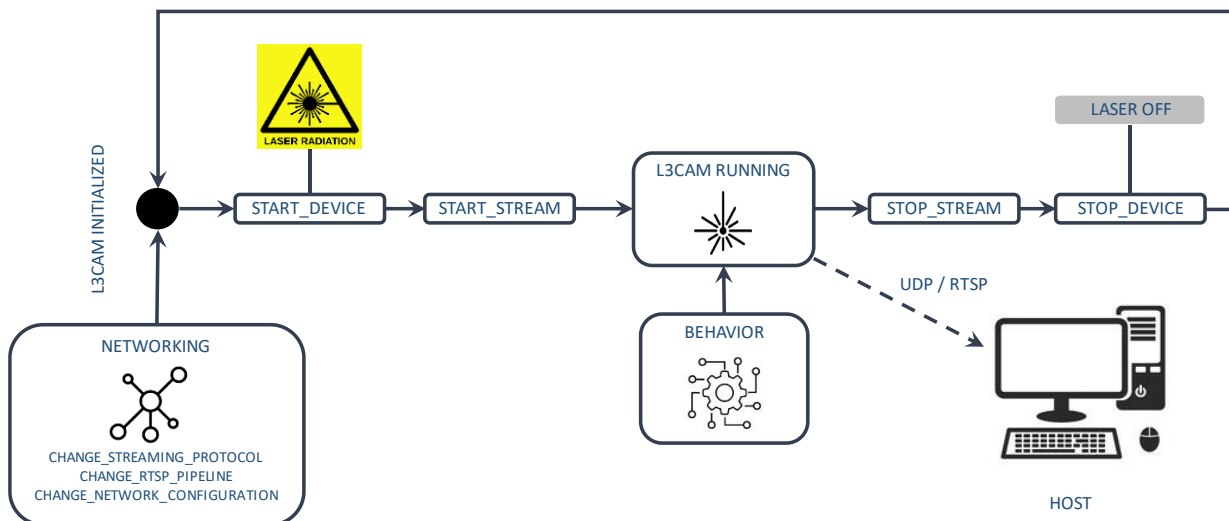


Figure 31. Operation sequence.

Version:	1.8	 L3CAM User manual	Page:	32/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

After starting the device, we ask it to start streaming to have it fully operational and sending UDP/RTSP packets to the host computer as will be described in the section below. In this state, we can use the functions under the behaviour category to change its comportment.

LibL3Cam uses the following ports for communications

Table 4. libL3Cam ports.

PROTOCOL	PORT	CONFIGURATION
TCP	6000	Fixed
UDP	6050 (LIDAR)	Fixed
UDP	6060 (Allied Wide RGB)	Fixed
UDP	6020 (Allied Narrow RGB)	Fixed
UDP	6030 (LWIR)	Fixed
UDP	6031 (LWIR temperature data)	Fixed
RTSP	5040 (LIDAR)	Reconfigurable
RTSP	5030 (Allied Wide RGB)	Reconfigurable
RTSP	5010 (Allied Narrow RGB)	Reconfigurable
RTSP	5020 (LWIR)	Reconfigurable

TCP is used internally by lib3Cam and is transparent to the user. However, the system host must have the required port available.

3.2 UDP PROTOCOL AND DATA DESCRIPTION

In this mode, L3CAM sends non-compressed LIDAR and multi-modal sensor data with the UDP protocol. Since each sensor uses a different UDP port as described in Table 4, it is recommended to implement a different thread for each sensor to properly receive the data.

To optimize networking resources, it is required to enable jumbo frames as described in section 2.3.

3.2.1 Point cloud

The transmission sequence of a point cloud frame is described in Figure 32.



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

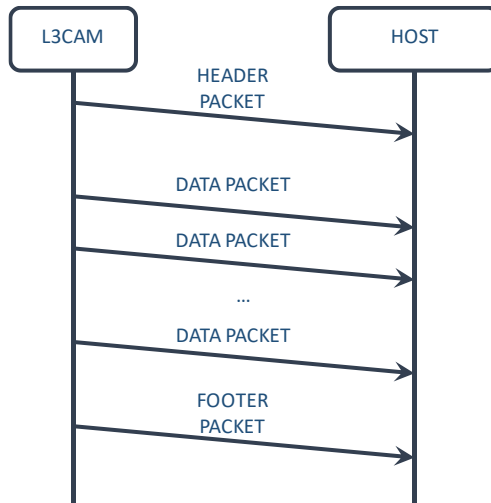


Figure 32. Transmission sequence of point cloud data.

The transmission of the frame starts with a header packet, is followed by a sequence of data packets and is finalized with a footer packet.

The header packet is as follows:

Table 5. Point cloud frame header.

FIELD	STX	Total number of points	Ctrl value 1	Ctrl value 2	Timestamp
SIZE (Bytes)	1	4 (int32_t)	4 (int32_t)	4 (int32_t)	4 (uint32_t)
Value	0x02				

- STX is a header packet identifier with value 0x02.
- Total number of points: the number of points that will be sent in the sequence of data packets.
- ctrl value 1: reserved.
- ctrl value 2: reserved.
- Timestamp: an integer that represents the time when the frame was received. The time is in the L3CAM Jetson TX2 time and contains **hhmmsszzz**. For example, a frame received at 18:32:12:346 will return the timestamp value 183212346;

The data packets are organized as described in the following table:

Table 6. Data packet structure.

FIELD	Number of points in packet	Point 1	Point 2	...	Point N
SIZE (Bytes)	4 (int32_t)	20 (point)	20 (point)	20 (point)	20 (point)

Each data packet contains 400 points (packet size = 8000 + 4 bytes) but the last one, which can contain fewer points depending on the detected LiDAR points.

Version:	1.8		Page:	34/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

Each point has the following structure:

Table 7. Point structure.

FIELD	X	Y	Z	Intensity	RGB
SIZE (Bytes)	4	4	4	4	4
	(int32_t)	(int32_t)	(int32_t)	(int32_t)	(int32_t)

- X, Y, Z are the coordinates of the hit¹ in the coordinate system described in Figure 33;
- Intensity²;
- RGB is a field that can represent different features depending on the configuration:
 - colour scale³ for the distance of the hit (d),
 - colour scale for the intensity of the hit,
 - RGB value of the sensor with which the point cloud has been fused.

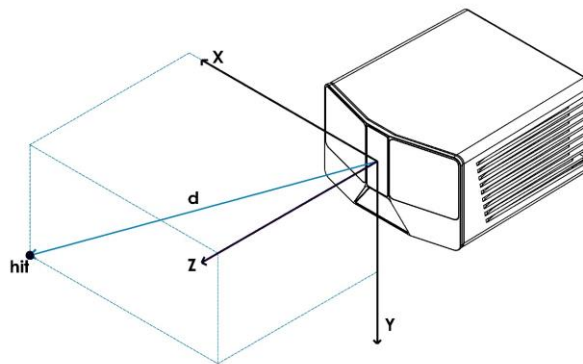


Figure 33. Point cloud reference coordinate system.

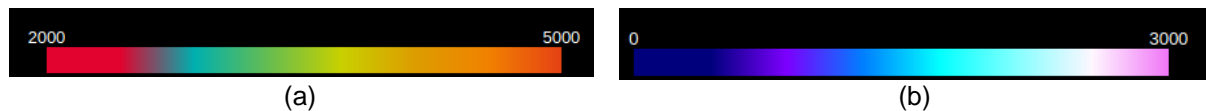


Figure 34. Colour scale for distance (a) and intensity (b).

When all data packets have been sent, L3CAM signals the end of transmission of the point cloud frame with a footer package with STX = 0x03:

Table 8. Point cloud frame footer.

FIELD	ETX
SIZE (Bytes)	1
Value	0x03

In APPENDIX 1 there is a sample code describing how to read the point cloud frame from the host.

¹ See description in the definitions section.

² Intensity is the same as the TOT value described in 1.2.

³ Colour scales can be changed with the API.



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

3.2.2 Image

The transmission sequence of an image frame is described in Figure 35. Note that in systems with the perception functions enabled, the image data is complemented with the result of the perception AI algorithms.

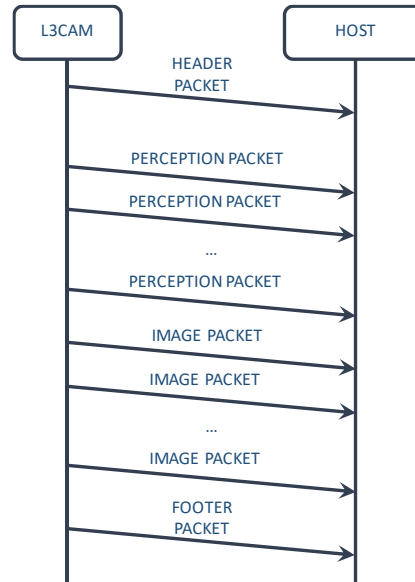


Figure 35. Transmission sequence of image data with results from perception functions.

The transmission begins with a header packet. Then, if the perception functions have detected objects in the image, L3CAM transmits as many perception packets as necessary. The transmission continues with the image packets and ends with a footer packet.

The header packet is as follows:

Table 9. Image frame header.

FIELD	STX	HE	WI	CHANNELS	TIMESTAMP	N_DET
SIZE (Bytes)	1	2 (int16_t)	2 (int16_t)	1 (byte)	4 (uint32_t)	1 (byte)
Value	0x02					

- STX is a header packet identifier with value 0x02.
- HE: image height in pixels.
- WI: image width in pixels.
- Channels: number of channels in the image.
- Timestamp: an integer that represents the time when the frame was received. The time is in the L3CAM Jetson TX2 time and contains **hhmmsszzz**. For example, a frame received at 18:32:12:346 will return the timestamp value 183212346.
- N_DET: number of object detections.

Version:	1.8		Page:	36/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

If the perception functions are enabled and L3CAM detects objects, it sends one perception packet per object detected. Each perception packet contains the 2D coordinates of the bounding rectangle of the detected object along with the confidence level of the detection. The structure of the perception packet is described in Table 10.

Table 10. Perception packet.

FIELD	Conf	X0	Y0	HE	WI	Label
SIZE (Bytes)	2 (uint16_t)	2 (int16_t)	2 (int16_t)	2 (int16_t)	2 (int16_t)	2 (uint16_t)

- Conf: confidence level; range [0-100].
- X0: x coordinate of the origin of the bounding box of the detected object (in pixels).
- Y0: y coordinate of the origin of the bounding box of the detected object (in pixels).
- HE: height of the bounding box (in pixels).
- WI: width of the bounding box (in pixels).
- Label: Index that represents the class of the detected object.

After the perception packets, or in case there are no perception packets, L3CAM starts sending the image packets. The image packets have a size of 8000 bytes. The last image packet can have a smaller size.

The end of transmission (footer) packet is:

Table 11. Point cloud frame footer.

FIELD	ETX
SIZE (Bytes)	1
Value	0x03

In APPENDIX 2 there is a sample code describing how the read images from the UDP stream.

3.2.3 Thermal data

Starting with beamagine_app version 2.2.9 the temperatures obtained by the thermal camera are being sent as an array of float values with the same size as the thermal image. The transmission sequence of a temperature array is described in Figure 36.



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

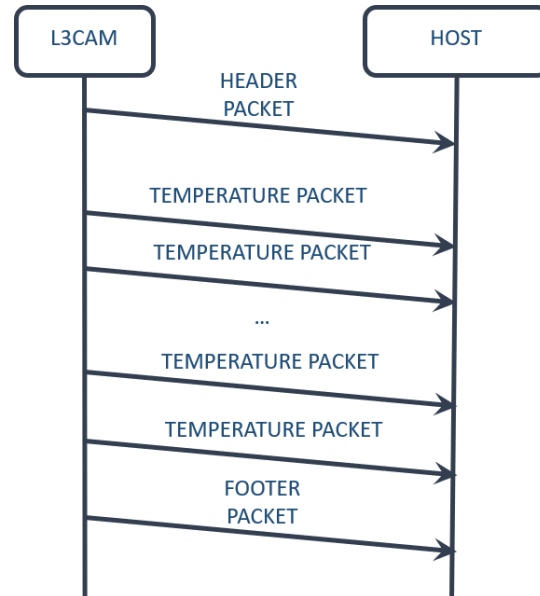


Figure 36 Transmission sequence of temperatures

The transmission begins with a header packet, this packet content is shown in table 12

Table 12 Temperature header packet

FIELD	STX	HEIGHT		WIDTH		TIMESTAMP
SIZE (Bytes)	1	2 (int16_t)		2 (int16_t)		4 (uint32_t)
Value	0x02	0x01	0x40	0x00	0xF0	

- STX is a header packet identifier with value 0x02.
- HEIGHT is the height of the array
- WIDTH is the width of the array
- TIMESTAMP is an integer that represents the time when the frame was received. The time is in the L3CAM Jetson TX2 time and contains **hhmmsszzz**. For example, a frame received at 18:32:12:346 will return the timestamp value 183212346.

After sending the header packet, the temperature data is being send in packets of 800 Bytes, except for the last packet that has less size, then the end of transmission (footer) packet is sent.

In APPENDIX 3 there is a sample code describing how to read the temperature arrays from the UDP stream.

Version:	1.8	 L3CAM User manual	Page:	38/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

3.3 RTSP PROTOCOL AND DATA DESCRIPTION

In this mode, L3CAM generates a separate pipeline for each sensor using the GStreamer API. The internal pipelines are shown in the next table. The pipelines can be modified using the API calls described in the API section. By default, L3CAM sends the video streams using the H264 compression.

Table 13. GStreamer pipeline.


