1. ROS camera driver

1.1. What is ROS?

The Robot Operating System (ROS) is a set of software libraries and tools that help you build robot applications. From drivers to state-of-the-art algorithms and with powerful developer tools, ROS has what is needed for a robotics project. It is all open source (Source: ROS.org). For more details, also refer to ROS.org and ROS Wiki sensors.

1.2. Installation

System requirement: Linux operating system.

Download the “TOFCAM635_SW_Package” from the website www.espros.com, section Downloads, 02_Cameras_and_Modules.

There is enclosed the “TOFCAM635_ROS_driver” file. Unpack this ZIP file.

This driver works with two types of cameras: TOFcam635 and TOFcam635-S. The driver automatically detects which camera is connected. The camera TOFcam635-S has less parameters and unusable parameters will be ignored.

1.3. Running the ROS driver

Change to the home directory and open the bash-file:

```
> cd ~
> gedit .bashrc
```

Insert the following line at the end of the bash-file:

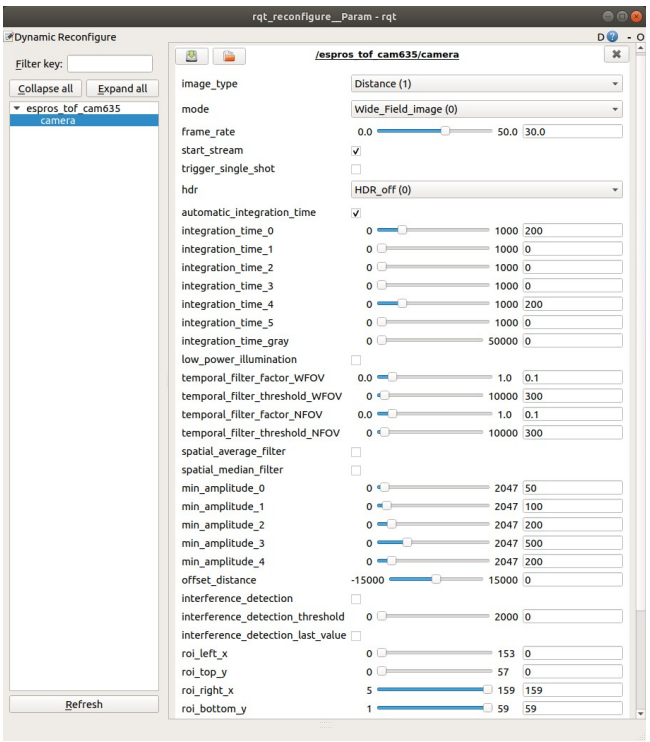
```
source ~/projects/cam635_driver/devel/setup.bash
```

Save the file, close editor and re-login linux.

Start the ROS with GUI in terminal mode with the following command:

```
roslaunch espros_cam635 camera.launch
```

The ROS tool opens with the different node windows and is ready to use.



Text 1: Figure 16: Example of the “dynamic reconfigure” node window

Start the camera operation by changing in the menu the parameter “start_stream” from false to true.

2. ROS API

This is the official driver for the ESPROS TOFcam635. The annotation follows the rules of ROS.org.

2.1. Start of the node

If you use in terminal mode the APIs only, without GUI:
Start the ROS operating system in a terminal with the command: `roscore&`
Start the cam635_node with the command: `rosrun espros_cam635 cam635_node`

2.2. Published topics

Topic name	ROS msgs file	ROS message type	Function
camera/image_raw1	sensor_msgs	Image	Sends the grayscale or amplitude image according the selected image type parameter
camera/image_raw2	sensor_msgs	Image	Sends the distance image for image type parameters which include distance
camera/image_header	std_msg	Int32MultiArray	Sends the image header. Refer to Table 2.
camera/points	sensor_msgs	PointCloud2	Sends the point cloud image for image type parameters which include distance

Table 1: ESPROS ROS topics

Entry	Index
Header version	1
Frame counter	2
Timestamp	3
TOFCOS version	4
Hardware version	5
Chip ID	6
Image width (x-axis)	7
Image height (y-axis)	8
Image origin X	9
Image origin Y	10
CurrentIntegrationTime3D WFOV	11
CurrentIntegrationTime3D NFOV	12
CurrentIntegrationTimeGrayscale	13
IntegrationTimeGrayscale	14
IntegrationTime0 WFOV	15
IntegrationTime1 WFOV	16
IntegrationTime2 WFOV	17
IntegrationTime3 WFOV	18
IntegrationTime4 NFOV	19
Integration time5 NFOV	20
reserved1	21

Entry	Index
reserved2	22
AmplitudeLimit0 WFOV	23
AmplitudeLimit1 WFOV	24
AmplitudeLimit2 WFOV	25
AmplitudeLimit3 WFOV	26
AmplitudeLimit4 NFOV	27
Binning type	29
DistanceTemporalFilter-Factor WFOV	30
DistanceTemporalFilter-Threshold WFOV	31
SingleValueTemporalFilter-Factor NFOV	32
SingleValueTemporalFilter-Threshold NFOV	33
Modulation frequency	34
Modulation channel	35
Flags	36
Temperature	37
Illumination beam	38
NFOV distance	39
NFOV amplitude	40
reserved3	41
n/a	n/a
n/a	n/a

Table 2: Header parameters (see also notes below)

Notes:

- ROS header bytes: In total 164 bytes. Each parameter is transmitted in corresponding 32 bit data format (Int32MultiArray).

