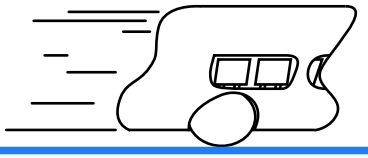
Khepera^2



user manual

K-TEAM

version 1.1
march 2002

Documentation author

K-Team S.A.
Ch. de Vuasset, CP 111
1028 Préverenges
Switzerland

email: info@k-team.com
Url: www.k-team.com

Trademark Acknowledgments

IBM PC: International Business Machine Corp.
Macintosh: Apple Corp.
SUN Sparc-Station: SUN Microsystems Corp.
LabVIEW: National Instruments Corp.
MatLab: MathWorks Corp.
Khepera: K-Team and LAMI

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

Khepera has originally been designed as a research and teaching tool in the framework of a Swiss Research Priority Program. It was first developed in 1992, by a research team from the Microprocessor and Interface Laboratory (LAMI) at the Swiss Federal Institute of Technology Lausanne (EPFL). It allows confrontation to the real world of algorithms developed in simulation for trajectory execution, obstacle avoidance, pre-processing of sensory information, hypothesis on behaviors processing. The Khepera robot is now widely used around the world as a platform for various robotics experiments and applications. The Khepera II robot has extended capabilities, is fully compatible with the original and can use any of the Khepera's accessories. It is an easy to use, robust, and standard platform for many robotics applications.

To be able to program the robot easily, LabVIEW® is proposed as a development environment. It is a graphical programming software, basically dedicated to instrumentation, which allows quick development of input-output interfaces, a necessity when dealing with the real world, by definition unpredictable and noisy. Please note that LabVIEW® is just a suggestion and is absolutely not needed to use the robot. Any other environment able to deal with the serial port of your computer can be use instead of LabVIEW®. Some of the products known to support communication with a Khepera are listed below, please visit K-Team website (www.k-team.com) for a complete list of supported products:

- Matlab® from *MathWorks*
- Sysquake® from *Calerga*
- Webots® from *Cyberbotics*
- YAKS® freeware (www.ida.his.se/ida/~johanc/yaks/)

Code can also be uploaded into the Khepera's memory for a standalone execution. Programs, written in C language or in M68000 assembly language, can be compiled under many environments using a cross compiler and uploaded in RAM or flashed in non-volatile memory. A complete API is available, either in C or assembly language, for programs to interface with the robot hardware.

The communication protocol implemented on Khepera and used by LabVIEW® is presented in section 6: Serial communication mode and described in detail in appendix. The API is fully documented in a separated manual, the “Khepera BIOS Reference Manual”.

1.1 How to Use this Manual

This manual is introducing the Khepera robots and its operating modes. For a quick start and overview of the robot's functions, please read chapter 1 to 5.

Refer to the following summary if a particular information is needed. If the manual does not cover a particular problem, many more technical documentation is available online from K-Team website (www.k-team.com) and especially a Frequently Asked Question document to solve most common problems and questions.

- **Unpacking and Inspection:** Khepera's package description
- **The robot and its accessories:** Khepera Hardware overview, Running modes list and main functions and accessories description
- **Unpacking Test:** First test to be performed after unpacking
- **Connections:** detailed cables connections for various usage
- **Serial Communication mode:** detailed description for the Serial communication mode between a computer and the robot.
- **S Loader mode:** How to download and application program into the Khepera's memory using the serial connection
- **User Application Mode:** How to store an user application in non volatile memory and instructions for standalone execution
- **uKos Upgrade Mode:** How to upgrade the robot's operating system
- **Using LabVIEW:** instructions to use the LabVIEW environment.

1.2 Safety Precaution

Check the unit's operating voltage before operation.
It must be identical with that of your local power supply. The operating voltage is indicated on the nameplate at the rear of the power supply.

Don't plug or unplug any connector when the system is switched ON.
All connections (including extension addition or disconnection) must be made when the robot and the interface are switched OFF. Otherwise damages can occur.

