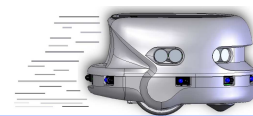


KHIII-LRE



user manual



version 1.0
december 2011

Documentation Author

Frédéric Lambercy
Julien Tharin
K-Team S.A.
Rue Galilee 9, Y-Park
1400 Yverdon-les-Bains
Switzerland

Email: info@k-team.com

Url: www.k-team.com

LEGAL NOTICE:

- The contents of this manual are subject to change without notice
- All efforts have been made to ensure the accuracy of the content of this manual. However, should any error be detected, please inform K-Team.
- The above notwithstanding, K-Team can assume no responsibility for any error in this manual.

TABLE OF CONTENTS

1. INTRODUCTION.....	1
1.1. HOW TO USE THIS MANUAL.....	1
1.2. SAFETY PRECAUTIONS.....	2
1.3. RECYCLING.....	2
2. THE LRF EXTENSION.....	3
2.1. UNPACKING AND INSPECTION.....	3
2.2. GLOBAL VIEW.....	3
2.3. LRF SENSOR ASSEMBLY.....	4
2.4. URG-04LX-UG01 SENSOR SPECIFICATIONS.....	6
2.5. BATTERY.....	7
3. CONNECTIONS.....	8
3.1. ASSEMBLING.....	8
3.2. DISASSEMBLING.....	9
3.3. UNPACKING TEST.....	9
4. PROGRAMMING THE LRF.....	10
4.1. INSTALLATION OF THE LATEST LIBKOREBOT AND CONFIGURATION FILE.....	10
4.2. USING THE LRF WITH KLRF_TEST SOFTWARE.....	12
4.2.1. <i>klrf_test</i> commands.....	12
4.3. USING KH3-LRF WITH PLAYER/STAGE.....	13
4.4. COMPILING YOUR OWN PROGRAM USING THE LIBKOREBOT.....	14
4.4.1. <i>Constants</i>	14
4.4.2. <i>Variables</i>	14
4.4.3. <i>High-level functions of the libkorebot</i>	15
4.5. SOFTWARE EXAMPLE.....	17
5. WARRANTY.....	18



1. INTRODUCTION

The hardware of the KheperaIII is based on a modular concept. The LRF is a turret that can be plugged on the robot **only if a KoreBotII is already plugged**. Due to the configuration of the LRF, other turrets cannot be plugged on the top of it.

The KheperaIII LRF turret was designed to adapt the URG-04LX-UG01 from Hokuyo. This module is a laser sensor for area scanning. The scan area is a 240° semicircle with a maximal distance of detection of 4m. The sensor retrieve one measure each 0.36° (682 steps exactly). For more information about the sensor, please look at the sensor specifications here:

http://ftp.k-team.com/KheperaIII/LRF/URG-04LX_UG01_spec.pdf

The KheperaIII LRF is equipped with a Li-Pol Battery to keep the autonomy of the KheperaIII robot at the same level as without an extension. This battery must be charged with the same charger as the KheperaIII robot.

This extension can be purchased with or without the URG-04LX sensor. In case you would like to purchase it by your own way, look at the chapter [2.3](#) to know how to mount the sensor on the turret.

1.1. How to use this manual

This manual introduces the KheperaIII LRF extension. To learn how to make the best use of your LRF turret, you are urged to read all the chapters 2 through 6.

If this manual does not answer one of the problems you are confronted with, please consult the K-Team web site (www.k-team.com) and especially the Forum and the FAQs.

- **Introduction:** Presentation of the LRF and the way to use it.
- **Unpacking and Inspection:** LRF's package description and first start-up.
- **The LRF sensor:** Description of all the LRF's functionalities.
- **Connections:** Explanation on how to connect (or disconnect) the LRF to the robot.
- **Programming the LRF:** Instructions to program the LRF using the libkorebot.
- **Warranty:** Legal notice of the LRF warranty.