SENSOR	GSTREAMER PIPELINE
LIDAR	appsrc ! video/x-raw,format=BGR,width=720,height=480 framerate=10/1 ! videoscale ! videoconvert ! nvvidconv ! nvv4l2h264enc control-rate=constant_bitrate bitrate=8000000 preset-level=2 maxperf-enable=1 ! video/x-h264, stream-format=byte-stream ! h264parse ! rtph264pay pt=96 mtu=8950 ! udpsink host=@host_ip port=5040
Polarimetric Camera	appsrc ! video/x-raw,format=BGR,width=2448,height=2048 framerate=10/1 ! videoscale ! videoconvert ! nvvidconv ! nvv4l2h264enc control-rate=constant_bitrate bitrate=8000000 preset-level=2 maxperf-enable=1 ! video/x-h264, stream-format=byte-stream ! h264parse ! rtph264pay pt=96 mtu=8950 ! udpsink host=@host_ip port=5030
RGB Camera	appsrc ! video/x-raw,format=BGR,width=1920,height=1080 framerate=10/1 ! videoscale ! videoconvert ! nvvidconv ! nvv4l2h264enc control-rate=constant_bitrate bitrate=8000000 preset-level=2 maxperf-enable=1 ! video/x-h264, stream-format=byte-stream ! h264parse ! rtph264pay pt=96 mtu=8950 ! udpsink host=@host_ip port=5010
LWIR Thermal Camera	appsrc ! video/x-raw,format=BGR,width=320,height=240 framerate=10/1 ! videoscale ! videoconvert ! nvvidconv ! nvv4l2h264enc control-rate=constant_bitrate bitrate=8000000 preset-level=2 maxperf-enable=1 ! video/x-h264, stream-format=byte-stream ! h264parse ! rtph264pay pt=96 mtu=8950 ! udpsink host=@host_ip port=5020
Allied Wide RGB Camera 1800 U-508c	appsrc ! video/x-raw,format=BGR,width=1232,height=1028 framerate=10/1 ! videoscale ! videoconvert ! nvvidconv ! nvv4l2h264enc control-rate=constant_bitrate bitrate=8000000 preset-level=2 maxperf-enable=1 ! video/x-h264, stream-format=byte-stream ! h264parse ! rtph264pay pt=96 mtu=8950 ! udpsink host=@host_ip port=5030
Allied Narrow RGB Camera 1800 U-319c	appsrc ! video/x-raw,format=BGR,width=2064,height=1554 framerate=10/1 ! videoscale ! videoconvert ! nvvidconv ! nvv4l2h264enc control-rate=constant_bitrate bitrate=8000000 preset-level=2 maxperf-enable=1 ! video/x-h264, stream-format=byte-stream ! h264parse ! rtph264pay pt=96 mtu=8950 ! udpsink host=@host_ip port=5010



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	39/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

3.3.1 Point cloud

The LIDAR image resolution is not a video standard, and to be able to send it over RTSP using GStreamer API it is necessary to add black padding on the right and the bottom of the image before sending it over RTSP. **This image mode is available only on demand, by default is not supported.**

The result is a 720x480 pixels image sent over RTSP, and this black padding can be removed in the same pipeline used to receive the RTSP stream as shown in the example below.

```
gst-launch-1.0 udpsrc address="@host_ip" port=5040 ! application/x-
rtp,media=video,payload=96,clock-rate=90000,encoding-name=H264,framerate=10/1 !
rtph264depay ! h264parse ! decodebin ! videocrop top=0, left=0, right=40,
bottom=230 ! autovideosink sync=false show-preroll-frame=false
```

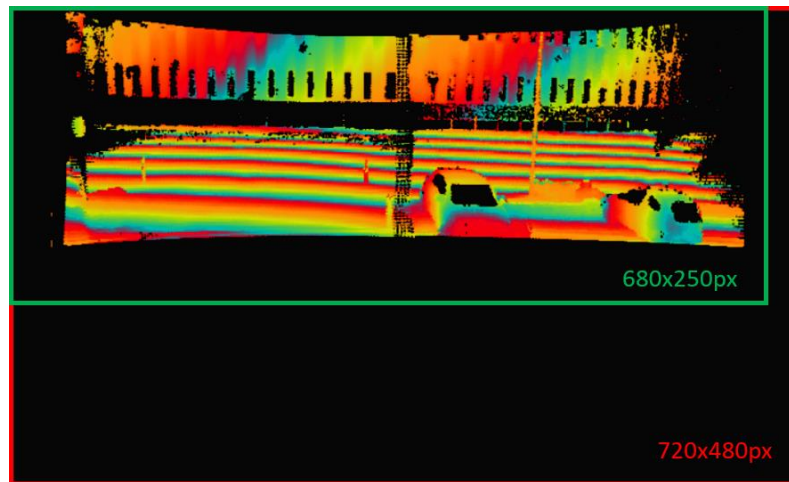


Figure 37. Point cloud and cropping area.

Version:	1.8		Page:	40/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

3.3.2 Image Polarimetric

To receive the RTSP video stream from the Polarimetric camera, two pipelines can be used, depending if the camera is streaming in raw format or in H264 format.

For H264 format use the following pipeline

```
gst-launch-1.0 udpsrc address="@host_ip" port=5030 ! application/x-rtp,media=video,payload=96,clock-rate=90000,encoding-name=H264,framerate=10/1 ! rtpH264depay ! h264parse ! decodebin ! autovideosink sync=false show-preroll-frame=false
```

For RAW format use the following pipeline

```
gst-launch-1.0 udpsrc address="@host_ip" port="5030" caps="application/x-rtp,media=(string)video, clock-rate=(int)90000, encoding-name=(string)RAW,sampling=(string)BGR, depth=(string)8, width=(string)2448, height=(string)2048,colorimetry=(string)BT601-5, payload=(int)96, framerate=10/1" ! rtpvrawdepay ! tee name=t t. ! queue ! videoconvert ! autovideosink window-x=640, window-y=480, sync=false show-preroll-frame=false
```



Figure 38. Polarimetric Bayer image. See image format in APPENDIX 4.

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	41/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

3.3.3 Image RGB

To receive the RTSP video stream from the RGB camera the following pipeline can be used.

```
gst-launch-1.0 udpsrc address="@host_ip" port=5010 ! application/x-
rtp,media=video,payload=96,clock-rate=90000,encoding-name=H264,framerate=10/1 !
rtph264depay ! h264parse ! decodebin ! autovideosink sync=false show-preroll-
frame=false
```



Figure 39. RGB image from econ camera

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	42/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

3.3.4 Thermal image

To receive RTSP video stream from the Thermal camera the following pipeline can be used

```
gst-launch-1.0 udpsrc address="@host_ip" port=5020 ! application/x-
rtp,media=video,payload=96,clock-rate=90000,encoding-name=H264,framerate=10/1 !
rtph264depay ! h264parse ! decodebin ! autovideosink sync=false show-preroll-
frame=false
```

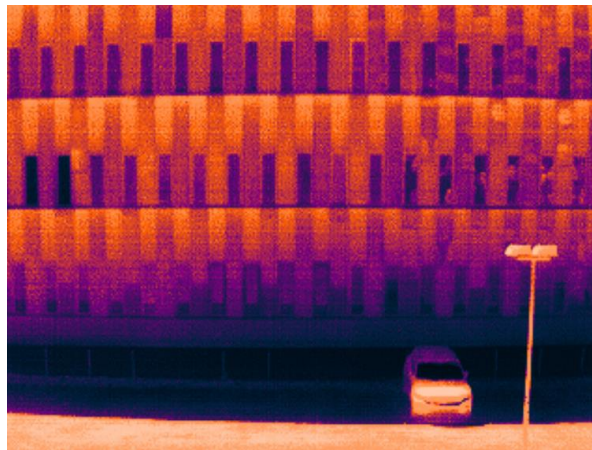


Figure 40. Thermal image using TYRIAN colormap

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	43/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

3.3.5 Image RGB Allied Wide

The Allied sensor 1800 U-508c is configured with the KOWA-LM6JC lens, that generates the wide field of view. The image resolution is 1232x1028.

To receive the RTSP video stream from the allied wide camera the following pipeline can be used.

```
gst-launch-1.0 -v udpsrc port=5030 caps="application/x-rtp, media=(string)video,
clock-rate=(int)90000, encoding-name=(string)H264, payload=(int)96,
height=(string)1028, width=(string)1232, framerate=(fraction)10/1 " !
rtph264depay ! decodebin ! videoconvert ! autovideosink window-width=640, window-
height=480
```



Figure 41. Allied 1800 U-508c with KOWA-LM6JC

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	44/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

3.3.6 Image RGB Allied Narrow

The Allied sensor 1800 U-319c is configured with the M2514-MP2 lens, that generates the narrow field of view. The image resolution is 2064x1554.

To receive the RTSP video stream from the allied narrow camera the following pipeline can be used.

```
gst-launch-1.0 -v udpsrc port=5010 caps="application/x-rtp, media=(string)video,
clock-rate=(int)90000, encoding-name=(string)H264, payload=(int)96,
height=(string)1544, width=(string)2064, framerate=(fraction)10/1 " !
rtph264depay ! decodebin ! videoconvert ! autovideosink window-width=640, window-
height=480
```



Figure 42. Allied 1800 U-319c with M2514-MP2

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	45/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

3.4 API FUNCTIONS LIST

The following tables lists the functions of the library libL3Cam for each sensor divided in different header files. Please note that the code to receive image data and point cloud data is not included in the API. It is the responsibility of the user to implement those functions following the examples provided in the appendixes.

Table 14. API functions libL3Cam.h.

FUNCTION NAME	CATEGORY
registerErrorCallback	Initialization and support
getBeamErrorDescription	Initialization and support
GET_VERSION	Initialization and support
INITIALIZE	Networking
TERMINATE	Finalization support
FIND_DEVICES	Initialization and support
GET_LOCAL_SERVER_ADDRESS	Initialization and support
GET_DEVICE_STATUS	Initialization and support
GET_SENSORS_AVAILABLE	Initialization and support
CHANGE_STREAMING_PROTOCOL	Networking
GET_RTSP_PIPELINE	Networking
CHANGE_RTSP_PIPELINE	Networking
GET_NETWORK_CONFIGURATION	Networking
CHANGE_NETWORK_CONFIGURATION	Networking
POWER_OFF_DEVICE	Operation sequence
START_DEVICE	Operation sequence
STOP_DEVICE	Operation sequence
START_STREAM	Operation sequence
STOP_STREAM	Operation sequence
CHANGE_POINTCLOUD_COLOR	Behaviour
CHANGE_POINTCLOUD_COLOR_RANGE	Behaviour
CHANGE_DISTANCE_RANGE	Behaviour
GET_DEVICE_TEMPERATURES	Behaviour
SET_BIAS_LONG_RANGE	Behaviour
SET_BIAS_SHORT_RANGE	Behaviour
ENABLE_AUTO_BIAS	Behaviour
CHANGE_BIAS_VALUE	Behaviour

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	46/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

Table 15. API functions libL3Cam_allied.h

FUNCTION NAME	CATEGORY
CHANGE_ALLIED_CAMERA_BLACK_LEVEL	Behaviour
CHANGE_ALLIED_CAMERA_EXPOSURE_TIME_US	Behaviour
ENABLE_ALLIED_CAMERA_AUTO_EXPOSURE_TIME	Behaviour
CHANGE_ALLIED_CAMERA_AUTO_EXPOSURE_TIME_RANGE	Behaviour
CHANGE_ALLIED_CAMERA_GAIN	Behaviour
ENABLE_ALLIED_CAMERA_AUTO_GAIN	Behaviour
CHANGE_ALLIED_CAMERA_AUTO_GAIN_RANGE	Behaviour
CHANGE_ALLIED_CAMERA_GAMMA	Behaviour
CHANGE_ALLIED_CAMERA_SATURATION	Behaviour
CHANGE_ALLIED_CAMERA_SHARPNESS	Behaviour
CHANGE_ALLIED_CAMERA_HUE	Behaviour
CHANGE_ALLIED_CAMERA_INTENSITY_AUTO_PRECEDENCE	Behaviour
ENABLE_ALLIED_CAMERA_AUTO_WHITE_BALANCE	Behaviour
CHANGE_ALLIED_CAMERA_BALANCE_RATIO_SELECTOR	Behaviour
CHANGE_ALLIED_CAMERA_BALANCE_RATIO	Behaviour
CHANGE_ALLIED_CAMERA_BALANCE_WHITE_AUTO_RATE	Behaviour
CHANGE_ALLIED_CAMERA_BALANCE_WHITE_AUTO_TOLERANCE	Behaviour
CHANGE_ALLIED_CAMERA_AUTO_MODE_REGION	Behaviour
CHANGE_ALLIED_CAMERA_INTENSITY_CONTROLLER_REGION	Behaviour
CHANGE_ALLIED_CAMERA_INTENSITY_CONTROLLER_TARGET	Behaviour
CHANGE_ALLIED_CAMERA_MAX_DRIVER_BUFFERS_COUNT	Behaviour
GET_ALLIED_CAMERA_BLACK_LEVEL	Behaviour
GET_ALLIED_CAMERA_EXPOSURE_TIME_US	Behaviour
GET_ALLIED_CAMERA_AUTO_EXPOSURE_TIME	Behaviour
GET_ALLIED_CAMERA_AUTO_EXPOSURE_TIME_RANGE	Behaviour
GET_ALLIED_CAMERA_GAIN	Behaviour
GET_ALLIED_CAMERA_AUTO_GAIN	Behaviour
GET_ALLIED_CAMERA_AUTO_GAIN_RANGE	Behaviour
GET_ALLIED_CAMERA_GAMMA	Behaviour
GET_ALLIED_CAMERA_SATURATION	Behaviour
GET_ALLIED_CAMERA_SHARPNESS	Behaviour
GET_ALLIED_CAMERA_HUE	Behaviour
GET_ALLIED_CAMERA_INTENSITY_AUTO_PRECEDENCE	Behaviour
GET_ALLIED_CAMERA_AUTO_WHITE_BALANCE	Behaviour
GET_ALLIED_CAMERA_BALANCE_RATIO_SELECTOR	Behaviour
GET_ALLIED_CAMERA_BALANCE_RATIO	Behaviour
GET_ALLIED_CAMERA_BALANCE_WHITE_AUTO_RATE	Behaviour
GET_ALLIED_CAMERA_BALANCE_WHITE_AUTO_TOLERANCE	Behaviour
GET_ALLIED_CAMERA_AUTO_MODE_REGION	Behaviour
GET_ALLIED_CAMERA_INTENSITY_CONTROLLER_REGION	Behaviour
GET_ALLIED_CAMERA_INTENSITY_CONTROLLER_TARGET	Behaviour
GET_ALLIED_CAMERA_MAX_DRIVER_BUFFERS_COUNT	Behaviour

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	47/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

Table 16. API functions libL3Cam_econ.h


FUNCTION NAME	CATEGORY
SET_RGB_CAMERA_DEFAULT_SETTINGS	Behaviour
CHANGE_RGB_CAMERA_BRIGHTNESS	Behaviour
CHANGE_RGB_CAMERA_CONTRAST	Behaviour
CHANGE_RGB_CAMERA_SATURATION	Behaviour
CHANGE_RGB_CAMERA_SHARPNESS	Behaviour
CHANGE_RGB_CAMERA_GAMMA	Behaviour
CHANGE_RGB_CAMERA_GAIN	Behaviour
CHANGE_RGB_CAMERA_WHITE_BALANCE	Behaviour
CHANGE_RGB_CAMERA_EXPOSURE_TIME	Behaviour
ENABLE_RGB_CAMERA_AUTO_WHITE_BALANCE	Behaviour
ENABLE_RGB_CAMERA_AUTO_EXPOSURE_TIME	Behaviour
CHANGE_RGB_CAMERA_RESOLUTION	Behaviour
CHANGE_RGB_CAMERA_FRAMERATE	Behaviour

Table 17. API functions libL3Cam_polarimetric.h

FUNCTION NAME	CATEGORY
SET_POLARIMETRIC_CAMERA_DEFAULT_SETTINGS	Behaviour
CHANGE_POLARIMETRIC_CAMERA_BRIGHTNESS	Behaviour
CHANGE_POLARIMETRIC_CAMERA_BLACK_LEVEL	Behaviour
CHANGE_POLARIMETRIC_CAMERA_GAIN	Behaviour
ENABLE_POLARIMETRIC_CAMERA_AUTO_GAIN	Behaviour
CHANGE_POLARIMETRIC_CAMERA_AUTO_GAIN_RANGE	Behaviour
CHANGE_POLARIMETRIC_CAMERA_EXPOSURE_TIME	Behaviour
ENABLE_POLARIMETRIC_CAMERA_AUTO_EXPOSURE_TIME	Behaviour
CHANGE_POLARIMETRIC_CAMERA_AUTO_EXPOSURE_TIME_RANGE	Behaviour

Table 18. API functions libL3Cam_thermal.h

FUNCTION NAME	CATEGORY
CHANGE_THERMAL_CAMERA_COLORMAP	Behaviour
ENABLE_THERMAL_CAMERA_TEMPERATURE_FILTER	Behaviour
CHANGE_THERMAL_CAMERA_TEMPERATURE_FILTER	Behaviour
CHANGE_THERMAL_CAMERA_PROCESSING_PIPELINE	Behaviour
ENABLE_THERMAL_CAMERA_TEMPERATURE_DATA_UDP	Behaviour

Version:	1.8		Page:	48/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

3.5 API FUNCTIONS DESCRIPTION

registerErrorCallback

```
void getBeamErrorDescription(l3camErrorCallback callback);
```

Description

Function to register a static function where the errors of the L3CAM device will be notified.

