MandalaRobotics

Mandala 3D-Unit

User’s manual

Original manual

2015

Device identification m3d-unit

Please contact Mandala if you have any technical problem

Safety

Danger from unexpected movement of axis:

- Make sure that rotation head is clear to move
- Perform risk assessment according to EC machine directive
- Based on risk evaluation, design safety system for entire device, including other devices

Protection against electric shock die to PELV (protective extra low voltage)

- Working with device use only for electrical power PELV circuits in accordance to IEC DIN EN 60204-1 (Protective extra low voltage). Also fulfill general requirements in IEC/DIN EN 60204-1. Use only power sources which guarantee reliable electrical disconnection of the voltage as IEC/DIN EN 60204-1.

Intended use:

- Laboratory equipment,
- Research platforms,
- Navigation systems,

Device is not intended to use in life critical systems.

In event of damage caused by unintended use, unauthorized manipulation, violation of technical limit, manufacturer is not liable for damages.

Product overview

Device is intended to use with laser range finders and computer system. Refer to range finder manufacture for further information.

At this moment device allows to integrate with given devices:

Order number	Manufacturer	Necessary equipment
LMS100-10000 LMS101-10000 LMS102-10000 LMS111-10000 LMS151-10100 LMS153-10100	SICK AG	Mechanical adapter m3d-ma100 Interface cables m3d-c300
TIM55x	SICK AG	Mechanical adapter m3d-ma500 Interface cables m3d-c200

Device offers integration listed sensors in mobile systems. It is intended to use with open-source software delivered by manufacturer. There are two frameworks which are introduced:

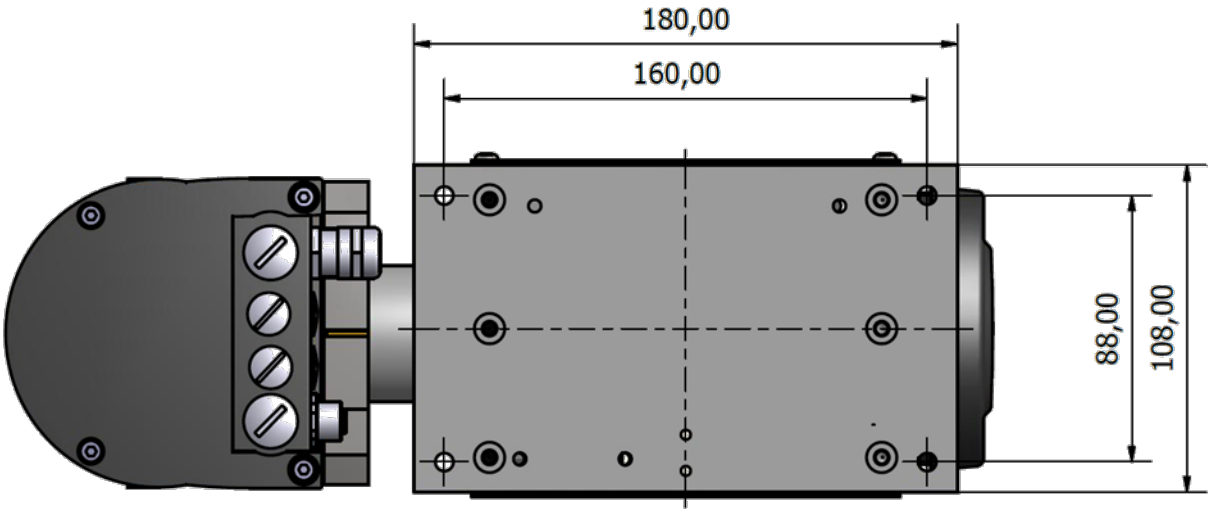
- High-integrated stand-alone driver
  This driver comes in form integrated software interface. It is intended to use in non-ROS robot systems, both on Linux and Windows platforms.

- ROS driver (Robot Operating System)
  This framework allows to integrate using ROS framework. There are multiple nodes provided (range finders driver, unit driver). This driver is more elastic solution and is recommended if ROS is present in system. At this moment Hydro and Indigo versions are supported.

For software installation, usage, building from source, refer repositories pages.

Installation

Device is intended to mount on flat stable surface, using four M5 screws.