1.2. Safety precautions

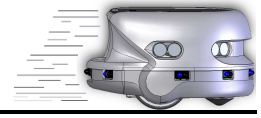
Here are some recommendations on how to correctly use the KheperaIII LRF:

- **Keep the turret away from wet area.** Contact with water could cause malfunction and/or breakdown.
- **Store your turret in a stable position.** This will avoid the risks of falls, which could break it or cause damage to a person.
- **Use only the official charger which is delivered with the KheperaIII robot.** Do not try to use another charger; this can cause irreversible damage to the battery.
- **Do not plug or remove the turret while the robot is powered on.** To avoid any damage, make all connections when the robot power is off.
- **Never leave the KheperaIII and the LRF powered when it is unused.** When you have finished working with KheperaIII, turn it off. It will save the battery life

1.3. Recycling

Think about the end of life of your robot! Parts of the robot can be recycled and it is important to do so. It is for instance important to keep batteries out of the solid waste stream. When you throw away a battery, it eventually ends up in a landfill or municipal incinerator. These batteries, which contain Lithium Polymer, can contribute to the toxicity levels of landfills or incinerator ash. By recycling the batteries through recycling programs, you can help to create a cleaner and safer environment for generations to come. For those reasons please take care to the recycling of your robot at the end of its life cycle, for instance sending back the robot to the manufacturer or to your local dealer.

Thanks for your contribution to a cleaner environment!



2. THE LRF EXTENSION

2.1. Unpacking and inspection

First check that you have a complete package. You should find:

- the KheperaIII LRF Turret
- mini-USB to Micro-MaTch cable
- URG-04LX-UG01 sensor (optional)
- the support CD with:
 - this User manual
 - the Libkorebot library

2.2. Global View

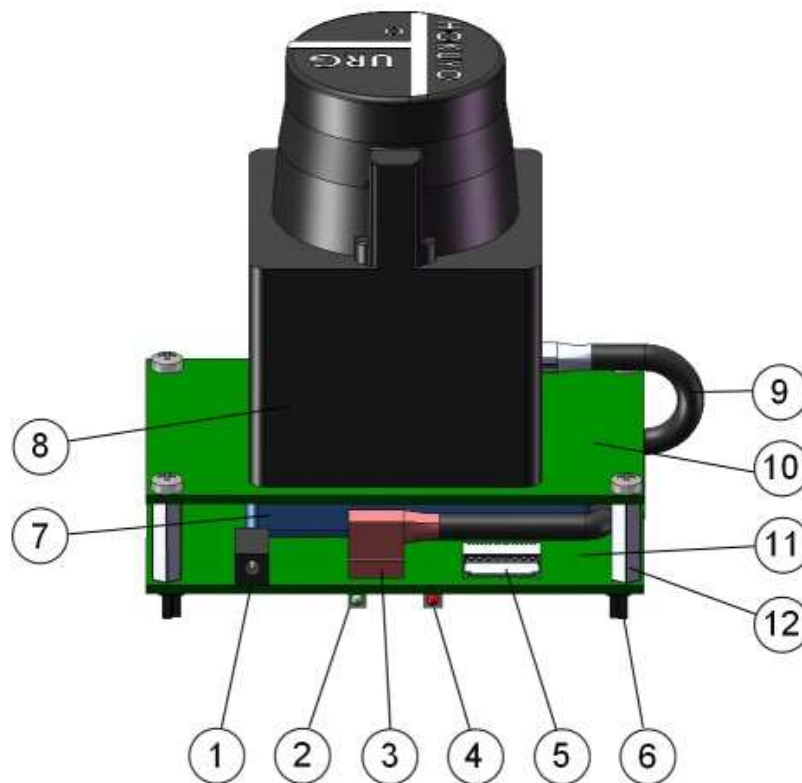
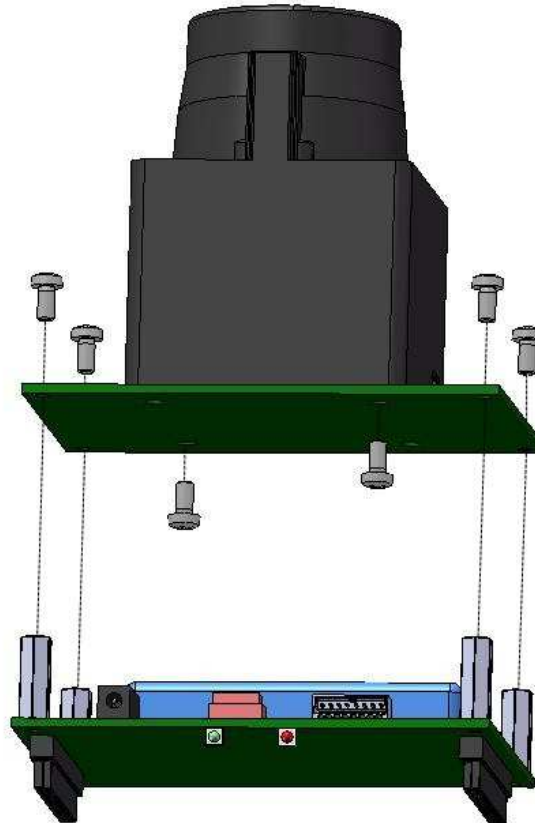