Parameters

callback	Static function where the error will be notified
----------	--

Example

```
static void errorNotification(int const *error){
    int ierr = *error;
    char *desc = NULL;

    desc = getBeamErrorDescription(ierr);
    printf("Error notification received %d - %s\n", error, desc);
}

void main()
{
    ...
    registerErrorCallback(errorNotification);

    INITIALIZE(NULL, NULL);
    ...
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	49/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

getBeamErrorDescription

```
const char * getBeamErrorDescription(int error_code);
```

Description

Returns the description of the error code returned by any of the library functions.

Parameters

error_code	Integer with the returned error code.
------------	---------------------------------------

Returns

Char pointer with the description of the error returned by the library.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;

error = INITIALIZE();
desc = getBeamErrorDescription(error);
printf("Initialize response %d - %s\n", error, desc);
```

Version:	1.8	 BEAM\GINE	Page:	50/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_VERSION

```
const char * GET_VERSION();
```

Description

Returns the version of the library.

Returns

Char pointer with the version of the library.

Example

```
char *version = NULL;
version = GET_VERSION();
printf("Library version %s\n", version);
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	51/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

INITIALIZE

```
int INITIALIZE(char *local_ip_address, char *device_ip_address);
```

Description

This function initializes the library. Internally starts the TCP port for communications with an L3CAM device, and starts the discovery thread to find L3CAM devices in the network.

To avoid the broadcast discovery process, the user can specify a known host IP address and the L3CAM device IP address to make direct connection.

Both parameters must be NULL, or char pointer, if just one is NULL, the function will return error.

Parameters

local_ip_address	The host IP Address where the L3CAM is connected
device_ip_address	The IP Address of the L3CAM device.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;

error = INITIALIZE(NULL, NULL);
desc = getBeamErrorDescription(error);
printf("Initialize response %d - %s\n", error, desc);
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	52/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

TERMINATE

```
int TERMINATE(l3cam device);
```

Description

This function closes the library communications. Internally stops the TCP communications with an L3CAM device.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;

error = TERMINATE(devices[0]);
desc = getBeamErrorDescription(error);
printf("Terminate response %d - %s\n", error, desc);
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	53/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

FIND_DEVICES

```
int FIND_DEVICES (l3cam devices[], int *num_devices);
```

Description

Call this function to get the available devices found in the host network, the current version of the library only allows communications with a single L3CAM device, so the l3cam array should be initialized with one item as described in the example.

Parameters

devices []	Array to store the available devices in the network with l3cam structure.
num_devices	Pointer to integer where the number of devices found is returned.

Example

```
int num_devices = 0;
l3cam devices[1];
char *desc = NULL;

int32_t error = FIND_DEVICES(devices, &num_devices);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get devices response %d - %s\n", error, desc);
}

if(num_devices > 0){
    printf("ip address of device %s\n", devices[0].ip_address);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	54/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

GET_LOCAL_SERVER_ADDRESS

```
const char * GET_LOCAL_SERVER_ADDRESS (l3cam device);
```

Description

Call this function to get the local address of the NIC where the L3CAM has been connected.

Parameters

Device	Structure of an available device to execute the function.
--------	---

Returns

Char pointer with the IP address of the NIC where L3CAM device is connected.

Example

```
char *local_ip_address = NULL;

local_ip_address = GET_LOCAL_SERVER_ADDRESS(devices[0]);

printf("Local address: %s\n", local_ip_address);
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	55/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_DEVICE_STATUS

```
int GET_DEVICE_STATUS (l3cam device, int32_t *system_status);
```

Description

Call this function to get the current device status. See status definition for more information.

Parameters

device	Structure of an available device to execute the function.
system_status	Pointer to an integer (32 bit) where the status is returned.

Example

```
int32_t system_status = 0;
char *desc = NULL;

int32_t error = GET_DEVICE_STATUS(devices[0], &system_status);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get device status response %d - %s\n", error, desc);
}

printf("System status %d\n", system_status);
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	56/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_SENSORS_AVAILABLE

```
int GET_SENSORS_AVAILABLE (l3cam device, sensor sensors [], int *num_sensors);
```

Description

Call this function to get the available sensors and the information of each sensor in the device. The number of sensors available will vary depending on the L3CAM model.

Parameters

device	Structure of an available device to execute the function.
sensors[]	Sensor struct array to store the sensors data.
num_sensors	Pointer to store the number of sensors available in the device

Example

```
int num_sensors = 0;
sensor av_sensors[6];
char *desc = NULL;

int32_t error = GET_SENSORS_AVAILABLE(devices[0], av_sensors, &num_sensors);


if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get sensors available response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	57/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_STREAMING_PROTOCOL

```
int CHANGE_STREAMING_PROTOCOL (l3cam device, sensor *sensor_id);
```

Description

Call this function to change the streaming protocol for the desired sensor. See 3.2 for more information.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Pointer to the sensor to change the protocol

Example

```
char *desc = NULL;
av_sensors[0].protocol = protocol_gstreamer;

int32_t error = CHANGE_STREAMING_PROTOCOL(devices[0], &av_sensors[0]);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change streaming protocol response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	58/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_RTSP_PIPELINE

```
int GET_RTSP_PIPELINE(l3cam device, sensor sensor_id, char **pipeline);
```

Description

Call this function to get the current RTSP pipeline of a specific sensor.

Parameters

device	Structure of an available device to execute the function.
sensor_id	The sensor to get the RTSP pipeline
pipeline	Pointer to a pointer with the pipeline string

Example

```
char *desc = NULL;
char *rtsp_pipeline = NULL;

int32_t error = GET_RTSP_PIPELINE(devices[0], av_sensors[0], &rtsp_pipeline);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get RTSP pipeline response %d - %s\n", error, desc);
}
else{
    print("%s \n", rtsp_pipeline);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	59/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_RTSP_PIPELINE

```
int CHANGE_RTSP_PIPELINE(l3cam device, sensor sensor_id, char *pipeline);
```

Description

Call this function to change the current RTSP pipeline of the specific sensor.

Parameters

device	Structure of an available device to execute the function.
sensor_id	The sensor to change the RTSP pipeline
pipeline	Pointer with the new pipeline string

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;

error = CHANGE_RTSP_PIPELINE(devices[0], av_sensors[0], "new_pipeline");

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Initialize response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	60/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

GET_NETWORK_CONFIGURATION

```
int GET_NETWORK_CONFIGURATION(l3cam device, char **ip_address, char **netmask,
                             char **gateway);
```

Description

Call this function to get the current network configuration of the L3CAM device.

Parameters

device	Structure of an available device to execute the function.
ip_address	Pointer to pointer where the IP address is returned
netmask	Pointer to pointer where the netmask is returned as string
gateway	Pointer to pointer where the gateway is returned

Example

```
char *desc = NULL;
char *address = NULL;
char *netmask = NULL;
char *gateway = NULL;

int32_t error = L3CAM_OK;
error = GET_NETWORK_CONFIGURATION(devices[0], &address, &netmask, &gateway);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get network configuration response %d - %s\n", error, desc);
}else{
    printf("address %s netmask %s gateway %s\n", address, netmask, gateway);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	61/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_NETWORK_CONFIGURATION

```
int CHANGE_NETWORK_CONFIGURATION(l3cam device, char *ip_address,
                                char *netmask, char *gateway, bool enable_dhcp);
```

Description

Call this function to change the L3CAM device network configuration.

Parameters

device	Structure of an available device to execute the function.
ip_address	Pointer with the new IP address
netmask	Pointer with the new netmask as a string
gateway	Pointer with the new gateway
enable_dhcp	Boolean to enable or disable the DHCP functionality of the L3CAM

Example

```
char *desc = NULL;

/*FIXED IP ADDRESS DISABLES DHCP*/

int32_t error = L3CAM_OK;
error = CHANGE_NETWORK_CONFIGURATION(devices[0],
                                    "192.168.5.15", "255.255.255.0", "0.0.0.0",
                                    false);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change network response %d - %s\n", error, desc);
}

/*DYNAMIC IP ADDRESS ENABLES DHCP*/

int32_t error = L3CAM_OK;
error = CHANGE_NETWORK_CONFIGURATION(devices[0], NULL, NULL, NULL, true);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change network response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	62/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

POWER_OFF_DEVICE

```
void POWER_OFF_DEVICE (l3cam device);
```

Description

Call this function to properly shut down the L3CAM device.

Parameters

Device	Structure of an available device to execute the function.
--------	---

Example

```
POWER_OFF_DEVICE(devices[0]);
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	63/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

START_DEVICE

```
int START_DEVICE (l3cam device);
```

Description

Call this function to initialize the device, it initializes all the internal sensors and the streaming protocols. This call sends the current date-time of the host to the L3CAM to synchronize the date-time of the sensor and get correct timestamp values.

Parameters

device	Structure of an available device to execute the function.
--------	---

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = START_DEVICE(devices[0]);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Start device response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	64/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

STOP_DEVICE

```
int STOP_DEVICE (l3cam device);
```

Description

Call this function to stop the device. It stops the internal sensors and streaming protocols.

Parameters

device	Structure of an available device to execute the function.
--------	---

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = STOP_DEVICE(devices[0]);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Stop device response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	65/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

START_STREAM

```
int START_STREAM (l3cam device);
```

Description

Call this function to start the streaming of the device. Each sensor will stream with the selected streaming protocol. See 3.1 and 3.2 for more information.

Parameters

device	Structure of an available device to execute the function.
--------	---

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = START_STREAM(devices[0]);


if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Start stream response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	66/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

STOP_STREAM

```
int STOP_STREAM (l3cam device);
```

Description

Call this function to stop the sensors from streaming.

Parameters

device	Structure of an available device to execute the function.
--------	---

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = STOP_STREAM(devices[0]);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Stop stream response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	67/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_POINTCLOUD_COLOR

```
int CHANGE_POINTCLOUD_COLOR(l3cam device, int visualization_color);
```

Description

Call this function to change the colour representation of the point cloud. See 3.2 and APPENDIX 9 for more information about point cloud colours.

Parameters

device	Structure of an available device to execute the function.
visualization_color	Integer with the colour of the point cloud.

Example

```
char *desc = NULL;
int32_t error = CHANGE_POINTCLOUD_COLOR(devices[0], RAINBOW_Z);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change point cloud color response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	68/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_POINTCLOUD_COLOR_RANGE

```
int CHANGE_POINTCLOUD_COLOR_RANGE(l3cam device, int min_value, int max_value);
```

Description

Call this function to change the colour range representation of the point cloud. See 3.2 and APPENDIX 9 for more information about point cloud colours.

Parameters

device	Structure of an available device to execute the function.
min_value	Minimum value for the colour representation.
max_value	Maximum value for the colour representation.

Example

```
char *desc = NULL;
int32_t error = CHANGE_POINTCLOUD_COLOR_RANGE(devices[0], 1500, 50000);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change pointcloud color range response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	69/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_DISTANCE_RANGE

```
int CHANGE_DISTANCE_RANGE(l3cam device, int min_distance, int max_distance);
```

Description

Call this function to apply a distance filter to the point cloud. The resulting point cloud will show the points delimited between the minimum distance and the maximum distance.

Parameters

device	Structure of an available device to execute the function.
min_distance	Minimum distance of the point cloud, the value is in millimetres (mm)
max_distance	Maximum distance of the point cloud, the value is in millimetres (mm)

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
//!filter the pointcloud between 1.5 meters and 50 meters
error = CHANGE_DISTANCE_RANGE(devices[0], 1500, 50000);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change distance range response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	70/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_DEVICE_TEMPERATURES

```
int CHANGE_DEVICE_TEMPERATURES(l3cam device, int32_t *device_temperatures);
```

Description

Call this function to get the temperature information of the available sensors in the L3CAM device. The values returned have to be divided by 1000.0 in order to have the real value with 3 decimal numbers.

The temperature sensors available are given in an int32_t pointer in the following order:

Table 19. Device temperatures pointer content

Pointer index	Value
0	Jetson TX2 BCPU
1	Jetson TX2 MCPU
2	Jetson TX2 MGPU
3	Jetson TX2 PLL
4	Jetson TX2 BOARD
5	Jetson TX2 DIODE
6	Jetson TX2 PMIC
7	Jetson TX2 FAN
8	Beamagine Interface board temperature
9	Allied 508c temperature
10	Allied 319c temperature

Parameters

device	Structure of an available device to execute the function.
device_temperatures	Integer pointer where temperatures are stored in Celsius.



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	71/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
int32_t *temperatures = (int32_t*)malloc(sizeof(int32_t)*11);

error = GET_DEVICE_TEMPERATURES(devices[0], temperatures);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Device temperatures response %d - %s\n", error, desc);
}else{
    printf("Jetson TX2 BCPU temperature %f °C\n", temperatures[0]/1000.0);
    printf("Interface temperature %f °C\n", temperatures[8]/1000.0);
    printf("Allied 508 temperature %f °C\n", temperatures[9]/1000.0);
    printf("Allied 319 temperature %f °C\n", temperatures[10]/1000.0);
}

free(temperatures);
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	72/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

SET_BIAS_LONG_RANGE

```
int SET_BIAS_LONG_RANGE(l3cam device);
```

Description

Call this function to enable the L3CAM maximum range for the autogain algorithm. This function is useful when using the L3CAM in outdoors and day conditions.

Parameters

device	Structure of an available device to execute the function.
--------	---

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = SET_BIAS_LONG_RANGE(devices[0]);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Set bias long range response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	73/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

SET_BIAS_SHORT_RANGE

```
int SET_BIAS_SHORT_RANGE(l3cam device);
```

Description

Call this function to enable the L3CAM minimum range for the autogain algorithm. This function is useful when using the L3CAM in indoors or outdoors night conditions.

Parameters

device	Structure of an available device to execute the function.
--------	---

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = SET_BIAS_SHORT_RANGE(devices[0]);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Set bias short range response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	74/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

ENABLE_AUTO_BIAS

```
void ENABLE_AUTO_BIAS(l3cam device, bool enabled);
```

Description

Enables or disables the auto bias algorithm in the L3CAM device.