Rangefinders can be mounted using mechanical adapters using universal mechanical interface. Both front and rotating lasers has same interface. To mount range finder follow these steps:

- Mount the range finder to adapter using included screws. Refer to range finder's manual for further information.
- Put adapter on mechanical interface, tighten tightening screw
- Connect cable set. Typically rangefinders are equipped with male and female M12 ports. Refer to range finder's manual for further information.

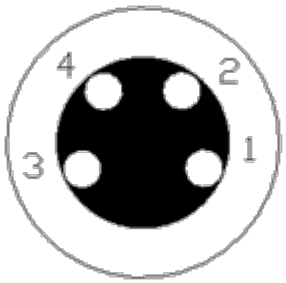
Connectors

Device is equipped with multiple interfaces organized in three panels.

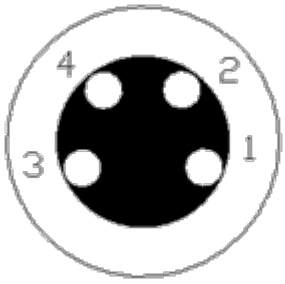
Panel	Connector	Interface
Front laser panel	M8-3pin	(X2.1)Front laser power supply
	M8-4pin	(X2.0)Front laser Ethernet port
Rotating laser panel	M8-3pin	(X1.1)Front laser power supply
	M8-4pin	(X1.0)Front laser Ethernet port
Interface panel	M12-4pin	(X0.0)Power supply
	M12-4pin	(X0.1) External Ethernet
	M12-4pin	(X0.2) External Ethernet
	M12-8pin	(X0.3) RS-232 server port
	M12-8pin	(X0.4) I/O port

Interfaces X1.0 and X2.0 pinouts:

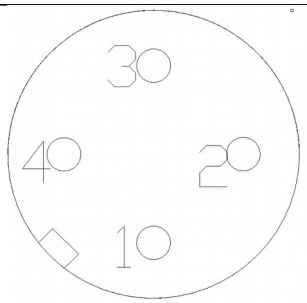
	Recommended color		
1	Brown	Vcc supply	
3	Blue	GND	

4	Black	N/C	 View from front of connector

Interfaces X1.1 and X2.1 pinouts:

	Recommended color		
1	Brown	RX+	 View from front of connector
2	White	TX-	
3	Blue	RX-	
4	Black	TX+	

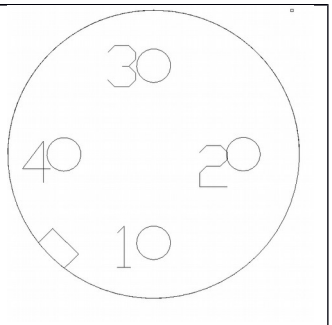
Interfaces X0.0

	Recommended color		
1	Brown	Vcc supply	 View from front of connector
3	Blue	GND	
4	Black	N/C	

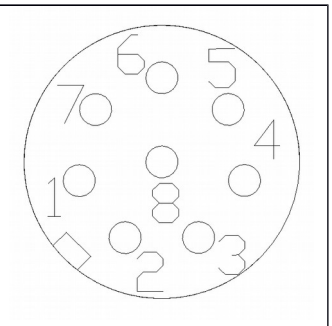
Note !

Keep right polarization. Inverting polarization will cause serious damages to unit

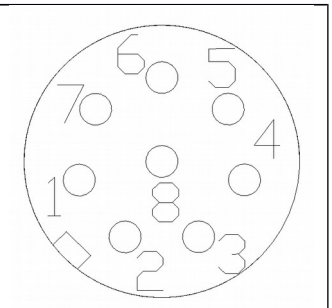
Interfaces X0.1 and X0.2 pinouts:

	Recommended color		
1	Brown	RX+	
2	White	TX-	
3	Blue	RX-	
4	Black	TX+	View from front of connector

Interface X0.3

	Recommended color		
1	Green/white	DCD	
2	Green	RTS	
3	Orange/White	DSR	
4	Blue	TxD	
5	Blue/white	RxD	
6	Orange	GND	
7	Brown/white	CTS	
8	Brown	DTR	View from front of connector