Figure 2.2: Overview of the KheperaIII LRF

1	Power input connector	7	Li-Pol Battery 950mAh 7.4V
2	Charge complete Led	8	URG-04LX-UG01 sensor
3	LRF connector	9	Mini-USB cable
4	Charge in progress Led	10	Fixing PCB
5	Serial connector (not used)	11	Main PCB
6	KB250 extension connector	12	Mechanical spacer

2.3. LRF sensor assembly

If you have purchased the LRF extension without the URG-04LX sensor, here's the step to follow:



- unscrew the four screws on the top of the turret to unmount the fixing PCB.
- mount the LRF on the TOP of the PCB (face where the indications FRONT & BACK are visible).
- Respect the orientation of the sensor as shown in the picture above.

- Use the two holes indicated in the picture below to screw the sensor.

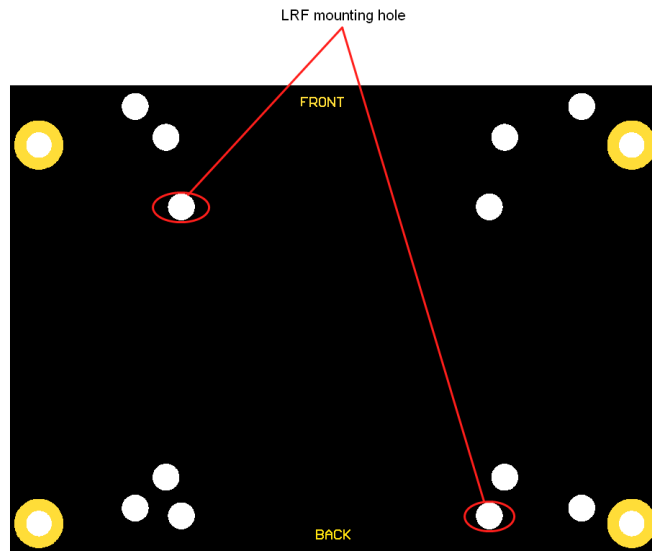


Figure 2.3: Mounting hole for the Hokuyo sensor

- Mount the fixing PCB with the sensor on the Main PCB of the turret and screw it.
- Connect the mini-USB/Micro-MaTch cable between the sensor and the Main PCB.

You can now mount your fully assembled turret on a KheperaIII to use it.

2.4. URG-04LX-UG01 sensor specifications

Light source	Semiconductor laser diode ($\lambda=785\text{nm}$),
Laser safety	Class 1 (IEC60825-1)
Power source	5V DC $\pm 5\%$ (USB buspower)
Current consumption	500mA or less (Rush current 800mA)
Detection distance	20mm ~ 4000mm
Accuracy	Distance 20mm ~ 1000mm : $\pm 30\text{mm}$ Distance 20mm ~ 4000mm : $\pm 3\%$ of measurement
Resolution	1 mm
Scan Angle	240°
Angular Resolution	0.36°
Scan Time	100msec/scan
Ambient (Temperature/Humidity)	-10 ~ 50°C / 85% or less (without dew and frost)
Preservation temperature	-25 ~ 75°C
Ambient Light Resistance	10000Lx or less
Impact Resistance	196 m/s ² , 10 times each in X, Y and Z direction
Insulation Resistance	10M Ω for DC 500Vmegger
Weight Approx.	160 g
Case	Polycarbonate

For more information, please look at the sensor datasheet:

http://ftp.k-team.com/KheperaIII/LRF/URG-04LX_UG01_spec.pdf

2.5. Battery

The KheperaIII LRF has an integrated battery to keep the autonomy of the robot at the same level than usual. The Li-Pol Battery of the LRF turret has a 950mAh capacity under 7.4V to ensure autonomy up to 2 hours.