Parameters

device	Structure of an available device to execute the function.
enabled	Boolean to enable or disable the algorithm

Example

```
ENABLE_AUTO_BIAS(devices[0], true);
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	75/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_BIAS_VALUE

```
void CHANGE_BIAS_VALUE(l3cam device, uint8_t index, int32_t bias);
```

Description

Manually modify the bias of the specified module of the L3CAM device. This function only works when auto bias has been disabled. Each of the L3CAM LiDAR modules has an independent bias value.

Parameters

device	Structure of an available device to execute the function.
index	Integer to indicate if the bias to change is in module left or right. Set this value to 1 to change right bias value. Set this value to 2 to change left bias value.
bias	The bias value to send. The range is 600 mV – 3500 mV

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;

//!Change bias for right module
CHANGE_BIAS_VALUE(devices[0], 1, 1850);
//!Change bias for left module
CHANGE_BIAS_VALUE(devices[0], 2, 1970);
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	76/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

SET_POLARIMETRIC_CAMERA_DEFAULT_SETTINGS

```
int SET_POLARIMETRIC_CAMERA_DEFAULT_SETTINGS(l3cam device);
```

Description

Call this function to change all the polarimetric camera parameters to the system defaults.

Parameters

device	Structure of an available device to execute the function.
--------	---

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = SET_POLARIMETRIC_CAMERA_DEFAULT_SETTINGS(devices[0]);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Set polarimetric camera default parameters response %d - %s\n",
        error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	77/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_POLARIMETRIC_CAMERA_BRIGHTNESS

```
int CHANGE_POLARIMETRIC_CAMERA_BRIGHTNESS(l3cam device, int brightness);
```

Description

Call this function to change the camera brightness.

Parameters

device	Structure of an available device to execute the function.
brightness	Integer with the new brightness value. The range value is 0 - 255

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_POLARIMETRIC_CAMERA_BRIGHTNESS(devices[0], 210);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change polarimetric brightness response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	78/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_POLARIMETRIC_CAMERA_BLACK_LEVEL

```
int CHANGE_POLARIMETRIC_CAMERA_BLACK_LEVEL(l3cam device, float black_level);
```

Description

Call this function to change the black level of the polarimetric camera.

Parameters

device	Structure of an available device to execute the function.
black_level	Float with the new black level value. The range value is 0 – 12.5

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_POLARIMETRIC_BLACK_LEVEL(devices[0], 8.5);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change polarimetric black level response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	79/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

ENABLE_POLARIMETRIC_CAMERA_AUTO_GAIN

```
int ENABLE_POLARIMETRIC_CAMERA_AUTO_GAIN(l3cam device, bool enabled);
```

Description

Call this function to enable or disable the auto-gain feature of the polarimetric camera.

Parameters

device	Structure of an available device to execute the function.
enabled	Boolean to enable/disable the auto gain feature. true to enable auto gain and false to disable it.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = ENABLE_POLARIMETRIC_CAMERA_AUTO_GAIN(devices[0], true);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Enable polarimetric auto gain response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	80/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_POLARIMETRIC_CAMERA_AUTO_GAIN_RANGE

```
int CHANAGE_POLARIMETRIC_CAMERA_AUTO_GAIN_RANGE(l3cam device, float min_gain,
                                                float max_gain);
```

Description

Call this function to change the range of the polarimetric camera auto-gain feature. When the auto-gain feature is enabled, the camera will adjust it using the range of the specified values.

Parameters

device	Structure of an available device to execute the function.
min_gain	Float with the minimum gain value for the auto-gain feature. The range is 0 – 48.0 dB
max_gain	Float with the maximum gain value for the auto-gain feature. The range is 0 – 48.0 dB

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_POLARIMETRIC_CAMERA_AUTO_GAIN_RANGE(devices[0], 20.5, 35.0);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change polarimetric auto gain range response %d - %s\n", error,
        desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	81/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_POLARIMETRIC_CAMERA_GAIN

```
int CHANGE_POLARIMETRIC_CAMERA_GAIN(l3cam device, float gain);
```

Description

Call this function to manually set the gain of the polarimetric camera. The auto gain feature must be disabled to use this function

Parameters

device	Structure of an available device to execute the function.
gain	The value of the gain. The range is 0 – 48.0 dB

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
//! Disable the polarimetric autogain
error = ENABLE_POLARIMETRIC_CAMERA_AUTO_GAIN(devices[0], false)

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Disable polarimetric auto gain response %d - %s\n", error, desc);
}
else{
    error = CHANGE_POLARIMETRIC_CAMERA_GAIN(devices[0], 30.0);
    if(error != L3CAM_OK){
        desc = getBeamErrorDescription(error);
        printf("Change polarimetric camera gain response %d - %s\n", error, desc);
    }
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	82/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

ENABLE_POLARIMETRIC_CAMERA_AUTO_EXPOSURE_TIME

```
int ENABLE_POLARIMETRIC_CAMERA_AUTO_EXPOSURE_TIME(l3cam device, bool enabled);
```

Description

Call this function to enable or disable the auto-exposure time feature of the polarimetric camera.

Parameters

device	Structure of an available device to execute the function.
enabled	Boolean to enable/disable the auto-exposure time feature. True to enable auto exposure time and false to disable it.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;

error = ENABLE_POLARIMETRIC_CAMERA_AUTO_EXPOSURE_TIME(devices[0], true);
if(error != L3CAM_OK ){
    desc = getBeamErrorDescription(error);
    printf("Enable polarimetric auto exposure time response %d - %s\n",
        error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	83/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_POLARIMETRIC_CAMERA_AUTO_EXPOSURE_TIME_RANGE

```
int CHANGE_POLARIMETRIC_CAMERA_AUTO_EXPOSURE_TIME_RANGE(l3cam device,
                                                         float min_exposure, float min_exposure);
```

Description

Call this function to change the auto-exposure time values range. When the auto-exposure time feature is enabled, the value will vary in the range of the specified values.

Parameters

device	Structure of an available device to execute the function.
min_exposure	Float with the minimum exposure time value for the auto exposure time feature. The range is 33.456us to 1000000.0us
max_exposure	Float with the maximum exposure time value for the auto exposure time feature. The range is 33.456us to 1000000.0us

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_POLARIMETRIC_CAMERA_AUTO_EXPOSURE_TIME_RANGE(devices[0], 100.0,
                                                             5000.0);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Polarimetric auto exposure time range response %d - %s\n",
           error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	84/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_POLARIMETRIC_CAMERA_EXPOSURE_TIME

```
int CHANGE_POLARIMETRIC_CAMERA_EXPOSURE_TIME(l3cam device,
                                              float exposure_time);
```

Description

Call this function to change the polarimetric camera exposure time. The auto-exposure time must be disabled to be able to use this function.

Parameters

device	Structure of an available device to execute the function.
exposure_time	Float with the exposure time value. The range is 33.456us to 1000000.0us

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
///! Disable the polarimetric auto exposure time
error = ENABLE_POLARIMETRIC_CAMERA_AUTO_EXPOSURE_TIME(devices[0], false)

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Disable polarimetric auto exposure time response %d - %s\n",
           error, desc);
}
else{
    error = CHANGE_POLARIMETRIC_CAMERA_EXPOSURE_TIME(devices[0], 3000.0);
    if(error != L3CAM_OK){
        desc = getBeamErrorDescription(error);
        printf("Change polarimetric camera exposure time response %d - %s\n",
               error, desc);
    }
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	85/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

SET_RGB_CAMERA_DEFAULT_SETTINGS

```
int SET_RB_CAMERA_DEFAULT_SETTINGS(l3cam device);
```

Description

Call this function to change all the RGB camera parameters to the system defaults.

Parameters

device	Structure of an available device to execute the function.
--------	---

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = SET_RGB_CAMERA_DEFAULT_SETTINGS(devices[0]);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Set RGB camera default parameters response %d - %s\n",
        error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	86/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_RGB_CAMERA_BRIGHTNESS

```
int CHANGE_RGB_CAMERA_BRIGHTNESS(l3cam device, int brightness);
```

Description

Call this function to change the camera brightness.

Parameters

device	Structure of an available device to execute the function.
brightness	Integer with the new brightness value. The range value is -15 - 15

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_RGB_CAMERA_BRIGHTNESS(devices[0], 10);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change RGB brightness response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	87/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_RGB_CAMERA_CONTRAST

```
int CHANGE_RGB_CAMERA_CONTRAST(l3cam device, int contrast);
```

Description

Call this function to change the contrast of the RGB camera.

Parameters

device	Structure of an available device to execute the function.
contrast	Integer with the new contrast value. The range value is 0 – 30

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_RGB_CAMERA_CONTRAST(devices[0], 5);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change RGB contrast response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	88/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_RGB_CAMERA_SATURATION

```
int CHANGE_RGB_CAMERA_SATURATION(l3cam device, int saturation);
```

Description

Call this function to change the saturation of the RGB camera.

Parameters

device	Structure of an available device to execute the function.
saturation	Integer with the new saturation value. The range value is 0 – 60

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_RGB_CAMERA_SATURATION(devices[0], 10);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change RGB saturation response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	89/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_RGB_CAMERA_SHARPNESS

```
int CHANGE_RGB_CAMERA_SHARPNESS(l3cam device, int sharpness);
```

Description

Call this function to change the sharpness of the RGB camera.

Parameters

device	Structure of an available device to execute the function.
sharpness	Integer with the new sharpness value. The range value is 0 – 127

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;

error = CHANGE_RGB_CAMERA_SHARPNESS(devices[0], 100);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change RGB sharpness response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	90/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_RGB_CAMERA_GAMMA

```
int CHANGE_RGB_CAMERA_GAMMA(l3cam device, int gamma);
```

Description

Call this function to change the gamma of the RGB camera.

Parameters

device	Structure of an available device to execute the function.
gamma	Integer with the new gamma value. The range value is 40 – 500

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;

error = CHANGE_RGB_CAMERA_GAMMA(devices[0], 150);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change RGB gamma response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	91/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_RGB_CAMERA_GAIN

```
int CHANGE_RGB_CAMERA_GAIN(l3cam device, int gain);
```

Description

Call this function to change the gain of the RGB camera.

Parameters

device	Structure of an available device to execute the function.
gain	Integer with the new gain value. The range value is 0 – 63

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_RGB_CAMERA_GAIN(devices[0], 20);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change RGB gain response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	92/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_RGB_CAMERA_WHITE_BALANCE

```
int CHANGE_RGB_CAMERA_WHITE_BALANCE(l3cam device, int white_balance);
```

Description

Call this function to change the white balance of the RGB camera.

Parameters

device	Structure of an available device to execute the function.
white_balance	Integer with the new white balance value. The range value is 1000 – 10000

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_RGB_CAMERA_WHITE_BALANCE(devices[0], 3000);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change RGB white balance response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	93/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_RGB_CAMERA_EXPOSURE_TIME

```
int CHANGE_RGB_CAMERA_EXPOSURE_TIME(l3cam device, int exposure_time);
```

Description

Call this function to change the exposure time of the RGB camera.

Parameters

device	Structure of an available device to execute the function.
exposure_time	Integer with the new exposure time value. The range value is 1 – 10000

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_RGB_CAMERA_EXPOSURE_TIME(devices[0], 3000);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change RGB exposure time response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	94/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_RGB_CAMERA_RESOLUTION

```
int CHANGE_RGB_CAMERA_RESOLUTION(l3cam device, econResolutions resolution);
```

Description

Call this function to change the resolution of the RGB image.

Parameters

device	Structure of an available device to execute the function.
resolution	Enum that indicates the desired resolution for the RGB image. The available resolutions are 640x480px; 1280x720px; 1920x1080px

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_RGB_CAMERA_RESOLUTION(devices[0], reso_1280_720);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change RGB image resolution response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	95/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_RGB_CAMERA_FRAMERATE

```
int CHANGE_RGB_CAMERA_FRAMERATE(l3cam device, int framerate);
```

Description

Call this function to change the framerate of the RGB camera.

Parameters

device	Structure of an available device to execute the function.
framerate	Integer with the new framerate value. The range value is 1 – 16 fps

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_RGB_CAMERA_FRAMERATE(devices[0], 10);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change RGB framerate response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	96/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

ENABLE_RGB_CAMERA_AUTO_WHITE_BALANCE

```
int ENABLE_RGB_CAMERA_AUTO_WHITE_BALANCE (l3cam device, bool enabled);
```

Description

Call this function to enable or disable the auto white balance feature of the RGB camera.

Parameters

device	Structure of an available device to execute the function.
enabled	Boolean to enable/disable the auto white balance feature.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = ENABLE_RGB_CAMERA_AUTO_WHITE_BALANCE(devices[0], true);


if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Enable RGB camera auto white balance response %d - %s\n", error,
        desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	97/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

ENABLE_RGB_CAMERA_AUTO_EXPOSURE_TIME

```
int ENABLE_RGB_CAMERA_AUTO_EXPOSURE_TIME (l3cam device, bool enabled);
```

Description

Call this function to enable or disable the auto exposure time feature of the RGB camera.

Parameters

device	Structure of an available device to execute the function.
enabled	Boolean to enable/disable the auto exposure time feature.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = ENABLE_RGB_CAMERA_AUTO_EXPOSURE_TIME(devices[0], true);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Enable RGB camera auto exp.time response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	98/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_THERMAL_CAMERA_COLORMAP

```
int CHANGE_THERMAL_CAMERA_COLORMAP(l3cam device, int32_t colormap);
```

Description

Call this function to change the current colormap of the thermal image.

Parameters

device	Structure of an available device to execute the function.
colormap	integer that indicates the current colormap for the thermal image.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_THERMAL_CAMERA_COLORMAP(devices[0], (int32_t)new_thermal_IRON);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Thermal colormap response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	99/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_THERMAL_CAMERA_TEMPERATURE_FILTER

```
int CHANGE_THERMAL_CAMERA_TEMPERATURE_FILTER(l3cam device,
                                              float min_temperature,
                                              float max_temperature);
```

Description

Call this function to change the thermal values for the filter of the thermal camera.

Parameters

device	Structure of an available device to execute the function.
min_temperature	float with the new minimum temperature value. The range value is -40 – 200 °C
max_temperature	float with the new maximum temperature value. The range value is -40 – 200 °C

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_THERMAL_CAMERA_TEMPERATURE_FILTER(devices[0], 20.5, 35.5);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Thermal filter values response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	100/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

ENABLE_THERMAL_CAMERA_TEMPERATURE_FILTER

```
int ENABLE_THERMAL_CAMERA_TEMPERATURE_FILTER (l3cam device, bool enabled);
```

Description

Call this function to enable or disable the temperature filter feature of the thermal camera.

Parameters

device	Structure of an available device to execute the function.
enabled	Boolean to enable/disable the filter feature.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = ENABLE_THERMAL_CAMERA_TEMPERATURE_FILTER(devices[0], true);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Enable thermal camera filter response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	101/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_THERMAL_CAMERA_PROCESSING_PIPELINE