Interface X0.4

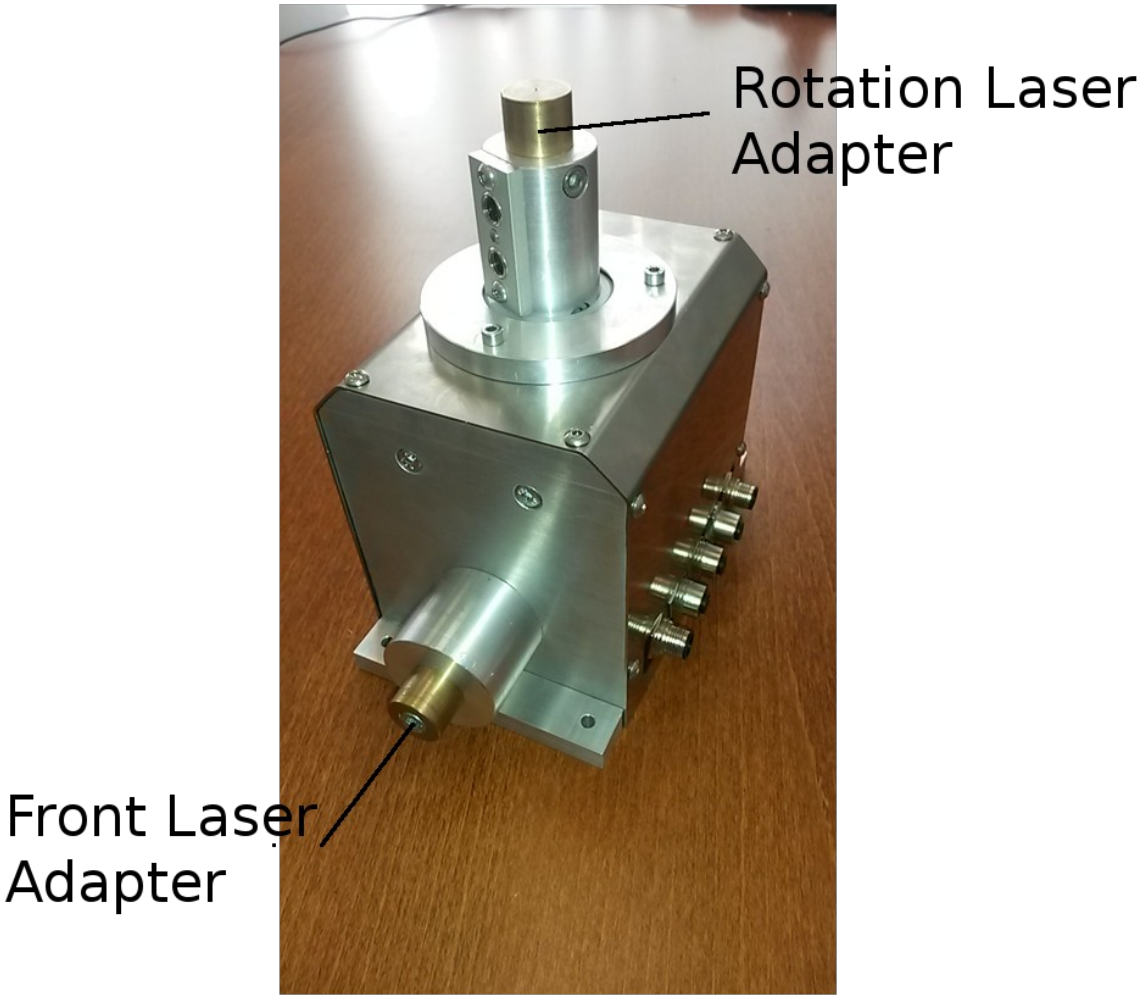
1		CAN-Lo	
2		CAN-Hi	
3		AIN0	
4		DIN0	
5		DIN1	
6		DOUT0	
7		VCC	
8		GND	View from front of connector

Key technical data

- Power supply: 12 – 24 VDC, maximum current (2A)
- Communication via Ethernet bus, build in Ethernet switch and RS-232 server for GPS or robot communication
- Signal of digital input:
  LOW (-10 .. 5V), HIGH (6 .. 30V)

- Signal of digital output
  LOW (-10 .. 5V), HIGH (6 .. 30V) 0-24VDC, max 500 mA.
- Signal of analog input :
  0 .. 10V
- Minimum rotation velocity 10 rpm, maximum rotation velocity 60 rpm
- Maximum torque 0.45 Nm
- 0.036˚ angular resolution
- IP44 protection
- Duralumin/stainless steel coating

Laser range finder assembly



Device is equipped with two universal mechanical and electrical adapters for laser range finders. First allows to mount horizontal 2D plane navigation rangefinder, second allows to mount rotation 3D rangefinder.