This battery doesn't replace the KheperaIII robot battery; you must plug a charged battery in the robot to use the robot and the LRF. The LRF will be supplied only if it's mounted on a KheperaIII which is correctly powered and turn on. As soon as the KheperaIII is turn off or the battery is empty, the LRF supply will be cut.

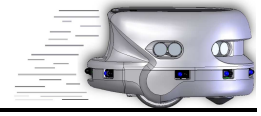
If you want to preserve the battery when you don't use the LRF sensor, you can cut the supply of the sensor in your application. Refer to Chapter [4](#) to know how to do it.

The charger is not provided with the LRF as it's the same as the KheperaIII Robot. To charge the LRF battery, you must use the same charger as the robot and connect it to the power connector (see n°4 in the overview). The red led (n°2 in the overview) will turn on to indicate that the battery is in charge. As soon as the battery is completely charged, the red led will turn off, and the green led (n°3 in the overview) will turn on.

You can charge the battery even if the LRF is mounted or not on a KheperaIII. The battery can also be charged if the KheperaIII and the LRF are turn on. The only difference will be the charged time. It's faster to charge the battery when the robot is turned off than the opposite.

Warnings:

- **Do not hot plug the charger on the LRF when it is in use. This may cause some errors with the I²C communication.**
- **Use only the official charger provided with the KheperaIII robot to charge the LRF.**



3. CONNECTIONS

Assembling and disassembling additional turrets is a delicate operation. Try to avoid it as much as possible and perform it carefully. Please follow the instructions below to avoid damage to your modules. K-Team can assume no responsibility for any damage caused by improper manipulation.

3.1. Assembling

Assembling is the easiest operation, but it is also necessary to perform it carefully:

- Before assembling the LRF on the KheperaIII, you must be sure that a KoreBotII is already mounted inside. If not, look at the KheperaIII User manual to know how to open the KheperaIII and plug the KoreBotII.
- If the black protection plate is mounted on KheperaIII, remove it before connecting the LRF and, very important, be sure that the KheperaIII robot is turn off.
- Then insert the LRF on the KheperaIII (the connectors of the Main PCB must be at the back of the robot). Do not try to mount the LRF the other way round; this can cause irreversible damages to the KoreBotII connector.
- When the LRF is correctly engaged in the KB-250 bus connectors, push the turret straight to plug it. If it is too hard to plug the turret, do not force on it. The connectors are certainly not correctly aligned.

3.2. Disassembling

This operation must be done very carefully and as infrequently as possible:

- First, switch OFF the KheperaIII robot; to remove the battery is not enough.
- Take the LRF turret with one hand and maintain the KheperaIII with the other. Do not pull on the URG-04LX sensor to unplug the LRF. Place your fingers on the Main PCB.
- Pull the LRF straight and very carefully. Once unplugged, place the LRF in its case to store it.

3.3. Unpacking test

After unpacking it is important to test the functionality of the LRF.

- Plug the LRF module on a KheperaIII equipped with a KoreBotII
- Connect the KheperaIII to a computer using a KoreConnect and a serial cable, or with ssh (see KheperaIII and Korebot II user manuals).
- Open a terminal on your computer and turn the robot on.
- Once the login passed, run the *klrf_test* program by typing *./klrf_test*. If it is not yet installed on your KoreBotII, follow the step in the section 4.1.
- Type the command *lrfinit* and push the RETURN key.