```
int CHANGE_THERMAL_CAMERA_PROCESSING_PIPELINE (l3cam device,
                                              int32_t pipeline);
```

Description

Call this function to change the thermal imaging pipeline mode.

Parameters

device	Structure of an available device to execute the function.
pipeline	Integer that indicates the selected pipeline mode.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_THERMAL_CAMERA_PROCSERING_PIPELINE(devices[0],
                                                  (int32_t)thermal_SEEK);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change thermal camera pipeline response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	102/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

ENABLE_THERMAL_CAMERA_TEMPERATURE_DATA_UDP

```
int ENABLE_THERMAL_CAMERA_TEMPERATURE_DATA_UDP (l3cam device, bool enabled);
```

Description

Call this function to enable or disable the streaming of the temperatures data over UDP.

Parameters

device	Structure of an available device to execute the function.
enabled	Boolean to enable/disable the streaming of the temperatures.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = ENABLE_THERMAL_CAMERA_TEMPERATURE_DATA_UDP(devices[0], true);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Enable temperature over UDP response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	103/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_ALLIED_CAMERA_BLACK_LEVEL

```
int CHANGE_ALLIED_CAMERA_BLACK_LEVEL (l3cam device, sensor sensor_id,
                                       float black_level);
```

Description

Call this function to change the black level of any of the allied cameras. This value only can be modified while the camera is not streaming.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
black_level	Float with the new black level value. The range value is 0 – 4096

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_BLACK_LEVEL(devices[0], av_sensors[2], 20.0);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied Black Level response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	104/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_ALLIED_CAMERA_EXPOSURE_TIME_US

```
int CHANGE_ALLIED_CAMERA_EXPOSURE_TIME_US (l3cam device, sensor sensor_id,
                                             float exposure_time);
```

Description

Call this function to change the exposure time of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
exposure_time	Float with the new exposure time value. The range value is 63.03 – 1e+07

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_EXPOSURE_TIME_US(devices[0], av_sensors[2],
                                              10000.0);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied Exposure Time response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	105/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

ENABLE_ALLIED_CAMERA_AUTO_EXPOSURE_TIME

```
int ENABLE_ALLIED_CAMERA_AUTO_EXPOSURE_TIME (l3cam device, sensor sensor_id,
                                              bool enable);
```

Description

Call this function to enable/disable the auto exposure time feature of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
enable	Boolean that enables/disables the auto exposure time feature. True to enable, false to disable.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = ENABLE_ALLIED_CAMERA_AUTO_EXPOSURE_TIME(devices[0], av_sensors[2],
                                              true);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Enable Allied Auto Exposure Time response %d - %s\n", error,
          desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	106/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_ALLIED_CAMERA_AUTO_EXPOSURE_TIME_RANGE

```
int CHANGE_ALLIED_CAMERA_AUTO_EXPOSURE_TIME_RANGE(l3cam device,
                                                    sensor sensor_id,
                                                    float min, float max);
```

Description

Call this function to change the auto exposure time range of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
min	Float with the minimum exposure time value. The range value is 63.03 – 0.9e+06
max	Float with the maximum exposure time value. The range value is 87.596 – 1e+07

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_AUTO_EXPOSURE_TIME_RANGE(devices[0],
                                                        av_sensors[2],
                                                        1000.0, 10000.0);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied Auto Exposure Time Range response %d - %s\n", error,
           desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	107/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_ALLIED_CAMERA_GAIN

```
int CHANGE_ALLIED_CAMERA_GAIN (l3cam device, sensor sensor_id, float gain);
```

Description

Call this function to change the gain of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
gain	Float with the new gain value. The range value is 0 – 48

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_GAIN(devices[0], av_sensors[2], 10.0);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied Gain response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	108/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

ENABLE_ALLIED_CAMERA_AUTO_GAIN

```
int ENABLE_ALLIED_CAMERA_AUTO_GAIN(l3cam device, sensor sensor_id,
                                   bool enable);
```

Description

Call this function to enable/disable the auto gain feature of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
enable	Boolean that enables/disables the auto gain feature. True to enable, false to disable.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = ENABLE_ALLIED_CAMERA_AUTO_GAIN(devices[0], av_sensors[2], true);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Enable Allied Auto Gain response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	109/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_ALLIED_CAMERA_AUTO_GAIN_RANGE

```
int CHANGE_ALLIED_CAMERA_AUTO_EXPOSURE_TIME_RANGE(l3cam device,
                                                    sensor sensor_id,
                                                    float min, float max);
```

Description

Call this function to change the auto gain range of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
min	Float with the minimum gain value. The range value is 0 – 48
max	Float with the maximum gain value. The range value is 0 – 48

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_AUTO_GAIN_RANGE(devices[0], av_sensors[2],
                                             100.0, 4000.0);


if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied Auto Gain Range response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	110/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_ALLIED_CAMERA_GAMMA

```
int CHANGE_ALLIED_CAMERA_GAMMA (l3cam device, sensor sensor_id, float gamma);
```

Description

Call this function to change the gamma of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
gamma	Float with the new gamma value. The range value is 0.4 – 2.4

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_GAMMA(devices[0], av_sensors[2], 1.4);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied Gamma response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	111/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_ALLIED_CAMERA_SATURATION

```
int CHANGE_ALLIED_CAMERA_SATURATION (l3cam device, sensor sensor_id,
                                     float saturation);
```

Description

Call this function to change the saturation of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
saturation	Float with the new saturation value. The range value is 0 – 1.0

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_SATURATION(devices[0], av_sensors[2], 0.5);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied Saturation response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	112/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_ALLIED_CAMERA_SHARPNESS

```
int CHANGE_ALLIED_CAMERA_SHARPNESS (l3cam device, sensor sensor_id,
                                     int sharpness);
```

Description

Call this function to change the sharpness of any of the allied cameras. This value only can be modified while the camera is not streaming.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
sharpness	Integer with the new sharpness value. The range value is -12 – 12

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_SHARPNESS(devices[0], av_sensors[2], 10);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied Sharpness response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	113/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_ALLIED_CAMERA_HUE

```
int CHANGE_ALLIED_CAMERA_HUE (l3cam device, sensor sensor_id, float hue);
```

Description

Call this function to change the hue of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
hue	Float with the new hue value. The range value is -40 – 40

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_HUE(devices[0], av_sensors[2], 20);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied HUE response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	114/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_ALLIED_CAMERA_INTENSITY_AUTO_PRECEDENCE

```
int CHANGE_ALLIED_CAMERA_INTENSITY_AUTO_PRECEDENCE (l3cam device,
                                                    sensor sensor_id,
                                                    int mode);
```

Description

Call this function to change the intensity auto precedence of any of the allied cameras. This value only can be modified while the camera is not streaming.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
mode	Value for the intensity auto precedence mode, see APPENDIX 9 for the available values. The possible values are allied_auto_precedence_minimize_noise and allied_auto_precedence_minimize_blur.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_INTENSITY_AUTO_PRECEDENCE(devices[0],
                                                        av_sensors[2],
                                                        allied_auto_precedence_minimize_noise);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied Camera intensity auto precedence response %d -%s\n",
           error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	115/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

ENABLE_ALLIED_CAMERA_AUTO_WHITE_BALANCE

```
int ENABLE_ALLIED_CAMERA_AUTO_WHITE_BALANCE(l3cam device, sensor sensor_id,
                                             bool enable);
```

Description

Call this function to enable/disable the auto white balance feature of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
enable	Boolean that enables/disables the auto white balance feature. True to enable, false to disable.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = ENABLE_ALLIED_CAMERA_AUTO_WHITE_BALANCE(devices[0], av_sensors[2],
                                                true);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Enable Allied Auto White Balance response %d -%s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	116/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_ALLIED_CAMERA_BALANCE_RATIO_SELECTOR

```
int CHANGE_ALLIED_CAMERA_BALANCE_RATIO_SELECTOR (l3cam device,
                                                  sensor sensor_id, int mode);
```

Description

Call this function to change the white balance ratio selector of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
mode	Value for the balance ratio selector mode, see APPENDIX 9 for the available values. The possible values are allied_balance_ratio_selector_red and allied_balance_ratio_selector_blue.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_BALANCE_RATIO_SELECTOR(devices[0], av_sensors[2],
                                                    allied_balance_ratio_selector_red);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied Camera balance ratio selector response %d -%s\n",
           error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	117/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_ALLIED_CAMERA_BALANCE_RATIO

```
int CHANGE_ALLIED_CAMERA_BALANCE_RATIO (l3cam device, sensor sensor_id,
                                         float balance_ratio);
```

Description

Call this function to change the balance ratio of the colour specified in the balance ratio selector for the white balance algorithm of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
balance_ratio	Float with the new balance ratio value. The range value is 0 – 8

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_BALANCE_RATIO(devices[0], av_sensors[2], 6.0);


if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied balance ratio response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	118/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_ALLIED_CAMERA_BALANCE_WHITE_AUTO_RATE

```
int CHANGE_ALLIED_CAMERA_BALANCE_WHITE_AUTO_RATE (l3cam device,
                                                    sensor sensor_id,
                                                    float balance_white_auto_rate);
```

Description

Call this function to change the rate at which the auto function changes the white balance of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
balance_white_auto_rate	Float with the new balance white auto ratio value. The range value is 1 – 100

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_BALANCE_WHITE_AUTO_RATE(devices[0],
                                                    av_sensors[2], 80.0);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied Balance white auto rate response %d - %s\n",
          error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	119/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_ALLIED_CAMERA_BALANCE_WHITE_AUTO_TOLERANCE

```
int CHANGE_ALLIED_CAMERA_BALANCE_WHITE_AUTO_TOLERANCE (l3cam device,
                                                         sensor sensor_id,
                                                         float balance_white_auto_tolerance);
```

Description

Call this function to change the tolerance from the ideal white balance value in which the auto white balance algorithm will not react in any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
balance_white_auto_tolerance	Float with the new balance white auto ratio value. The range value is 1 – 50

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_BALANCE_WHITE_AUTO_TOLERANCE(devices[0],
                                                           av_sensors[2],
                                                           10.0);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied Balance white auto tolerance response %d - %s\n",
           error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	120/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_ALLIED_CAMERA_AUTO_MODE_REGION

```
int CHANGE_ALLIED_CAMERA_AUTO_MODE_REGION (l3cam device, sensor sensor_id,
                                             int height, int width );
```

Description

Call this function to change the window used to measure values for auto functions/algorithms in any allied camera. This value only can be modified while the camera is not streaming and the values for height and width must be even.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
height	Integer value for the height of the rectangle. The range for each camera is different. Allied 508c 0 – 1028 Allied 319c 0 – 1554
width	Integer value for the width of the rectangle. The range for each camera is different. Allied 508c 0 – 1232 Allied 319c 0 – 2064

Example

```
/*
In this example the wide 508 camera is av_sensors[2] and the narrow 319 camera
is in av_sensors[3]
*/
char *desc = NULL;
int32_t error = L3CAM_OK;

error = CHANGE_ALLIED_CAMERA_AUTO_MODE_REGION(devices[0], av_sensors[2], 1028,
                                              1232);
if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied wide Camera auto mode region response %d -%s\n",
          error, desc);
}

error = CHANGE_ALLIED_CAMERA_AUTO_MODE_REGION(devices[0], av_sensors[3], 1554,
                                              2064);
if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied narrow Camera auto mode region response %d -%s\n",
          error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	121/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_ALLIED_CAMERA_INTENSITY_CONTROLLER_REGION

```
int CHANGE_ALLIED_CAMERA_INTENSITY_CONTROLLER_REGION (l3cam device,
                                                    sensor sensor_id,
                                                    int mode);
```

Description

Call this function to change between the full image or the subregion selected with CHANGE_ALLIED_CAMERA_AUTO_MODE_REGION function where the intensity controller operates in any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
mode	Value for the intensity controller region selector mode, see APPENDIX 9 for the available values. The possible values are allied_controller_region_auto_mode_region and allied_controller_region_full_image.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_INTENSITY_CONTROLLER_REGION(devices[0],
                                                         av_sensors[2],
                                                         allied_controller_region_auto_mode_region);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied Camera intensity controller region response %d-%s\n",
           error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	122/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

CHANGE_ALLIED_CAMERA_INTENSITY_CONTROLLER_TARGET

```
int CHANGE_ALLIED_CAMERA_INTENSITY_CONTROLLER_TARGET (l3cam device,
                                                    sensor sensor_id,
                                                    float intensity_controller_target);
```

Description

Call this function to change the target intensity value for auto intensity control in any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
intensity_controller_target	Float with the new intensity target. The range value is 10 – 90

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_INTENSITY_CONTROLLER_TARGET(devices[0],
                                                         av_sensors[2],
                                                         60.0);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied intensity controller target response %d - %s\n",
           error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	123/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_ALLIED_CAMERA_MAX_DRIVER_BUFFERS_COUNT

```
int CHANGE_ALLIED_CAMERA_MAX_DRIVER_BUFFERS_COUNT (l3cam device,
                                                    sensor sensor_id,
                                                    int max_driver_buffers_count);
```

Description

Call this function to change the maximum numbers of driver buffers used by the acquisition engine inside the Jetson TX2 for any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is modified in the u-508 or in the u-319 allied camera.
max_driver_buffers_count	Integer with the max driver buffers. The range value is 1 – 4096

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
error = CHANGE_ALLIED_CAMERA_MAX_DRIVER_BUFFERS_COUNT(devices[0],
                                                       av_sensors[2], 65);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Change Allied Camera max driver buffers count response %d -%s\n",
           error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	124/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_BLACK_LEVEL

```
int GE_ALLIED_CAMERA_BLACK_LEVEL (l3cam device, sensor sensor_id,
                                   float *black_level);
```

Description

Call this function to get the current black level of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
black_level	Float pointer where the black level value is returned

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
float black_level = -1.0;

error = GET_ALLIED_CAMERA_BLACK_LEVEL(devices[0], av_sensors[2], &black_level);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Black Level response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	125/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_EXPOSURE_TIME_US

```
int GET_ALLIED_CAMERA_EXPOSURE_TIME_US (l3cam device, sensor sensor_id,
                                         float *exposure_time);
```

Description

Call this function to get the exposure time value of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
exposure_time	Float pointer where the exposure time value is returned.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
float exposure = -1.0;
error = GET_ALLIED_CAMERA_EXPOSURE_TIME_US(devices[0], av_sensors[2],
                                           &exposure);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Exposure Time response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	126/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_AUTO_EXPOSURE_TIME

```
int GET_ALLIED_CAMERA_AUTO_EXPOSURE_TIME (l3cam device, sensor sensor_id,
                                           bool *enable);
```

Description

Call this function to know if the auto exposure time feature of any of the allied cameras is enabled or disabled.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
enable	Boolean pointer that indicates if the auto exposure time is enabled.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
bool enabled = false;
error = GET_ALLIED_CAMERA_AUTO_EXPOSURE_TIME(devices[0], av_sensors[2],
                                             &enabled);
if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Auto Exposure Time response %d - %s\n", error,
          desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	127/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

GET_ALLIED_CAMERA_AUTO_EXPOSURE_TIME_RANGE

```
int GET_ALLIED_CAMERA_AUTO_EXPOSURE_TIME_RANGE(l3cam device, sensor sensor_id,
                                                float *min, float *max);
```