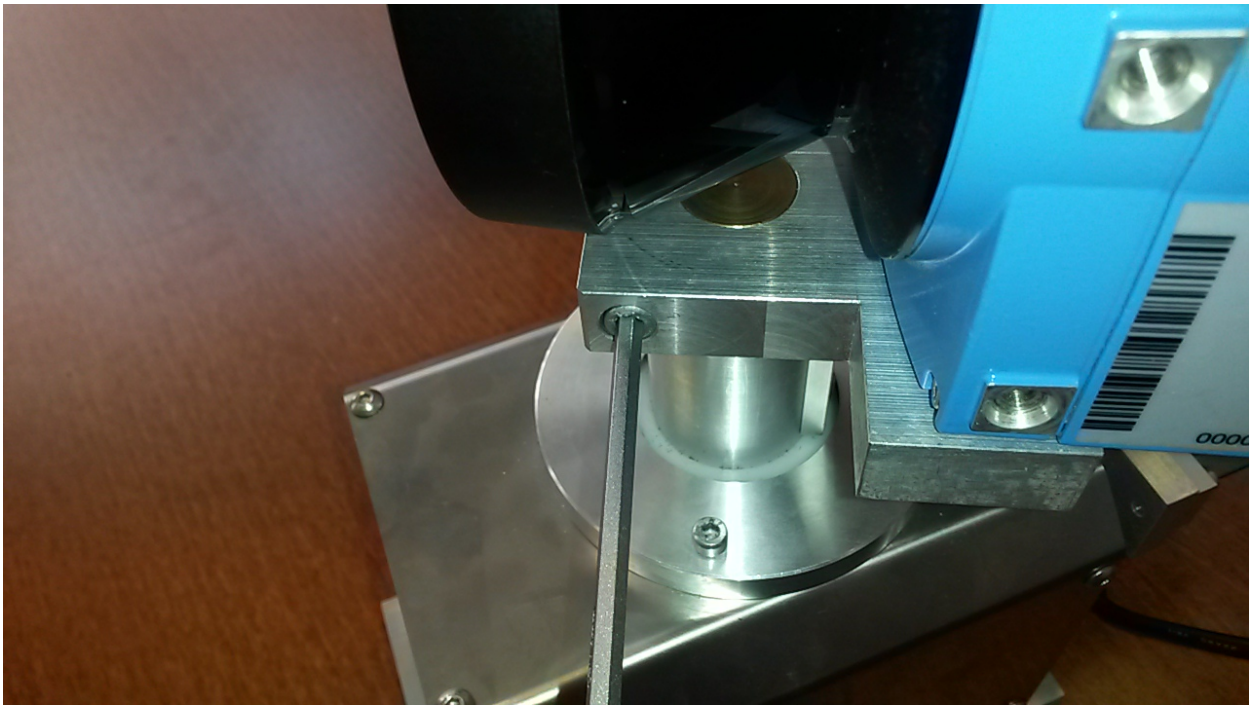


Figure 1 Rotation mechanical interface with laser range finder

Horizontal interface does not have lockage pin – that mean that angle of laser range finder can be regulated.