It should return the model, the motor speed and the connected port as follows:

```
model: URG-04LX(Hokuyo Automatic Co.,Ltd.)
scan_rpm: 600
URG is detected, port /dev/ttyACM0
```

- Type the command *lrfmeasure 0* and push the RETURN key.
 - ⇒ It should display continuously LRF data in a x-y graphic and you should recognize the environment shape around the LRF module.
- Push anykey then type the command *exit* and push the RETURN key. This will close the program.

If the LRF does not correctly perform this sequence of actions, please contact your local dealer.

More commands are described in the section 4.2.



4. PROGRAMMING THE LRF

The LRF is an extension that can be used only with the KoreBotII mounted in the KheperaIII. That means that the KoreBotII controls all the functions of the LRF. As all the KoreBotII extensions, a library including all the available functions is provided with the libkorebot version 1.17 or greater. If you already have a libkorebot installed in your computer but with an older version, you can download the latest version with the following link:

<http://ftp.k-team.com/KorebotII/software/common/libkorebot/libkorebot-VERSION.zip>
(Replace the *VERSION* by the version number, 1.17 or greater)

If your KoreBotII has already the latest libkorebot) installed, jump to section 4.2.

You can check if the last version is installed:

- Log on the KoreBotII (via ssh, Bluetooth or serial port)
- You can check if you have the 1.17 version in listing the present files:

```
ls -s /usr/lib/libkorebot*
```

⇒ This should give (end of the two lines):

```
/usr/lib/libkorebot.so -> /usr/lib/libkorebot.so.1.17  
/usr/lib/libkorebot.so.1.17
```

4.1. Installation of the latest libkorebot and configuration file

To use the LRF with the KheperaIII and KoreBotII, it's necessary to install the libkorebot version 1.17 or greater on the KoreBotII.

Normally, if you have received the KoreBotII at the same time than the LRF, the KoreBotII is ready to be used with the LRF; in this case, you can jump to section 4.2. Otherwise, if you have bought the LRF separately, you will need to execute the step described below:

- Log on the KoreBotII (via ssh, Bluetooth or serial port)

You have different ways to update the library:

Autonomously with the package:

- Upload the package file *libkorebot-1.17-r0_armv5te.ipk* by ssh, Bluetooth or Serial to the Korebot II.
- Remove the old one with the command: *ipkg remove libkorebot*
- Install the new one with the command:

```
ipkg install libkorebot-1.17-r0_armv5te.ipk
```

By hand:

- Remove the old libkorebot: ***rm /usr/lib/libkorebot****
- Copy or send the ***libkorebot.so.1.17*** file (locate in the ***build-korebot-2.6/lib/*** directory of the libkorebot source files) to the directory ***/usr/lib***.
- Link the libkorebot file as ***libkorebot.so***:

ln -s /usr/lib/libkorebot.so.1.17 /usr/lib/libkorebot.so

4.2. Using the LRF with `klrf_test` software

Before starting programming the LRF, it is important to test the LRF with the `klrf_test` software to understand the different functionalities and capabilities of the LRF. If the software is not yet in the `/home/root` directory of the KoreBotII, copy it (the executable file is located in the `src/tests/` directory of the `libkorebot`). Then execute it with the command: `./klrf_test`.

The `klrf_test` program waits that the user enters a command to control the lrf. To show all the available commands, type `help` and push the “RETURN” key. These commands are described below. See chapter 3.3 for a typical sequences of commands.

Execute `poweron` then `lrfinit` : a message appears as in the unpacking test chapter 3.3. If an error appears return to section 4.1 and try to install a new version of the `libkorebot` library.

Warning: this software does not support the backspace command. If you have typed a wrong command, push the “RETURN” key and restart typing the correct command. All wrong commands will be return an error message. The parameters of the command are separated by a space, but some commands do not need any parameter. In this case, type only the command and finally push the “RETURN” key.