Description

Call this function to get the auto exposure time range values of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
min	Float pointer with the minimum exposure time value.
max	Float pointer with the maximum exposure time value.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
float min = -1.0;
float max = -1.0;
error = GET_ALLIED_CAMERA_AUTO_EXPOSURE_TIME_RANGE(devices[0], av_sensors[2],
                                                    &min, &max);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Auto Exposure Time Range response %d - %s\n", error,
          desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	128/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_GAIN

```
int GET_ALLIED_CAMERA_GAIN (l3cam device, sensor sensor_id, float *gain);
```

Description

Call this function to get the gain value of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
gain	Float pointer with the current gain value.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
float gain = -1.0;
error = GET_ALLIED_CAMERA_GAIN(devices[0], av_sensors[2], &gain);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Gain response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	129/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_AUTO_GAIN

```
int GET_ALLIED_CAMERA_AUTO_GAIN(l3cam device, sensor sensor_id, bool *enable);
```

Description

Call this function to know if the auto gain feature of any of the allied cameras is enabled.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
enable	Boolean pointer that indicates if the auto gain is enabled or disabled.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
bool enabled = false;
error = GET_ALLIED_CAMERA_AUTO_GAIN(devices[0], av_sensors[2], &enabled);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Auto Gain response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	130/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_AUTO_GAIN_RANGE

```
int GET_ALLIED_CAMERA_AUTO_EXPOSURE_TIME_RANGE(l3cam device, sensor sensor_id,
                                                float *min, float *max);
```

Description

Call this function to get the auto gain range values of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
min	Float pointer with the minimum gain value.
max	Float pointer with the maximum gain value.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
float min = -1.0;
float max = -1.0;

error = GET_ALLIED_CAMERA_AUTO_GAIN_RANGE(devices[0], av_sensors[2],
                                           &min, &max);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Auto Gain Range response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	131/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_GAMMA

```
int GET_ALLIED_CAMERA_GAMMA(l3cam device, sensor sensor_id, float *gamma);
```

Description

Call this function to get the gamma value of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
gamma	Float pointer with the gamma value.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
float gamma = -1.0;

error = CHANGE_ALLIED_CAMERA_GAMMA(devices[0], av_sensors[2], &gamma);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Gamma response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	132/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_SATURATION

```
int GET_ALLIED_CAMERA_SATURATION (l3cam device, sensor sensor_id,
                                  float *saturation);
```

Description

Call this function to get the saturation value of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
saturation	Float with the saturation value.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
float saturation = -1.0;

error = GET_ALLIED_CAMERA_SATURATION(devices[0], av_sensors[2], &saturation);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Saturation response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	133/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_SHARPNESS

```
int GET_ALLIED_CAMERA_SHARPNESS (l3cam device, sensor sensor_id,
                                int *sharpness);
```

Description

Call this function to get the sharpness value of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
sharpness	Integer pointer with the sharpness value.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
int *sharpness = -1;
error = GET_ALLIED_CAMERA_SHARPNESS(devices[0], av_sensors[2], &sharpness);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Sharpness response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	134/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_HUE

```
int GET_ALLIED_CAMERA_HUE (l3cam device, sensor sensor_id, float *hue);
```

Description

Call this function to get the hue value of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
hue	Float pointer with the hue value.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
float hue = -1.0;

error = GET_ALLIED_CAMERA_HUE(devices[0], av_sensors[2], &hue);


if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied HUE response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	135/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_INTENSITY_AUTO_PRECEDENCE

```
int GET_ALLIED_CAMERA_INTENSITY_AUTO_PRECEDENCE (l3cam device,
                                                    sensor sensor_id, int *mode);
```

Description

Call this function to get the intensity auto precedence value of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
mode	Pointer for the intensity auto precedence mode.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
int mode = -1;

error = GET_ALLIED_CAMERA_INTENSITY_AUTO_PRECEDENCE(devices[0], av_sensors[2],
                                                    &mode);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Camera intensity auto precedence response %d -%s\n",
           error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	136/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_AUTO_WHITE_BALANCE

```
int GET_ALLIED_CAMERA_AUTO_WHITE_BALANCE(l3cam device, sensor sensor_id,
                                          bool *enable);
```

Description

Call this function to know if the auto white balance feature of any of the allied cameras is enabled or disabled.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
enable	Boolean pointer that indicates if the auto white balance feature status.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
bool enabled = false;

error = GET_ALLIED_CAMERA_AUTO_WHITE_BALANCE(devices[0], av_sensors[2],
                                             &enabled);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Auto White Balance response %d -%s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	137/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_BALANCE_RATIO_SELECTOR

```
int GET_ALLIED_CAMERA_BALANCE_RATIO_SELECTOR (l3cam device, sensor sensor_id,
                                              int *mode);
```

Description

Call this function to get the white balance ratio selector value of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
mode	Integer pointer where the value for the balance ratio selector mode is returned.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
int mode = -1;

error = GET_ALLIED_CAMERA_BALANCE_RATIO_SELECTOR(devices[0], av_sensors[2],
                                                &mode);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Camera balance ratio selector response %d -%s\n",
          error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	138/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_BALANCE_RATIO

```
int GET_ALLIED_CAMERA_BALANCE_RATIO (l3cam device, sensor sensor_id,
                                     float *balance_ratio);
```

Description

Call this function to get the balance ratio value of the colour specified in the balance ratio selector for the white balance algorithm of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
balance_ratio	Float pointer with the balance ratio value.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
float balance_ratio = -1.0;

error = GET_ALLIED_CAMERA_BALANCE_RATIO(devices[0], av_sensors[2],
                                       &balance_ratio);


if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied balance ratio response %d - %s\n", error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	139/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_BALANCE_WHITE_AUTO_RATE

```
int GET_ALLIED_CAMERA_BALANCE_WHITE_AUTO_RATE (l3cam device, sensor sensor_id,
                                                float *balance_white_auto_rate);
```

Description

Call this function to get the rate value at which the auto function changes the white balance of any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
balance_white_auto_rate	Float pointer with the balance white auto ratio value.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
float balance = -1.0;

error = GET_ALLIED_CAMERA_BALANCE_WHITE_AUTO_RATE(devices[0], av_sensors[2],
                                                  &balance);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Balance white auto rate response %d - %s\n",
          error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	140/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

GET_ALLIED_CAMERA_BALANCE_WHITE_AUTO_TOLERANCE

```
int GET_ALLIED_CAMERA_BALANCE_WHITE_AUTO_TOLERANCE (l3cam device,
                                                    sensor sensor_id,
                                                    float *balance_white_auto_tolerance);
```

Description

Call this function to get the tolerance value from the ideal white balance value in which the auto white balance algorithm will not react in any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
balance_white_auto_tolerance	Float pointer with the balance white auto ratio value.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
float balance = -1.0;

error = GET_ALLIED_CAMERA_BALANCE_WHITE_AUTO_TOLERANCE(devices[0],
                                                         av_sensors[2],
                                                         &balance);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Balance white auto tolerance response %d - %s\n",
           error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	141/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

GET_ALLIED_CAMERA_AUTO_MODE_REGION

```
int GET_ALLIED_CAMERA_AUTO_MODE_REGION (l3cam device, sensor sensor_id,
                                         int *height, int *width );
```

Description

Call this function to get the window used to measure values for auto functions/algorithms in any allied camera.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
height	Integer pointer with the value for the height of the rectangle.
width	Integer pointer with the value for the width of the rectangle.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
int height = -1;
int width = -1;

error = GET_ALLIED_CAMERA_AUTO_MODE_REGION(devices[0], av_sensors[2], &height,
                                           &width);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied wide Camera auto mode region response %d -%s\n",
           error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	142/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_INTENSITY_CONTROLLER_REGION

```
int GET_ALLIED_CAMERA_INTENSITY_CONTROLLER_REGION (l3cam device,
                                                    sensor sensor_id,
                                                    int *mode);
```

Description

Call this function to get the subregion of the image where the intensity controller operates in any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or from the u-319 allied camera.
mode	Integer pointer with the intensity controller region selector mode value.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
int mode = -1;

error = GET_ALLIED_CAMERA_INTENSITY_CONTROLLER_REGION(devices[0],
                                                    av_sensors[2], &mode);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Camera intensity controller region response %d-%s\n",
           error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	143/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

GET_ALLIED_CAMERA_INTENSITY_CONTROLLER_TARGET

```
int GET_ALLIED_CAMERA_INTENSITY_CONTROLLER_TARGET (l3cam device,
                                                    sensor sensor_id,
                                                    float *intensity_controller_target);
```

Description

Call this function to get the target intensity value for auto intensity control in any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
intensity_controller_target	Float pointer with the intensity target.

Example

```
char *desc = NULL;
int32_t error = L3CAM_OK;
intt target = -1.0;

error = GET_ALLIED_CAMERA_INTENSITY_CONTROLLER_TARGET(devices[0],
                                                    av_sensors[2],
                                                    &target);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied intensity controller target response %d - %s\n",
           error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	144/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

CHANGE_ALLIED_CAMERA_MAX_DRIVER_BUFFERS_COUNT

```
int GET_ALLIED_CAMERA_MAX_DRIVER_BUFFERS_COUNT(l3cam device,
                                                sensor sensor_id,
                                                int *max_driver_buffers_count);
```

Description

Call this function to get the maximum numbers of driver buffers used by the acquisition engine inside the Jetson TX2 for any of the allied cameras.

Parameters

device	Structure of an available device to execute the function.
sensor_id	Structure that indicates if the parameter is obtained from the u-508 or the u-319 allied camera.
max_driver_buffers_count	Integer pointer with the max driver buffers value.

Example

```
char *desc = NULL;

int32_t error = L3CAM_OK;
int buffers = -1;

error = GET_ALLIED_CAMERA_MAX_DRIVER_BUFFERS_COUNT(devices[0],
                                                    av_sensors[2], &buffers);

if(error != L3CAM_OK){
    desc = getBeamErrorDescription(error);
    printf("Get Allied Camera max driver buffers count response %d -%s\n",
           error, desc);
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	145/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

APPENDIXES

APPENDIX 1 READING A POINT CLOUD

Reading a point cloud requires UDP socket management and an infinite loop reading UDP packets. In the infinite loop, check for 3 different sizes of packets:

- 17 Bytes for header packet

```
if (size_read == 17){ //Header
    memcpy(&m_pointcloud_size, &buffer[1], 4);
    m_buffer_size = sizeof(int32_t) * ((m_pointcloud_size * 5) + 1);
    m_pointcloud_data = (int32_t *)malloc(m_buffer_size);

    memcpy(&m_pointcloud_data[0], &m_pointcloud_size, sizeof(int32_t));

    memcpy(&m_timestamp, &buffer[13], sizeof(uint32_t));
    m_is_reading_pointcloud = true;
    points_received = 0;
    pointcloud_index = 1;
}
```

- 8000 Bytes or less for data packets

```
if (size_read > 0 && m_is_reading_pointcloud) { // Data

    int32_t points = 0;
    memcpy(&points, &buffer[0], 4);
    int data_size = sizeof(int32_t) * points * 5;

    memcpy(&m_pointcloud_data[pointcloud_index], &buffer[4], data_size);


    pointcloud_index += (points * 5);
    points_received += points;
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	146/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

- 1 Byte for footer packet

```

if(size_read == 1 && m_is_reading_pointcloud){

    m_is_reading_pointcloud = false;

    int32_t *data_received = (int32_t*)malloc(m_buffer_size);
    memcpy(data_received, m_pointcloud_data, m_buffer_size);

    points_received = 1;

    //process point cloud buffer
}

```

Full source code and examples can be found on GitHub:


- [Ros1 example](#)
- [Ros2 example](#)
- [Qt application example](#)



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	147/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

APPENDIX 2 READING AN IMAGE

Reading an image requires UDP socket management and an infinite loop reading UDP packets. In the infinite loop, check for 3 different sizes of packets:

- 11 Bytes for header packet

```
if (size_read == 11) { //Header
    memcpy(&m_image_height, &buffer[1], 2);
    memcpy(&m_image_width, &buffer[3], 2);
    memcpy(&m_image_channels, &buffer[5], 1);
    memcpy(&m_timestamp, &buffer[6], sizeof(uint32_t));
    m_image_data_size = m_image_height * m_image_width * m_image_channels;
    image_pointer = (uint8_t *)malloc(m_image_data_size);
    m_is_reading_image = true;
    bytes_count = 0;
}
```

- 8000 Bytes or less for data packets

```
if (size_read > 0 && m_is_reading_image){

    if(m_image_detections > 0){
        //read detections packages
        memcpy(&curr_det.confidence, &buffer[0], 2);
        memcpy(&curr_det.box.x, &buffer[2], 2);
        memcpy(&curr_det.box.y, &buffer[4], 2);
        memcpy(&curr_det.box.height, &buffer[6], 2);
        memcpy(&curr_det.box.width, &buffer[8], 2);
        memcpy(&curr_det.label, &buffer[10], 2);
        memcpy(&curr_det.red, &buffer[12], 1);
        memcpy(&curr_det.green, &buffer[13], 1);
        memcpy(&curr_det.blue, &buffer[14], 1);
        m_detections.push_back(curr_det);
        --m_image_detections;
    }

    memcpy(&image_pointer[bytes_count], buffer, size_read);
    bytes_count+=size_read;
}
```



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	148/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

- 1 Byte for footer packet

```

if (size_read == 1 && bytes_count == m_image_data_size) { // End, process image
    m_is_reading_image = false;
    cv::Mat image_to_show;
    bytes_count = 0;

    if (m_image_channels == 1){
        image_to_show = cv::Mat(m_image_height, m_image_width, CV_8UC1, image_pointer);
        cv::cvtColor(image_to_show, image_to_show, cv::COLOR_GRAY2BGR);
    }else if(m_image_channels == 2){
        image_to_show = cv::Mat(m_image_height, m_image_width, CV_8UC2, image_pointer);
        cv::cvtColor(image_to_show, image_to_show, cv::COLOR_YUV2BGR_Y422);
    }else if (m_image_channels == 3){
        image_to_show = cv::Mat(m_image_height, m_image_width, CV_8UC3, image_pointer);
    }

    cv::cvtColor(image_to_show, image_to_show, cv::COLOR_BGR2RGB);

    //!process image as desired
}

```

Full source code and examples can be found on GitHub:

- [Ros1 example](#)
- [Ros2 example](#)
- [Qt viewer example](#)



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	149/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

APPENDIX 3 READING TEMPERATURES DATA

Reading temperature data array requires UDP socket management and an infinite loop reading UDP packets. In the infinite loop, check for 3 different sizes of packets:

- 9 Bytes for header packet

```
if (size_read == 9) { //Header
    memcpy(&m_image_height, &buffer[1], 2);
    memcpy(&m_image_width, &buffer[3], 2);
    memcpy(&m_timestamp, &buffer[5], 4);

    m_image_data_size = m_image_height * m_image_width * sizeof(float);

    thermal_data_pointer = (float *)malloc(m_image_data_size);
    m_is_reading_image = true;
    bytes_count = 0;
    float_pointer_cnt = 0;
}
```

- 8000 Bytes or less for data packets