Switch OFF the robot if you will not use it for more than a day.
Disconnect the power supply removing it from the wall socket.

Do not open the robot if you are not explicitly allowed to do so.
Do not open the robot for any reason until explicitly instructed. K-Team cannot be held responsible for failure occurring after opening the robot. If disassembling the robot is necessary please request instructions and documentation from K-Team before proceeding.

Do not manually force any mechanical movement.
Avoid to force, by any mechanical way, the movement of the wheels or any other part.
If you have any question or problem concerning the robot, please contact your local Khepera dealer.

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 heavy metals, 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 UNPACKING AND INSPECTION

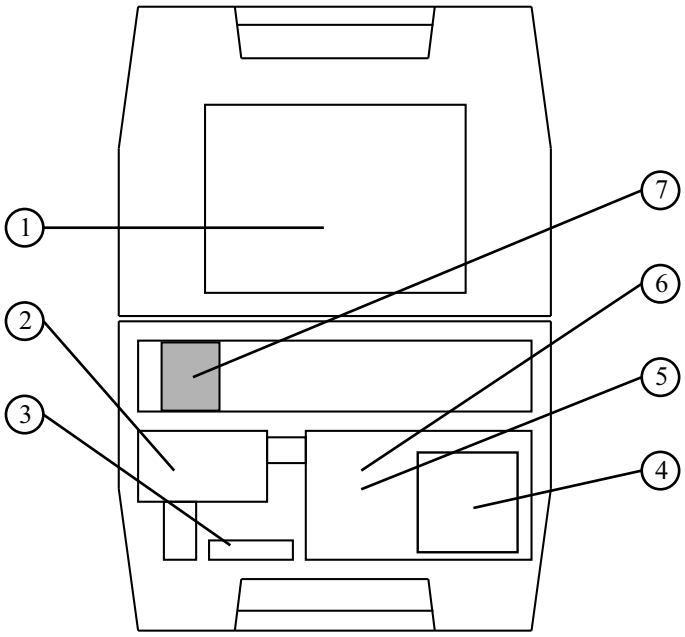


Figure 1: Position of the different parts in the bag.

Open the bag and check each item in the box against figure 1.

1. The documentation you are reading now
2. Power supply
3. Interface and charger module
4. The K-Team Support CD including;
   - Documentation
   - LabVIEW® VIs
   - Matlab® M-files
   - Cross Compiler
   - and more...
5. Cables:
   - Serial S cable
   - Battery charger cable
6. Spare parts:
   - Tires
7. Your Khepera robot in the basic version

3 THE ROBOT AND ITS ACCESSORIES

3.1 The Khepera Miniature Robot

3.1.1 Overview

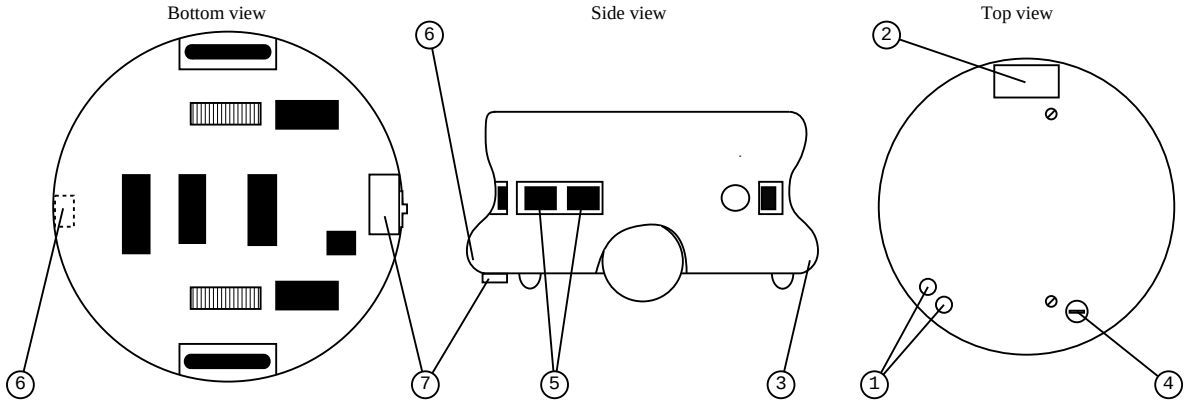


Figure 2: Overview of the Khepera robot

Make an external inspection of the robot. Note the location of the following parts:

1. LEDs
2. Serial line (S) connector.
3. Reset button.
4. Encoding wheel to select running mode.
5. Infra-Red proximity sensors.
6. Battery charger connector.
7. ON - OFF battery switch.