4.2.1. `klrf_test` commands

`poweron`

Effect: set the laser battery module power on. The lrf green led starts blinking.
Parameter: -

`lrfinit`

Effect: Initialize the lrf communication and turn the laser beam on. The `poweron` command should be run before.
Parameter: -

`laseron`

Effect: set the laser beam on. The lrf green led becomes steady on.
Parameter: -

`laseroff`

Effect: set the laser beam off (module is still active and consumes battery; use `poweroff` to switch power supply off). The lrf green led starts blinking.
Parameter: -

poweroff

Effect: set the laser battery module power off. The lrf green led becomes steady off.
Parameter: -

setfile

Effect: set the filename where the output of the lrf is saved. If not set, no data will be saved. File format is text. First and second lines are the header. Next are the data, with tab between fields: current measurement index, distance index in the current measurement, angle, distance, x position, y position. See figure 4.3 in chapter 4.4 for more details.
Parameter: name of the file in a char-array format

lrfmeasure

Effect: get lrf data, save raw data to file if specified by the *setfile* command and plot processed data in a x-y graphic. Push anykey to stop or it stops after the number of measurements taken from the first parameter. During the measurement the lrf led is green and not blinking.
Parameter: first: number of measurements to take (0 = runs forever)
second (optional): number for averaging each measurement.

quit , exit or bye

Effect: Close the software and return to the KoreBotII main prompt.
Parameter: -

help

Effect: display the list of commands above.
Parameter: -

4.3. Using KH3-LRF with Player/Stage

You can use the KH3-LRF with the Player/Stage software framework. The instructions and files are available from the link below:

http://ftp.k-team.com/KheperaIII/player_stage/korebotII

4.4. Compiling your own program using the libkorebot

To make your own software to control the LRF, you need to install the development tools for the KoreBotII on your computer (with Linux OS). If it is not already done, please look at the KoreBotII User Manual to know how to install it. Once done, you can start writing your program. The best solution is to copy the *klrf_small_ex.c* source code (see chapter 4.5) and modify it. The source code *klrf_test.c* is a more complete example. In any case, keep your program in the *src/tests/* directory, and modify the Makefile to compile your new software: add your new program filename in line 27 of the Makefile in the *src/test* directory or start with the template program in the *template* directory.

As for all the KoreBotII extensions, you need to initialize the library before using the functions described below.

The libkorebot provides high-level functions which include all the available capabilities of the LRF. Look at the section 4.3.1 to view the different functions provided by the libkorebot.

4.4.1. Constants

Constant defined in the header file is described below:

LRF_DATA_NB

Number of data in one measurement defined like this:

```
#define LRF_DATA_NB 682
```

4.4.2. Variables

Variable for transferring data that is already declared in the header file is described below:

kb_lrf_DistanceData

Array of long containing measurements of one scan in [mm], declared like this:

```
long kb\_lrf\_DistanceData [LRF_DATA_NB]
```

You can transform from Polar to Cartesian coordinates systems, with the LRF module at the origin by looking at figure 4.3. Values between 0 and 19 are error codes (see http://www.hokuyo-aut.jp/02sensor/07scanner/download/data/URG_SCIP20.pdf, table 3 “ERROR CODES”).

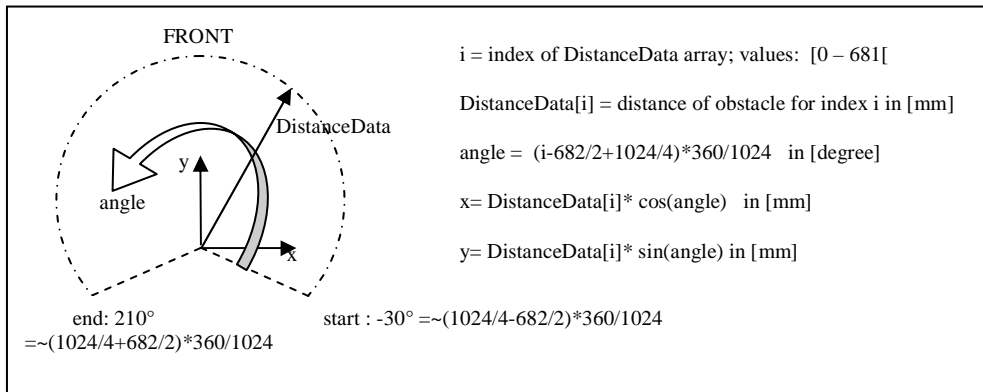


Fig 4.3 data description

4.4.3. High-level functions of the libkorebot

The **green** words indicate the type of the function (or parameter) and must not be included in your code.