2.3. Dynamically reconfigurable parameters

Refer for details on the dynamically reconfigurable parameters to the enclosed “dynamic_reconfigure package” or to http://wiki.ros.org/dynamic_reconfigure.

Parameter	Function	Data format	Default	Reference
~image_type	Sets the image acquisition type 0: Grayscale 1: Distance 2: Distance and amplitude 3: Distance and grayscale	int	1	n/a
~mode	Sets the operation mode 0: WFOV 1: NFOV manual 2: NFOV result 3: NFOV result and image 4: WFOV and NFOV result 5: Either WFOV, if object is inside range, else NFOV spot. 6: WFOV and NFOV image	int	0	n/a
~frame_rate	Sets image acquisition frame rate. Range 0 ... 50 frame/sec	double	30	n/a
~start_stream	Enables image streaming	bool	False	n/a
~trigger_single_shot	Starts single measurement after change from false to true	bool	False	n/a
~hdr	Sets HDR mode: 0: HDR OFF 1: HDR spatial 2: HDR temporal	int	0	n/a
~automatic_integration_time	Automatic mode: Integration time is set automatically for WFOV and NFOV between 1 and 1'000 µs.	bool	True	n/a
~integration_time_0	Sets the WFOV integration time for distance measurements in microseconds. Range: 1 … 4'000 µs	int	200	n/a
~integration_time_1		int	0	
~integration_time_2		int	0	
~integration_time_3		int	0	
~integration_time_4	Sets the NFOV integration time for distance measurements in microseconds. Range: 1 … 4'000 µs	int	200	n/a
~integration_time_5		int	0	
~integration_time_gray	Sets the integration time for grayscale measurements in microseconds. Range: 0 … 50'000 µs	int	0	n/a
~low_power_illumination	Enables low power illumination	bool	False	n/a
~temporal_filter_factor_wfov	Sets the factor 'k' of the WFOV temporal filter (Kalman)	int	10	n/a
~temporal_filter_threshold_wfov	Sets the threshold 'T' of the WFOV temporal filter (Kalman)	int	300	
~temporal_filter_factor_nfov	Sets the factor 'k' of the NFOV temporal filter (Kalman)	int	10	n/a
~temporal_filter_threshold_nfov	Sets the threshold 'T' of the NFOV temporal filter (Kalman)	int	300	
~spatial_average_filter	Enables the spatial average filter for distance filtering	bool	False	n/a
~spatial_median_filter	Enables the spatial median filter for distance filtering	bool	False	n/a
~interference_detection	Enables interference detection	bool	False	n/a
~interference_detection_threshold	Interference detection threshold. Range: 0 … 2000 mm	int	0	n/a
~interference_detection_last_value	Enables use last detection value	bool	False	n/a
~min_amplitude_0	Sets the amplitude limits for WFOV. Range 0 … 2'047 LSB	int	50	n/a
~min_amplitude_1		int	100	
~min_amplitude_2		int	200	
~min_amplitude_3		int	500	
~min_amplitude_4	Sets the amplitude limits for NFOV. Range 0 … 2'047 LSB	int	200	n/a
~offset_distance	Set distance offset. Range -15'000 … 15'000 mm	int	0	n/a
~roi_left_x	Sets the left edge of the ROI	int	0	n/a
~roi_right_x	Sets the right edge of the ROI	int	159	n/a
~roi_top_y	Sets the top edge of the ROI	int	0	n/a
~roi_bottom_y	Sets the top edge of the ROI	int	59	n/a

Table 3: ROS parameter table

Parameter	Function	Data format	Default	Reference
~lens_center_offset_x	Lens optical axis offset relative to sensor center (x direction). Range -100 … 100 pixels.	int	0	n/a
~lens_center_offset_y	Lens optical axis offset relative to sensor center (y direction). Range -100 … 100 pixels.	int	0	n/a
~enable_cartesian	Enables point cloud cartesian transformation (false = spheric)	bool	True	n/a
~enable_images	Activates imagePublisher1 and imagePublisher2 nodes to send information (camera/image_raw1/2)	bool	True	Table 1
~enable_point_cloud	Activates pointCloud2Publisher node to send information (camera/points)	bool	True	Table 1
~enable_image_header	Activates imageHeader node to send information (camera/imageHeader)	bool	True	Table 1

Table 3 cont: ROS parameter table

Notes: The parameters: mode, low_power_illumination, integration_time_4, integration_time5, min_amplitude_4, temporal_filter_factor_nfov, temporal_filter_threshold_nfov exist, but with the camera cam635-S are unused and will be ignored.