3.1.2 ON-OFF Battery Switch

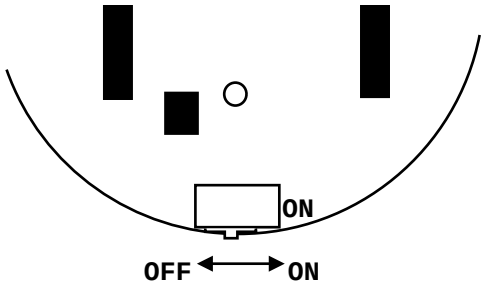


Figure 3: Position of the Battery power supply ON - OFF switch.

It allows the user to switch the battery of the robot ON or OFF. When ON, the robot is powered by the internal batteries. In this case the robot cannot be powered by an external supply. When switched OFF, the batteries are disconnected and the robot can be powered by an external power source, either the S connector or a power turret extension

3.1.3 Running modes, Reset Button and Settings

The ROM installed on your robot has an important library of software modules for the real time control of the Khepera robot. Part of these modules (building uKos) ensure the basic functionality of the Khepera robot, like motor control, sensors scanning etc. Another part of these modules ensure the interface with the user through the serial line.

Depending on your use of the robot (remote control, downloading, test, demo etc.) you can select a specific module by setting the correspondent running mode. The encoding wheel on the top of the Khepera is used to change mode (see section 3.1.1: Overview) The running mode is set according to the wheel position at boot time. One of the following sixteen modes can be selected:

0. Demonstration mode (9600 bits/s): Braitenberg vehicle algorithm (number 3 according to the “Vehicle” book [Braitenberg84]) for obstacle avoidance.
1. Serial communication mode (9600 bits/s): Mode to control the robot using the Serial communication protocol. The robot should be connected to a terminal using the S cable.
2. Serial communication mode (19200 bits/s): Same as mode 1.
3. Serial communication mode (38400 bits/s): Same as mode 1.
4. User application mode (9600 bits/s): Start an application stored in the robot's non volatile memory. The application should be flashed first using the S loader (see section 7 and 8 for details).
5. S loader mode (9600 bits/s): Robot waits for an application to be transferred in RAM and executes it when fully uploaded.
6. S loader mode (38400 bits/s): Same as mode 5.
7. Test mode (9600 bits/s): Successive tests are performed and the results are displayed using the serial link.
8. Serial communication mode (57600 bits/s): Same as mode 1.
9. Serial communication mode (115200 bits/s): Same as mode 1.
A. S loader mode (57600 bits/s): Same as mode 5.
B. User application mode (57600 bits/s): Same as mode 4.
C. Reserved
D. Reserved
E. Flash Erasing mode (38400 bits/s): The user segment of the non-volatile memory is erased.
F. uKos upgrade mode (38400 bits/s): Khepera BIOS upgrade mode.

The serial link set-up is always 8 bit, 1 start bit, 2 stop bit, no parity. Only the baud rate changes. The encoding wheel position can be changed at any time. **If the robot is running, a reset is necessary for the set-up to be effective.**

The reset button can be used at any time to reset the robot.

3.1.4 The S serial Line

The S serial line is an asynchronous serial line with TTL levels (0-5V). An interface is necessary to connect this line to a standard RS232 port. This interface is included in the interface/charger module present in the package (see section 3.4). The S serial line can power the robot. **The length of the S serial cable should be limited to two meters for proper operation.**

3.1.5 Motors and motor control

Each wheel is moved by a DC motor coupled with the wheel through a 25:1 reduction gearbox. An incremental encoder, placed on the motor axis, gives 24 pulses per revolution of the motor. This allows a resolution of 600 pulses per revolution of the wheel that corresponds to 12 pulses per millimeter of path of the robot.