The **blue** words are the parameters. You must declare it in your code before calling the function.

Only the **black** words can be directly added to your code.

int *kb_lrf_Init*(**char** **LRF_DeviceName*)

Initialize the library to use the different lrf functions and power the laser on. This function must be called at the beginning of each program using the lrf.

Return: a handle number if success or <0 if error

Parameter: ***LRF_DeviceName:** name of the device where the lrf is connected.
Should be "/dev/ttyACM0".

int *kb_lrf_GetDistances*(**int** *LRF_DeviceHandle*)

Get one set of radius distances between objects and the center of the lrf.

Return: a handle if success or <0 if error

Parameter: **LRF_DeviceHandle:** Value handle received by *kb_lrf_Init* function.

int *kb_lrf_GetDistances_Averaged*(**int** *LRF_DeviceHandle* , **int** *average*)

Get multiple set of radius distances between objects and the center of the lrf averages the data.

Return: a handle if success or <0 if error

Parameters: **LRF_DeviceHandle:** Value handle received by *kb_lrf_Init* function.
average: number of times the measure is averaged.

int kb_lrf_Close(int LRF_DeviceHandle)

It powers off the LRF and closes the port device.

Return: -

Parameter: **LRF_DeviceHandle:** Value handle received by *kb_lrf_Init* function.

void kb_lrf_Laser_On(int LRF_DeviceHandle)

Set the LRF laser beam on.

Return: -

Parameter: **LRF_DeviceHandle:** Value handle received by *kb_lrf_Init* function.

void kb_lrf_Laser_Off(int LRF_DeviceHandle)

Set the LRF laser beam off (module still initialized and motor keeps running).

Return: -

Parameter: **LRF_DeviceHandle:** Value handle received by *kb_lrf_Init* function.

void kb_lrf_Power_On(void)

Set the battery module power on.

Return: -

Parameter: -

void kb_lrf_Power_Off(void)

Set the battery module power off.

Return: -

Parameter: -

long kb_lrf_Get_Timestamp (void)

Get the timestamp of the last measurement set.

Return: Return the timestamp of the last measurement set in [ms].

Parameter: -

4.5. Software example

A small example of software controlling the LRF is enclosed below. You can find its source code file in *libkorebot-VERSION/src/tests/krlf_small_ex.c*.

```
#include <math.h>
#include <korebot/korebot.h>

// device where the LRF is connected: here USB port
#define LRF_DEVICE "/dev/ttyACM0"

int main( int argc , char * argv[] )
{
    int rc,i;
    int LRF_DeviceHandle; // serial port handle for lrf

    float angle,x,y;

    /* reset the screen */
    printf("\033[2J"); // erase the whole console */
    printf("\033[0;0f"); // Move cursor to the top left */

    printf("Led Range Finder Small Example Program (C) K-Team S.A.\r\n");

    /* Set the libkorebot debug level - Highly recommended for development. */
    kb_set_debug_level(2);

    // initialise the libkorebot
    if((rc = kb_init( argc , argv )) < 0 )
    {
        printf("\nERROR: port %s could not initialise libkorebot!\n");
        return -1;
    }

    kb_lrf_Power_On(); // activate the power supply battery module

    // initialise LRF device
    if ((LRF_DeviceHandle = kb_lrf_Init(LRF_DEVICE))<0)
    {
        printf("\nERROR: port %s could not initialise LRF!\n");
        return -2;
    }

    // get distances
    if (kb_lrf_GetDistances(LRF_DeviceHandle)<0)
    {
        printf("\nERROR: port %s could not initialise LRF!\n");
        kb_lrf_Close(LRF_DeviceHandle);
        return -3;
    }

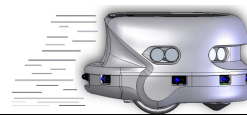
    printf("index dist[mm] angle[deg] x[mm] y[mm]\n");

    // process distances:
    // You have the distances radii from the center of the robot in [mm],
    // starting at -30 deg and rotating to counterclockwise direction
    // inside kb_lrf_DistanceData array. Values < 20 are errors.
    // You can get the distances average with function kb_lrf_GetDistances(lrfHandle,average).
    for (i=0;i<LRF_DATA_NB;i++)
    {
        angle= (i- LRF_DATA_NB /2+1024/4)*360.0/1024.0; // angle of each data

        // convert from polar to cartesian
        x=kb_lrf_DistanceData[i]*cos(angle*M_PI/180.0); // direction: right side of robot
        y=kb_lrf_DistanceData[i]*sin(angle*M_PI/180.0); // direction: front of robot
        printf("%3d\t%4d\t %+.1f \t%.1f \t%.1f\n",i,kb_lrf_DistanceData[i],angle,x,y);
    }

    // close the lrf device
    kb_lrf_Close(LRF_DeviceHandle);
    return 0;
}
```