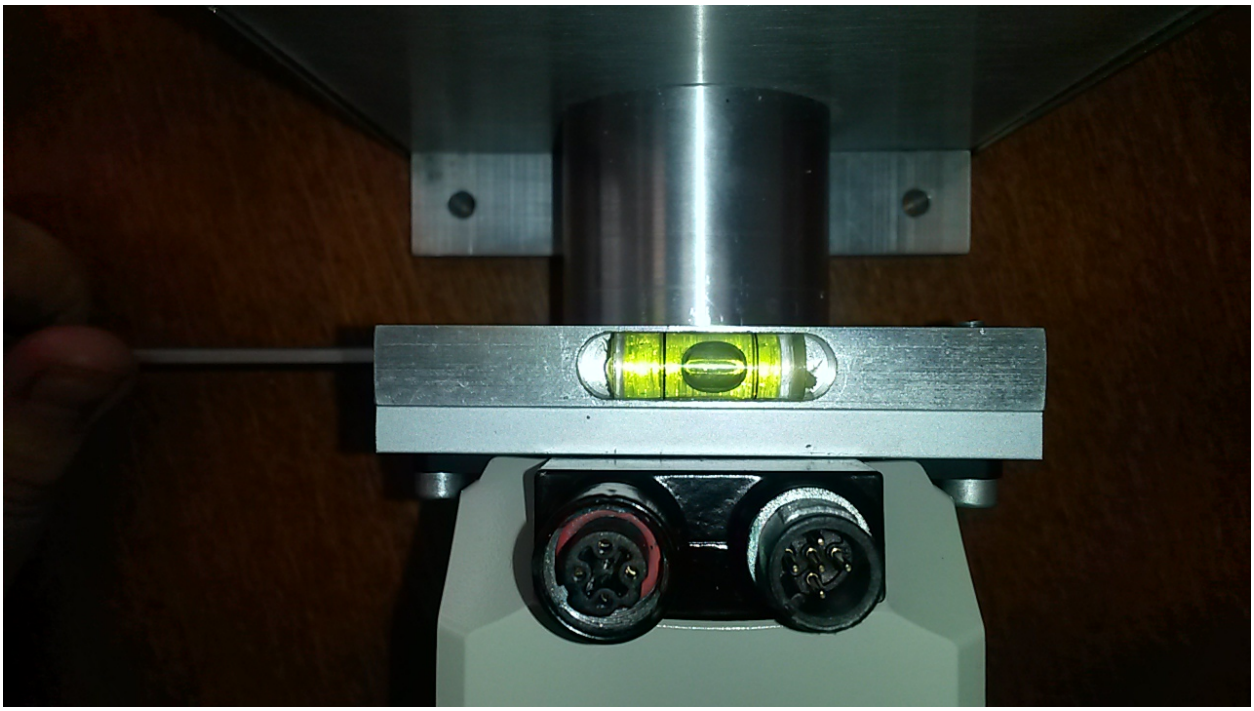


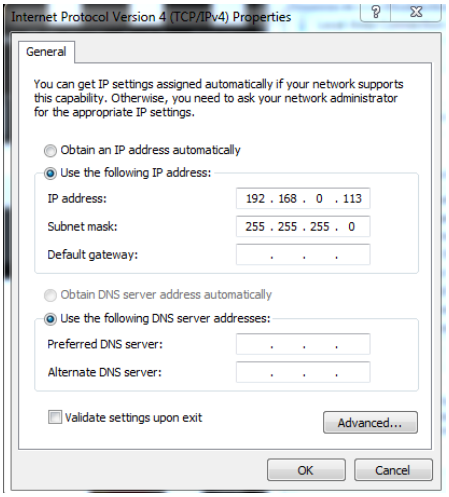
Figure 2 Front laser adapter

Setting up device

To start working with device following steps must be taken:

- Make sure that laser range finders are mounted properly on adapters
- Make sure that adapters are properly attached
- Connect included power supply or supply device using interface X0.0
- Connect Ethernet cable using attached cable using Ethernet interface X0.1 or X0.2
- Plugin Ethernet into yours computer adapter
- Set IP address of you network adapter. By default IP address of 3d unit is 192.168.0.150. Addresses of rangefinders can be set using SICK SOPAS. It is recommended to set computer's

adapter address to 192.168.0.xxx



- Run one of the examples for eg. simpleGL. Note that XML file must be regulated to suit current configuration