The Khepera main processor has the direct control on the motor power supply and can read the pulses of the incremental encoder. An interrupt routine detects every pulse of the incremental encoder and updates a wheel position counter.

The motor power supply can be adjusted by the main processor by switching it ON and OFF at a given frequency and during a given time. The basic switching frequency is constant and sufficiently high not to let the motor react to the single switching. By this way, the motor react to the time average of the power supply, which can be modified by changing the period the motor is switched ON. This means that only the ratio between ON and OFF periods is modified, as illustrated in figure 4 This power control method is called “pulse width modulation” (PWM). The PWM value is defined as the time the motor is switched ON.

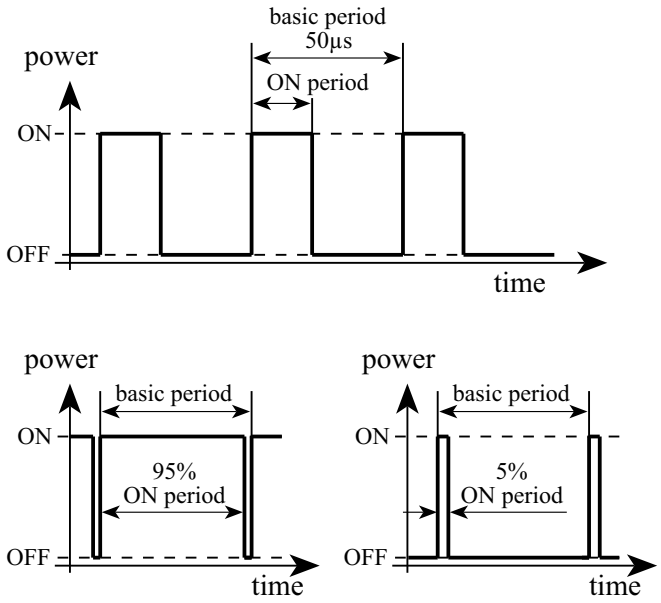


Figure 4: The "pulse width modulation" (PWM) power supply mode is based on a ratio between the ON time and the total time. The basic switching frequency is constant.

The PWM values can be set directly, or can be managed by a local motor controller. The motor controller can perform the control of the speed or position of the motor, setting the correct PWM value according to the real speed or position read on the incremental encoders.

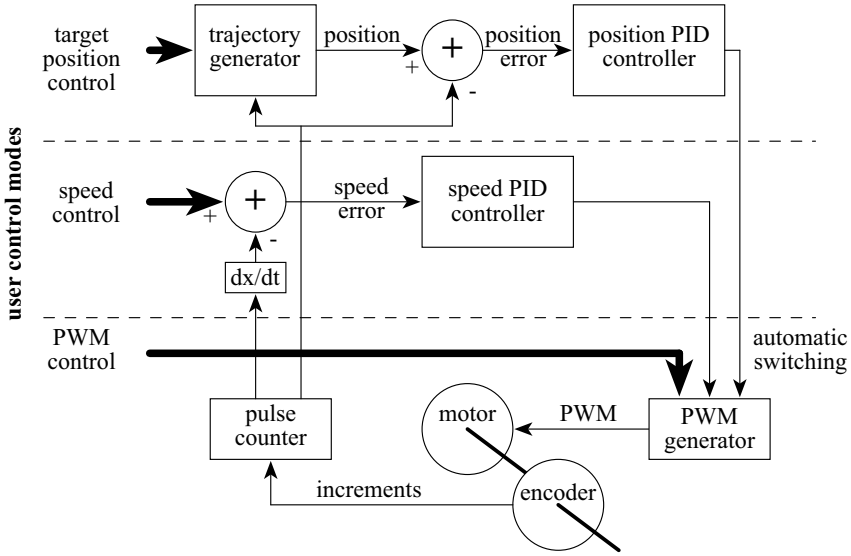


Figure 5: Structure of the motor controllers and levels of user access.