```
if (size_read > 0 && m_is_reading_image){
    memcpy(&thermal_data_pointer[float_pointer_cnt], buffer, size_read);
    bytes_count += size_read;
    float_pointer_cnt += size_read / 4;
}
```

- 1 Byte for footer packet

```
else if (size_read == 1 && bytes_count == m_image_data_size){ // End, send image
    m_is_reading_image = false;
    bytes_count = 0;
    float_pointer_cnt = 0;
    cv::Mat float_image = cv::Mat(m_image_height,m_image_width,CV_32FC1,thermal_data_pointer);
    //process thermal data
}
```

Full source code and examples can be found on GitHub:

- [Ros1 example](#)
- [Ros2 example](#)
- [Qt viewer example](#)



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	150/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual


APPENDIX 4 POLARIMETRIC PIXEL FORMAT

L3CAM supports one Lucid PHX050S1 polarized colour camera. These cameras have unprocessed BayerRG pixel formats. The data is transmitted by the camera following the GenICam standard's Pixel Format naming convention [RD 2]. L3CAM forwards the data using the same format.

The colour filter array of the PHX050S1 is as follows:

90°	45°	90°	45°
135°	0°	135°	0°
90°	45°	90°	45°
135°	0°	135°	0°

Figure 43. Colour filter array of Lucid PHX050S1.

Version:	1.8	 BEAM\GINE	Page:	151/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

APPENDIX 5 THERMAL IMAGE PROCESSING PIPELINES

The new SeekLibrary adds the processing pipelines that improves the thermal image quality, the possible values are **Lite**, **Legacy** and **SeekVision**, the differences between the 3 modes is shown in the following images.



Figure 44 Lite Thermal Image



Figure 45 Legacy Thermal Image



Figure 46 SeekVision Thermal Image



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	152/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

APPENDIX 6 PRODUCT DESCRIPTION

The following table shows the sensors that can be integrated in the L3CAM device, as specified in the provided test report.

Table 20. Compatible sensors list

ID	MANUFACTURER	MODEL	DESCRIPTION
CAX01	ECON	See3CAM_CU130_CHL	RGB
CAX02	Basler	daA2500 – 14uc	RGB
CAX03	Lucid	PHX050S-PC	Polarimetric Mono
CAX04	Lucid	PHX050S-QC	Polarimetric RGB
CAX05	Lucid	Phoenix GigE	RGB and Mono
CAX06	Seek Thermal	C306SP	34°
CAX07	Seek Thermal	C304SP	56°
CAX08	Allied Vision	Alvium 1800 U319c	RGB – USB3
CAX09	Allied Vision	Alvium 1800 U508c	RGB – USB3
CAX10	Seek Thermal	C302NP	105°
CAX11	ECON	See3CAM_CU22_CHL	110° RGB



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 L3CAM User manual	Page:	153/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

APPENDIX 7 RETURN CODES

Table 21. Library return codes.

RETURN CODE	DESCRIPTION
0	L3CAM_OK – The API call executed successfully
2	L3CAM_UNHANDLED_ERROR – Unexpected error in device
3	L3CAM_ERROR_DEVICE_IS_STREAMING – The device is already streaming; some parameters cannot be modified in this state
4	L3CAM_ERROR_UNDEFINED_PROTOCOL – The protocol index is not in the available protocol list.
5	L3CAM_ERROR_DEVICE_IS_STARTED – The device is already started; some parameters cannot be modified in this state.
11	L3CAM_NO_SENSORS_AVAILABLE – The L3CAM device was not able to find any sensor attached to it.
12	L3CAM_INVALID_PARAMETERS – The L3CAM device has received invalid parameters for the new network configuration.
50	L3CAM_ERROR_BUILDING_RTSP_PIPELINE – The application was unable to build the requested RTSP pipeline.
51	L3CAM_ERROR_STARTING_RTSP_SENDER – The application was unable to start the rtsp OpenCV engine, check the pipeline and the streaming parameters
52	L3CAM_ERROR_RTSP_PIPELINE_EMPTY – The requested gstreamer pipeline is empty
70	L3CAM_ERROR_GETTING_NETWORK_GATEWAY – The device is unable to retrieve network gateway information.
71	L3CAM_ERROR_GETTING_NETWORK_ADDRESS_INFO – The device is unable to retrieve network address information.
72	L3CAM_ERROR_GETTING_NETWORK_DHCP_INFO – The device is unable to retrieve DHCP configuration information.
210	L3CAM_TIMEOUT_ERROR – The library timed out waiting for a response from L3CAM
211	L3CAM_ERROR_OUT_OF_MEMORY – The library has no available memory
222	L3CAM_UNDEFINED_SENSOR – The sensor input is erroneous
223	L3CAM_VALUE_OUT_OF_RANGE – The user input for a value is erroneous
224	L3CAM_SENSOR_NOT_AVAILABLE – The sensor input is erroneous
225	L3CAM_FORBIDEN_SENSOR – The request for the specified ID sensor is erroneous.
230	L3CAM_ERROR_CREATING_TCP_CLIENT_SOCKET – Unable to create a socket to connect with the L3CAM TCP server
231	L3CAM_ERROR_INITIALIZING_WSA - Windows library version only. Error initializing the WSA subsystem.
232	L3CAM_ERROR_CONNECTING_WITH_TCP_SERVER – The library failed to connect with the L3CAM TCP server
233	L3CAM_ERROR_SENDING_TCP_MESSAGE – Error while trying to send a message over TCP to L3CAM
234	L3CAM_ERROR_CREATING_TCP_SERVER_SOCKET – Error creating the Library TCP server
235	L3CAM_EROR_BINDING_SOCKET – Error when binding library TCP socket, check local ports and system privileges.
236	L3CAM_ERROR_STARTING_TCP_SERVER – The library is unable to start the local TCP server.



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\AGINE L3CAM User manual	Page:	154/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

RETURN CODE	DESCRIPTION
237	L3CAM_ERROR_ACCEPTING_TCP_CLIENT – The local TCP server was unable to accept the L3CAM connection

Table 22. LiDAR errors

ERROR CODE	DESCRIPTION
100	ERROR_APD_CURRENT_NOT_DEFINED – The software version and firmware version mismatch
172	ERROR_LIDAR_TIMED_OUT – The internal communications of the LiDAR sensor have failed, restart the device.

Table 23. Econ RGB camera errors

ERROR CODE	DESCRIPTION
620	ERROR_OPENING_ECON_CAMERA – The L3CAM was unable to open the econ camera.
621	ERROR_SETTING_ECON_CAMERA_PARAMETER – The L3CAM failed when setting a parameter.
622	ERROR_SETTING_ECON_DEFAULT – The L3CAM failed when setting default values to econ RGB camera.

Table 24. Polarimetric camera errors.

ERROR CODE	DESCRIPTION
300	ERROR_POL_STD_EXCEPTION
301	ERROR_POL_RUNTIME_EXCEPTION
302	ERROR_POL_GENERIC_EXEPTION
303	ERROR_POL_TIMEOUT_EXCEPTION
304	ERROR_POL_UNDEFINED_ERROR
305	ERROR_POL_PERCEPTION_EXCEPTION

Table 25. Thermal camera errors.


ERROR CODE	DESCRIPTION
400	ERROR_OPENING_THERMAL_CAMERA
401	ERROR_THERMAL_IMAGE_SIZE
402	ERROR_CLOSING_THERMAL_CAMERA
403	ERROR_SETTING_THERMAL_CAMERA_LUT
404	ERROR_GETTING_THERMAL_CAMERA_LUT
405	ERROR_SETTING_THERMAL_CAMERA_SHARPEN
406	ERROR_SETTING_THERMAL_CAMERA_SMOOTH
407	ERROR_GETTING_THERMAL_CAMERA_SYSTEM
408	ERROR_SETTING_THERMAL_CAMERA_SYSTEM
410	ERROR_THERMAL_CAMERA_TIMEOUT - The thermal camera is disconnected.
411	ERROR_SETTING_THERMAL_CAMERA_SHUTTER
412	ERROR_INITIALIZING_THERMAL_CAMERA



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	155/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

ERROR CODE	DESCRIPTION
413	ERROR_CREATING_THERMAL_CAMERA_CALLBACK
414	ERROR_RELEASING_THERMAL_CAMERA
415	ERROR_STARTING_THERMAL_CAMERA
416	ERROR_SETTING_THERMAL_CAMERA_PIPELINE
417	ERROR_GETTING_THERMAL_CAMERA_SERIAL_NUMBER
418	ERROR_TRIGGERING_THERMAL_CAMERA_SHUTTER

Table 26. Allied Cameras errors

ERROR CODE	DESCRIPTION
500	ERROR_ALLIED_PARAMETER_NOT_DEFINED
501	ERROR_ALLIED_BLACK_LEVEL_NOT_AVAILABLE
502	ERROR_ALLIED_CONFIGURING_BLACK_LEVEL
503	ERROR_ALLIED_CONFIGURING_EXPOSURE_TIME
504	ERROR_ALLIED_EXPOSURE_TIME_NOT_AVAILABLE
505	ERROR_ALLIED_CONFIGURING_AUTO_EXPOSURE_TIME
506	ERROR_ALLIED_AUTO_EXPOSURE_TIME_NOT_AVAILABLE
507	ERROR_ALLIED_CONFIGURING_AUTO_EXPOSURE_RANGE_MAX
508	ERROR_ALLIED_AUTO_EXPOSURE_MAX_NOT_AVAILABLE
509	ERROR_ALLIED_CONFIGURING_AUTO_EXPOSURE_RANGE_MIN
510	ERROR_ALLIED_AUTO_EXPOSURE_MIN_NOT_AVAILABLE
511	ERROR_ALLIED_CONFIGURING_GAIN
512	ERROR_ALLIED_GAIN_NOT_AVAILABLE
513	ERROR_ALLIED_CONFIGURING_AUTO_GAIN
514	ERROR_ALLIED_AUTO_GAIN_NOT_AVAILABLE
515	ERROR_ALLIED_CONFIGURING_AUTO_GAIN_RANGE_MAX
516	ERROR_ALLIED_AUTO_GAIN_MAX_NOT_AVAILABLE
517	ERROR_ALLIED_CONFIGURING_AUTO_GAIN_RANGE_MIN
518	ERROR_ALLIED_AUTO_GAIN_MIN_NOT_AVAILABLE
519	ERROR_ALLIED_CONFIGURING_GAMMA
520	ERROR_ALLIED_GAMMA_NOT_AVAILABLE
521	ERROR_ALLIED_CONFIGURING_SATURATION
522	ERROR_ALLIED_SATURATION_NOT_AVAILABLE
523	ERROR_ALLIED_CONFIGURING_SHARPNESS
524	ERROR_ALLIED_SHARPNESS_NOT_AVAILABLE
525	ERROR_ALLIED_CONFIGURING_HUE
526	ERROR_ALLIED_HUE_NOT_AVAILABLE
527	ERROR_ALLIED_CONFIGURING_INTENSITY_AUTO_PRECEDENCE
528	ERROR_ALLIED_INTENSITY_AUTO_PRECEDENCE_NOT_AVAILABLE
529	ERROR_ALLIED_CONFIGURING_BALANCE_RATIO_SELECTOR
530	ERROR_ALLIED_BALANCE_RATIO_SELECTOR_NOT_AVAILABLE
531	ERROR_ALLIED_CONFIGURING_BALANCE_RATIO

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	156/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

ERROR CODE	DESCRIPTION
532	ERROR_ALLIED_BALANCE_RATIO_NOT_AVAILABLE
533	ERROR_ALLIED_CONFIGURUNG_AUTO_BALANCE_WHITE
534	ERROR_ALLIED_AUTO_BALANCE_WHITE_NOT_AVAILABLE
535	ERROR_ALLIED_CONFIGURING_BALANCE_WHITE_AUTO_RATE
536	ERROR_ALLIED_BALANCE_WHITE_AUTO_RATE_NOT_AVAILABLE
537	ERROR_ALLIED_CONFIGURING_WHITE_AUTO_TOLERANCE
538	ERROR_ALLIED_WHITE_AUTO_TOLERANCE_NOT_AVAILABLE
539	ERROR_ALLIED_CONFIGURING_AUTO_MODE_REGION_HEIGHT
540	ERROR_ALLIED_AUTO_MODE_REGION_HEIGHT_NOT_AVAILABLE
541	ERROR_ALLIED_CONFIGURING_AUTO_MODE_REGION_WIDTH
542	ERROR_ALLIED_AUTO_MODE_REGION_WIDTH_NOT_AVAILABLE
543	ERROR_ALLIED_CONFIGURING_INTENSITY_CONTROLLER_REGION
544	ERROR_ALLIED_INTENSITY_CONTROLLER_REGION_NOT_AVAILABLE
545	ERROR_ALLIED_CONFIGURING_INTENSITY_CONTROLLER_TARGET
546	ERROR_ALLIED_INTENSITY_CONTROLLER_TARGET_NOT_AVAILABLE
547	ERROR_ALLIED_CONFIGURING_MAX_DRIVER_BUFFERS_COUNT
548	ERROR_ALLIED_MAX_DRIVER_BUFFERS_COUNT_NOT_AVAILABLE
549	ERROR_ALLIED_GETTING_BLACK_LEVEL
550	ERROR_ALLIED_GETTING_EXPOSURE_TIME
551	ERROR_ALLIED_GETTING_AUTO_EXPOSURE_TIME
552	ERROR_ALLIED_GETTING_AUTO_EXPOSURE_RANGE_MAX
553	ERROR_ALLIED_GETTING_AUTO_EXPOSURE_RANGE_MIN
554	ERROR_ALLIED_GETTING_GAIN
555	ERROR_ALLIED_GETTING_AUTO_GAIN
556	ERROR_ALLIED_GETING_AUTO_GAIN_RANGE_MAX
557	ERROR_ALLIED_GETTING_AUTO_GAIN_RANGE_MIN
558	ERROR_ALLIED_GETTING_GAMMA
559	ERROR_ALLIED_GETTING_SATURATION
560	ERROR_ALLIED_GETTING_SHARPNESS
561	ERROR_ALLIED_GETTING_HUE
562	ERROR_ALLIED_GETTING_INTENSITY_AUTO_PRECEDENCE
563	ERROR_ALLIED_GETTING_BALANCE_RATIO_SELECTOR
564	ERROR_ALLIED_GETTING_BALANCE_RATIO
565	ERROR_ALLIED_GETTING_AUTO_BALANCE_WHITE
566	ERROR_ALLIED_GETTING_BALANCE_WHITE_AUTO_RATE
567	ERROR_ALLIED_GETTING_WHITE_AUTO_TOLERANCE
568	ERROR_ALLIED_GETTING_AUTO_MODE_REGION_HEIGHT
569	ERROR_ALLIED_GETTING_AUTO_MODE_REGION_WIDTH
570	ERROR_ALLIED_GETTING_INTENSITY_CONTROLLER_REGION
571	ERROR_ALLIED_GETTING_INTENSITY_CONTROLLER_TARGET
572	ERROR_ALLIED_GETTING_MAX_DRIVER_BUFFERS_COUNT



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	157/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

APPENDIX 8 STATUS

The L3CAM device returns the device status encoded in a 16-bit unsigned integer, each bit corresponds to a possible status.

Table 27. Status bits.

BIT	Description
1	BOOTING
2	ALARMS AT BOOTING
3	ERRORS AT BOOTING
4	BOOTING UNCONFIGURED
5	STARTED
6	STARTED WITH ALARMS
7	STARTED WITH ERRORS
8	STARTED UNCONFIGURED
9	STOPPING
10	STOPED
11	STOPED WITH ALARMS
12	UNUSED
13	UNUSED
14	UNUSED
15	UNUSED
16	UNUSED


The expected value when the initialization of the device is successful is 16. The binary representation of 16 is 0000 0000 0001 0000, enabling only the 5th bit of the uint16_t response value. Other states can be retrieved



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	158/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

APPENDIX 9 DATA STRUCTURES

Data structures defined in **beamagine.h** library file.

Enum sensorTypes