Software

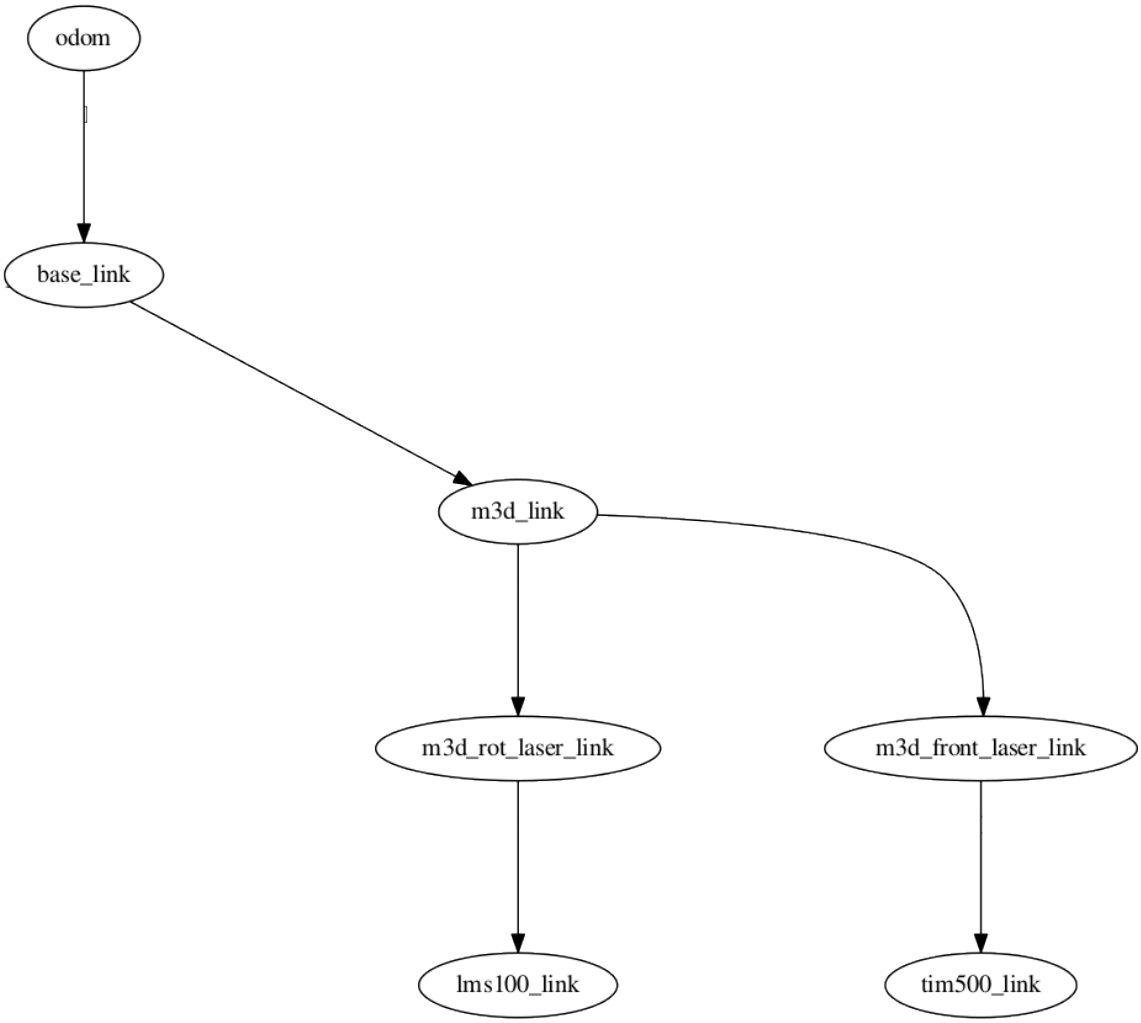
Device comes with two software solutions for multiple OS:

Software Driver	Linux	Windows
m3dunit_driver	ROS only (Linux Ubuntu)	
m3d_universal_driver	X	X

ROS driver

First solution is usable with ROS (www.ros.org) robotics system. This driver allows to publish current transformations of both adapters via tf system. It also accepts speed topics for device control. Driver comes with ROS drivers for laser rangefinders. That driver is operative with ROS Indigo and Hydro. For compilation, usage, and most recent manual: https://github.com/mandalarobotics/m3dunit_install/wiki

Driver publishes following tree:



Only frame m3d_rot_laser_link is not static and attached to rotating mechanical interface. m3d_front_laser_link is frame attached to front laser link. Frames lms100_link and tim500_link give transformations from mechanical interface to laser optical center for SICK TIM 500 and LMS100 are given:

Laser	XML code of transform publisher
LMS100 as rotating	<node pkg="tf" type="static_transform_publisher" name="lms100_link" args="0.0695 0 0.072 0 0 0 1 m3d_rot_laser_link lms100_link 100"/>
LMS100 as planar	<node pkg="tf" type="static_transform_publisher" name="lms100_link" args="0.0695 0 0.072 0 0 0 1 m3d_front_laser_link lms100_link 100"/>

TIM500 as rotating	<node pkg="tf" type="static_transform_publisher" name="tim500_link" args="0 0.0035 0.0 0 0 0 1 m3d_rot_laser_link tim500_link 100"/>
TIM500 as planar	<node pkg="tf" type="static_transform_publisher" name="tim500_link" args="0 0.0035 0.0 0 0 0 1 m3d_front_laser_link tim500_link 100"/>

Above code must be attached to launch script accordingly to actual configuration.

Laser driver are spawned with given launch code:

```
<node name="front_laser" pkg="sick_minimal_driver" type="sick_minimal_driver" output="screen">

<param name="ip" value="192.168.0.202"/>

<param name="frame_id" value="tim500_link"/>

<param name="inv" value="true"/>

</node>
```

This node has multiple parameters :

ip	Adress of laser. Can be changed using SICK SOPAS on Windows
frame_id	Frame in which output data will be published
Inv	Is laser is upside-down or not

Next m3d_unit driver launch code

```
<node name="rotation" pkg="m3dunit_driver" type="m3d_driver_node">

</node>
```

Optional, but very helpful is aggregator node. That node allows to aggregate scan data and republish it as Pointcloud2. It is especially useful in mapping applications.

```
<node name="aggregator" pkg="m3dunit_driver" type="m3d_pc_aggregator" output="screen">

<remap to="/rotation_laser/laserScan" from="/aggregator/rotLaserScan"/>
```

```
</node>
```

Point cloud aggregation can be triggered using /aggregator/rqst topic, for sake of usability joystick listener is introduced:

```
<node pkg="m3dold_unit_driver" type="joyToBool.py" name="joyTrig" output="screen">

<param name="buttonNo" value="2"/>

<remap to="/aggregator/rqst" from="/joyTrig/button"/>

</node>
```

Created that way launch file should be included to rest of robotics system. Third party software for laser drivers can be used, but result may be vary.

Universal driver

Universal driver is introduced for non-ROS systems. This driver can be used for standalone application or non-robotics, Windows based system. This driver is published as library with some examples. For usage or compilation, and most recent manual refer https://github.com/mandalarobotics

Driver is build using CMake. Applications using library can use CMake, but it is not essential. Minimum additional dependency is Boost (www.boost.org).

The most basic workflow is presented

```
m3d::_3dUnitConfig cfg;
// load config file to configuration object
cfg.readConfigFromXML("3dunitCfg.xml");
// create driver object
m3d::_3dUnitDriver* d= (new m3d::_3dUnitDriver (cfg));
// initialize, connect to physical devices
d->initialize();
// set rotation speed
d->setSpeed(8);
// request for pointcloud
d->requestPointcloud();
// wait for pointcloud, ddo something
d->waitForPointCloud();
...
...
...
//close
delete d;
```

There is also async method of pointcloud gathering. Please refer to example “simpleWrtier.cpp”

Like ROS driver configuration file has to be loaded in before connecting to Unit.

```
<?xml version="1.0" encoding="utf-8"?>
<!-- small unit with LMS100 -->
<m3dUnitDriver>
<outputFolder>outtest</outputFolder>
<adresses>
<rotLaser>192.168.0.201</rotLaser>
<unit>192.168.0.150</unit>
</adresses>
<angles>
<maximum> 3.14 </maximum>
<laserOffset>-135.0</laserOffset>
</angles>
<calibration>
<__type>Mat4</__type>
<__method>XYZYPR</__method>
<data> 0.0695 0 0 0 0 0</data>
</calibration>
</m3dUnitDriver>
```

In XML config file are set IP addresses of laser and Unit. Sections angles keeps maximum scanning angle and angular offset of starting angle of laser scanner. In case of SICK LMS100 or TIM500 it is -135.0.

Next is offset of optical center of laser scanner it is given using translation and Euler angles.