Both DC motors can be controlled by a PID controller executed in an interrupt routine of the main processor. Every term of this controller (Proportional, Integral, Derivative) is associated to a constant, setting the weight of the corresponding term: Kp for the proportional, Ki for the integral, Kd for the derivative.

The motor controller can be used in two control modes: The speed and the position modes. The active control mode is set according to the kind of command received. If the controller receives a speed control command, it switches to the speed mode. If the controller receives a position control command, the control mode is automatically switched to the position mode. Different control parameters (Kp, Ki and Kd) can be set for each of the two control modes.

Used in speed mode, the controller has as input a speed value of the wheels, and controls the motor to keep this wheel speed. The speed modification is made as quick as possible, in an abrupt way. No limitation in acceleration is considered in this mode.

Used in position mode, the controller has as input a target position of the wheel, an acceleration and a maximal speed. Using this values, the controller accelerates the wheel until the maximal speed is reached, and decelerates in order to reach the target position. This movement follows a trapezoidal speed profile, as described in figure 6.

The input values and the control mode of this controller can be changed at every moment. The controller will update and execute the new profile in the position mode, or control the wheel speed following the new value in the speed mode. A status of the controller indicates the active control mode, the phase of the speed profile (on target or in movement) and the position error of the controller.

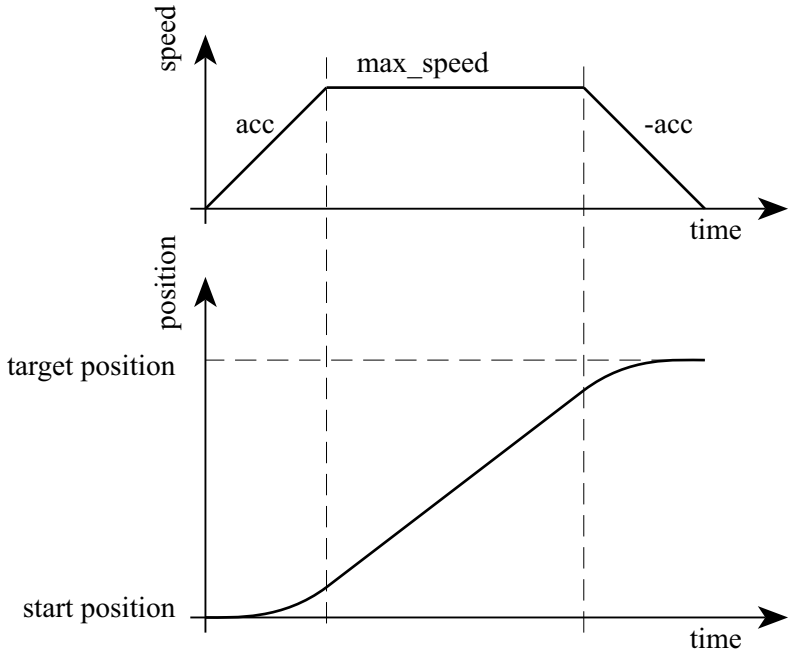


Figure 6: Speed profile used to reach a target position with a fixed acceleration (acc) and a maximal speed (max speed).

3.1.6 Infra-red Proximity sensors

Eight sensors are placed around the robot and are positioned and numbered as shown in figure 7.

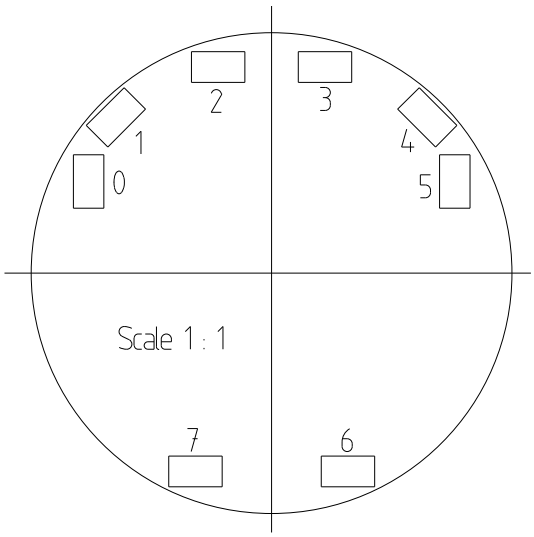


Figure 7: Position of the 8 IR sensors.