5. WARRANTY



K-TEAM warrants that this product is free from defects in materials and workmanship and in conformity with the respective specifications of the product for the minimal legal duration, respectively one year from the date of delivery, under normal use conditions.

Upon discovery of a defect in materials, workmanship or failure to meet the specifications in the Product during the afore mentioned period, Customer must request help on K-Team Internet forum on <http://www.k-team.com/forum/> by detailing:

- The type of the product used (package, version, & serial number).
- The extension modules.
- The programming environment of the robot (standard, version, OS).
- The standard use of Product before the appearance of the problem.
- The description of the problem.

If no answer is received within two working days, Customer can contact K-TEAM support by phone or by electronic mail with the full reference as stated below

If the defect is identified as a “warranty” related problem, K-TEAM shall then, at K-TEAM's sole discretion, either repair such Product or replace it with the equivalent product without charging any technical labour fee and repair parts cost to Customer, under the condition that Customer brings such Product to K-TEAM within the period mentioned before. Repair or replacement under warranty does not entitle to original warranty team extension.

This limited warranty is invalid if the factory-applied serial number has been altered or removed from the Product.

This limited warranty covers only the hardware and software components contained in the Product. It does not cover technical assistance for hardware or software usage and it does not cover any software products contained in the Product.

This limited warranty is non-transferable.

It is likely that the contents of Customer's flash memory will be lost or reformatted in the course of the service and K-TEAM will not be responsible for any damage to or loss of any programs, data or other information stored on any media or any part of the Product serviced hereunder or damage or loss arising from the Product not being available for use before, during or after the period of service provided or any indirect or consequential damages resulting therefore.

If during the repair of the product the contents of the flash memory are altered, deleted or in any way modified, K-Team is not responsible whatever. Customer's

product will be returned to customer configured as originally purchased (subject to availability of software).

Be sure to remove all third parties' hardware, software, features, parts, options, alterations, and attachments not warranted by K-TEAM prior to Product service. K-TEAM is not responsible for any loss or damage to these items.

This warranty is limited as set out herein and does not cover, any consumable items (such as batteries) supplied with the Product; any accessory products which is not contained in the Product; cosmetic damages; damage or loss to any software programs, data, or removable storage media; or damage due to (1) acts of God, accident, misuse, abuse, negligence, commercial use or modifications of the Product; (2) improper operation or maintenance of the Product; (3) connection to improper voltage supply; or (4) attempted repair by any party other than a K-TEAM authorized robot service facility.

This limited warranty does not apply when the malfunction results from the use of the Product in conjunction with any accessories, products or ancillary or peripheral equipment, or where it is determined by K-Team that there is no fault with the Product itself.

K-Team expressly disclaims all other warranties than stated hereinbefore, expressed or implied, including without limitation implied warranties of merchantability and fitness for a particular purpose to the fullest extent permitted by law.

Limitation of Liability: In not event shall either party be liable to the other for any indirect, special, incidental or consequential damages resulting from performance or failure to perform under the contract, or from the furnishing, performance or use of any goods or service sold or provided pursuant hereto, whether due to a breach of contract, breach of warranty, negligence, or otherwise. Save that nothing herein shall limit either party's liability for death or personal injury arising from its negligence, neither party shall have any liability to the other for indirect or punitive damages or for any claim by any third party except as expressly provided herein.



K-Team S.A.
RUE GALILEE 9
1400 YVERDON-LES-BAINS
SWITZERLAND