```
typedef enum sensorTypes {
    sensor_lidar = 1,
    sensor_econ_rgb,
    sensor_pol,
    sensor_thermal,
    sensor_allied_narrow,
    sensor_allied_wide
}sensorTypes;
```

This structure contains the possible sensors available in the L3CAM.

Table 28. sensorTypes enum fields.

Parameter	Description
Sensor_lidar	Identifier for LIDAR sensor
Sensor_econ_rgb	Identifier for RGB camera
Sensor_pol	Identifier for Polarimetric camera (Mono or RGB)
Sensor_thermal	Identifier for Thermal camera
Sensor_allied_narrow	Identifier for the allied camera 1800 U319c
Sensor_allied_wide	Identifier for the allied camera 1800 U508c

Enum pointCloudColor

```
enum pointCloudColor{
    RAINBOW = 0,
    RAINBOW_Z,
    INTENSITY,
    RGB_FUSION,
    POLARIMETRIC_FUSION,
    POL_PROCESSED_FUSION,
    THERMAL_FUSION,
    RGBT_FUSION,
    ALLIED_NARROW_FUSION = 12,
    ALLIED_WIDE_FUSION
};
```

This structure contains the possible colour values for the point cloud data.



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	159/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

Table 29. pointCloudColor enum fields.

Parameter	Description
RAINBOW	Colour that represents the spherical distance of the point
RAINBOW_Z	Colour that represents the distance on the Z-axis of the point
INTENSITY	Colour that represents the intensity of the point
RGB_FUSION	Point cloud with the RGB camera information fused
POLARIMETRIC_FUSION	Point cloud with the Polarimetric camera information fused
POL_PROCESSED_FUSION	Point cloud with the Polarimetric camera processed information fused
THERMAL_FUSION	Point cloud with the Thermal camera information fused
RGBT_FUSION	Point cloud with the RGB-Thermal fusion camera information fused
ALLIED_NARROW_FUSION	Point cloud with the RGB Allied 1800 U 319c camera information fused
ALLIED_WIDE_FUSION	Point cloud with the RGB Allied 1800 U 508c camera information fused

Enum alliedCamerasIds

```
enum alliedCamerasIds {
    wide_camera = 0,
    narrow_camera
};
```

This structure is used to identify if the sensor element refers to the Allied narrow camera (1800 U319c) or the Allied wide camera (1800 U508c)

Allied configuration values

```
#define allied_auto_precedence_minimize_noise 0
#define allied_auto_precedence_minimize_blur 1

#define allied_balance_ratio_selector_red 0
#define allied_balance_ratio_selector_blue 1

#define allied_controller_region_auto_mode_region 0
#define allied_controller_region_full_image 4
```


The defines are used for specific parameters of the Allied cameras where an integer with specific value is needed.



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	160/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

Enum streamingProtocols

```
typedef enum streamingProtocols {
    protocol_raw_udp = 1,
    protocol_gstreamer,
    protocol_rtsp
}streamingProtocols;
```

This structure describes the available streaming protocols for each sensor.

Table 30. streamingProtocols enum fields.

Parameter	Description
protocol_raw_udp	Value to enable Raw UDP streaming mode
protocol_gstreamer	Value to enable the RTSP streaming mode using gstreamer pipelines
protocol_rtsp	Value to enable RTSP streaming using RTP/UDP

Struct sensor

```
typedef struct sensor{
    streamingProtocols protocol;
    sensorTypes sensor_type;
    uint8_t sensor_status;
    uint8_t image_type;
    bool perception_enabled;
    bool sensor_available;
}sensor;
```

This structure describes the parameters of each sensor in the L3CAM.

Table 31. sensor struct fields.


Parameter	Description
Protocol	Streaming protocol of the sensor
Sensor_type	Identify the sensor type
Sensor_status	Integer with the current sensor status
Image_type	Identify the current image type that the sensor is streaming
Sensor_available	The availability of a sensor. If the sensor is meant to be, this parameter indicates if there has been an error while detecting the sensor.
Perception_enabled	Boolean that shows if the perception algorithm is enabled in the sensor



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8		Page:	161/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

Struct l3cam

```
typedef struct l3cam{
    char ip_address[16];
    char serial_number[16];
    char app_version[20];
    uint8_t model;
}l3cam;
```

This structure describes the parameters of each L3CAM device available in the network.

Table 32. l3cam struct fields.

Parameter	Description
Ip_addres	The IP address of the device
Serial_number	The serial number of the device
App version	The version of the L3CAM application.
Model	The model of the device

Enum econResolutions

```
typedef enum econResolutions{
    reso_640_480 = 1,
    reso_1280_720,
    reso_1920_1080
}econResolutions;
```

This structure describes the available resolutions for the econ RGB camera image streaming.

Table 33. econResolutions enum fields.

Parameter	Description
Reso_640_480	Use this value for a resolution of 640 x 480 px
Reso_1280_720	Use this value for a resolution of 1280 x 720 px (HD)
Reso_1920_1080	Use this value for a resolution of 1920 x 1080 px (FHD)



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	162/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

L3CAM User manual

Enum thermalTypes

```
typedef enum thermalTypes{
    thermal_WHITE = 1,
    thermal_BLACK = 17,
    thermal_IRON = 20,
    thermal_COOL = 2,
    thermal_AMBER = 9,
    thermal_INDIGO = 10,
    thermal_TYRIAN = 16,
    thermal_GLORY = 8,
    thermal_ENVY = 16,
    thermal_WHITE_NEW = 100,
    thermal_BLACK_NEW,
    thermal_SPECTRA,
    thermal_PRISM,
    thermal_TYRIAN_NEW,
    thermal_AMBER_NEW,
    thermal_IRON_NEW,
    thermal_HI,
    thermal_HILO,
}thermalTypes;
```

This structure contains the colormap names for the thermal camera, this is a duplicate of the thermal library structures.

Enum newThermalTypes

```
/*!thermal image modes beamagine_app version >= master_2.2.9
typedef enum newThermalTypes{
    new_thermal_WHITE_HOT = 0,
    new_thermal_BLACK_HOT,
    new_thermal_SPECTRA,
    new_thermal_PRISM,
    new_thermal_TYRIAN,
    new_thermal_IRON,
    new_thermal_AMBER,
    new_thermal_HI,
    new_thermal_GREEN,
}newThermalTypes;
```

This structure contains the colormap names for the new library version of thermal camera, this is a duplicate of the thermal library structures.



Beamagine S.L.

support@beamagine.com

<https://beamagine.com/>

Version:	1.8	 BEAM\GINE	Page:	163/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodriguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024


L3CAM User manual

Enum ThermalPipelines

```
typedef enum thermalPipelines{
    thermal_LITE = 0;
    thermal_LEGACY,
    thermal_SEEK,
}thermalPipelines;
```

This structure contains the index for the thermal pipelines imagery processing, this is a duplicate of the thermal library structures.



Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	164/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

APPENDIX 10 L3CAM DIMENSIONS

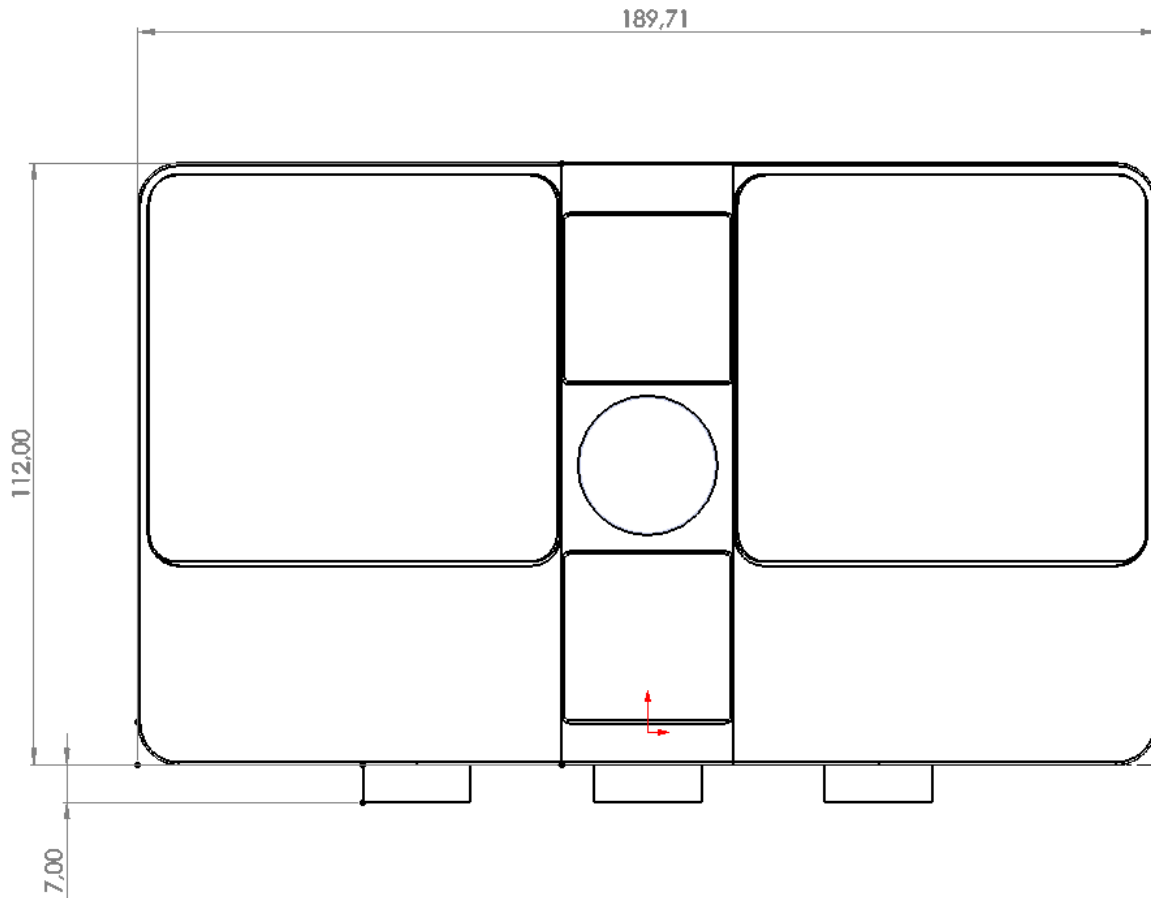


Figure 47. L3CAM front side dimensions.

Version:	1.8	 BEAM\GINE L3CAM User manual	Page:	165/165
Author/s:	Eduardo Bernal, Pablo Garcia, Ana Rodríguez, Isidro Bas		Approved by:	Jordi Riu
Revision date:	13/03/2024		Approval date:	14/03/2024

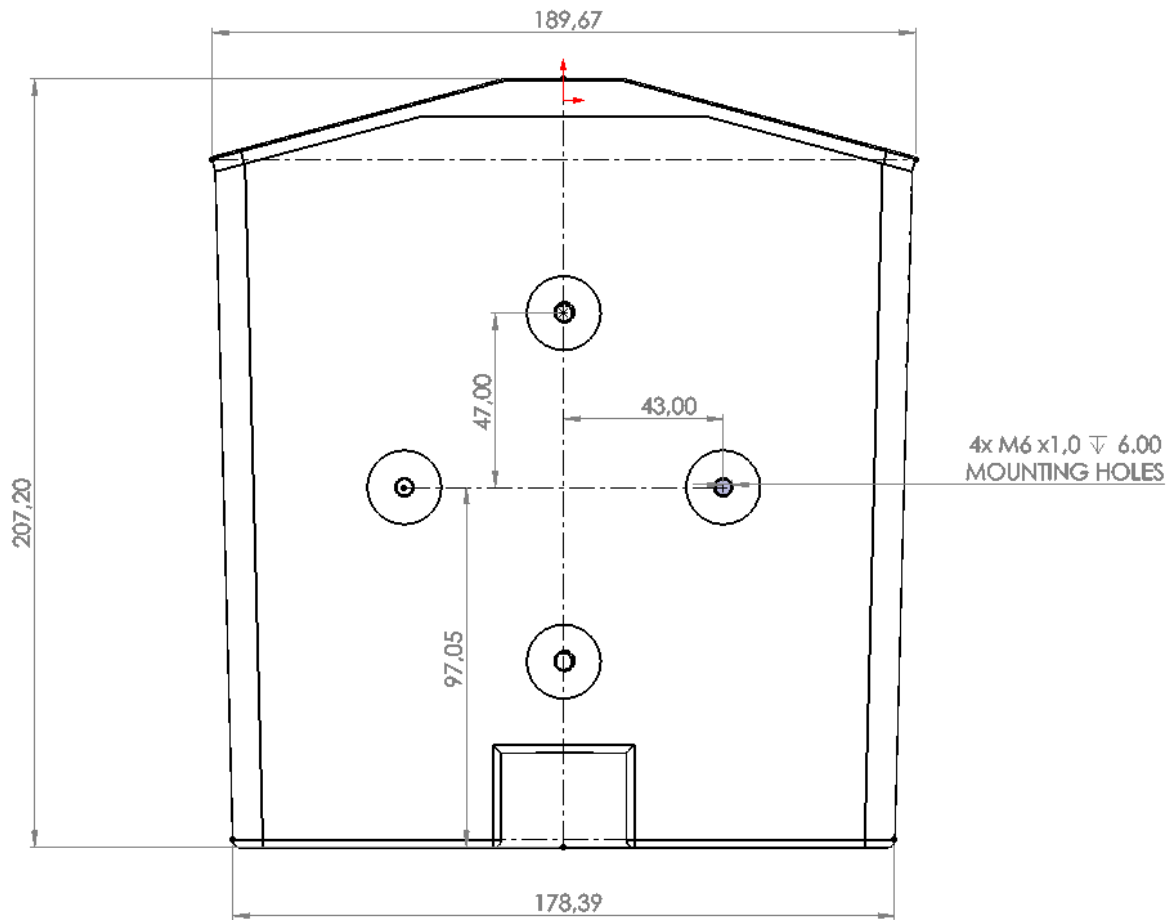


Figure 48. L3CAM mount holes dimensions.