These sensors embed an infra-red light emitter and a receiver. For detailed description, please refer to the manufacturer's datasheet. The eight sensors are TCRT1000, reflective optical sensors form *Vishay Telefunken*.

This sensor device allows two measures:

- The normal ambient light. This measure is made using only the receiver part of the device, without emitting light with the emitter. A new measurement is

made every 20 ms. During the 20 ms, the sensors are read in a sequential way every 2.5 ms. The value returned at a given time is the result of the last measurement made.

- The light reflected by obstacles. This measure is made emitting light using the emitter part of the device. The returned value is the difference between the measurement made emitting light and the light measured without light emission (ambient light). A new measurement is made every 20 ms. During the 20 ms, the sensors are read in a sequential way every 2.5 ms. The value returned at a given time is the result of the last measurement made.

The output of each measurement is an analogue value converted to a 10 bit digital value . The following two sections (section 3.1.7 Ambient light measurements and section 3.1.8 Reflected light measurements) illustrate the meaning of this 10 bit values.

3.1.7 Ambient light measurements

Ambient light measurement is strongly influenced by the robot's environment. Depending on the light source type, color, and distance, ambient light measurement profile might vary. **It is not recommended to use light source with large emission in the infrared range, as this could confuse the IR sensors.** The following graph shows ambient light profile, for a 50W light source, placed above the Khepera at the given distance.

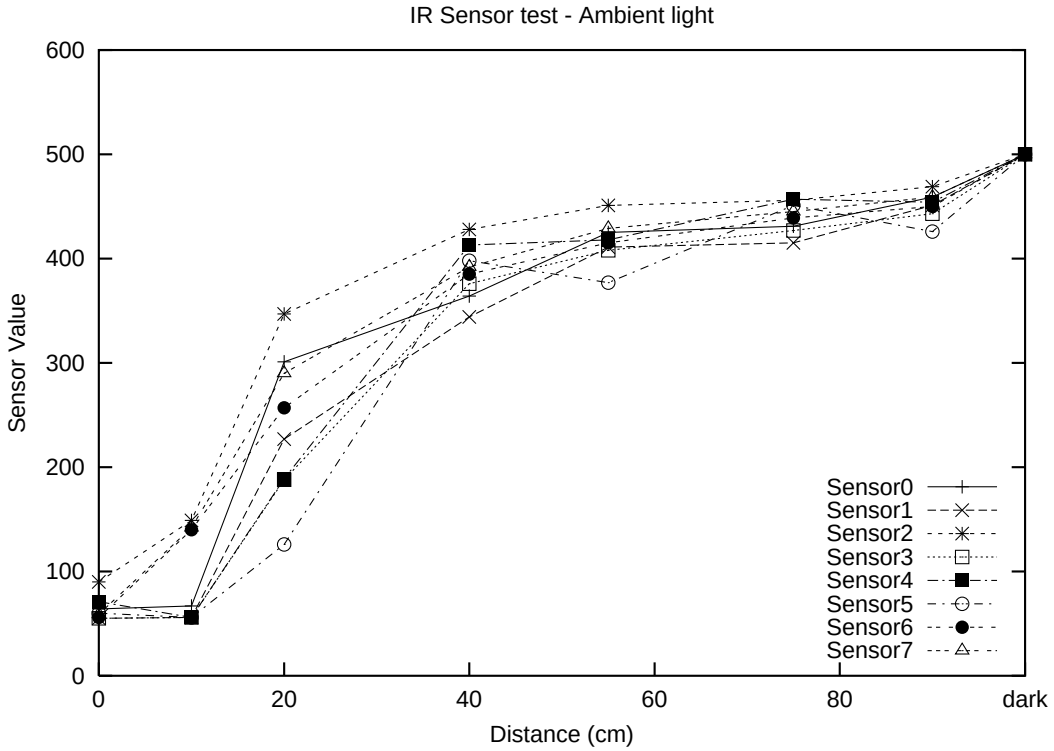


Figure 8: Ambient light measurement using a 50W light source

Sensors value is increasing when light intensity is decreasing. The maximum value is 500 units usually reached in nearly dark conditions. Ideal lighting conditions for the Khepera is an uniform, constant, and clear light, but sensors are robust enough to function under most circumstances.

3.1.8 Reflected light measurements

Sensors are mainly meant to detect obstacles around the Khepera. Measurements for reflected light depends on objects reflectivity and on ambient light conditions. Objects color, materials and surfaces do have an influence on the sensors response. Moreover, as any sensor,IR sensors are subject to environmental noise. For all these reasons, graphics below are given for information only and should not be considered as references. Please refer to the complete documentation on sensors sensitivity and tuning, for further information.

White paper detection and other robot detection test results are detailed below. These two tests are the easiest way to illustrate IR sensors response profile. Standard white paper, 5cm by 5cm square surface is used for the white paper detection test. The surface is placed in parallel with the robot's front sensors at the given distance.

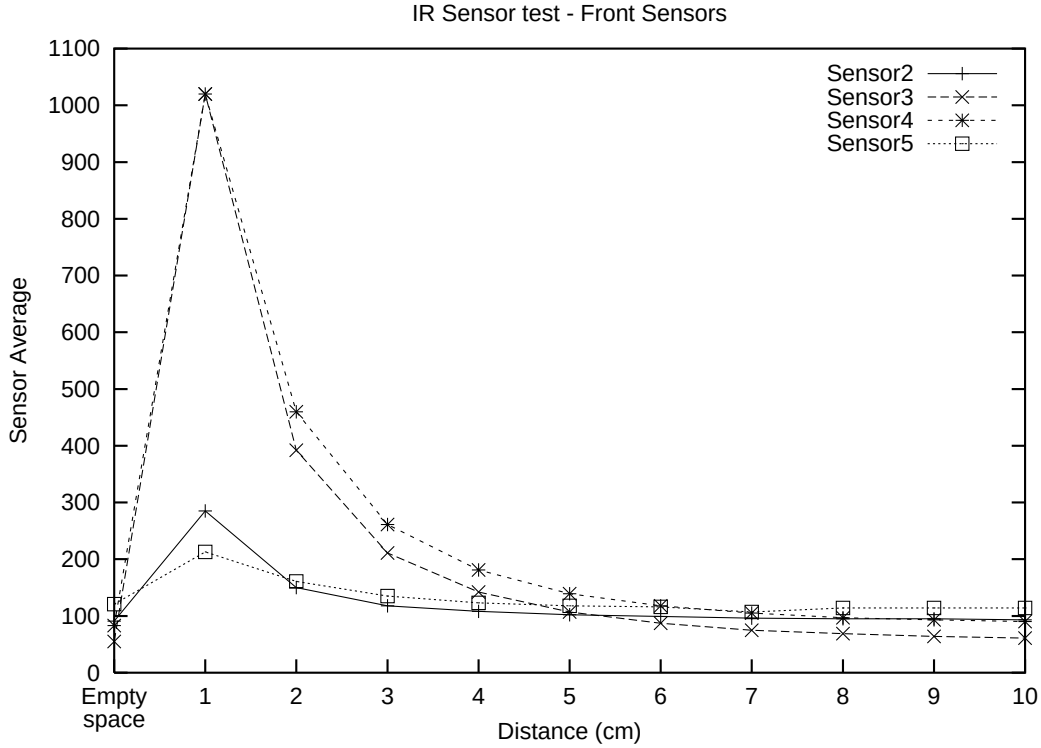


Figure 9: White paper detection test

For this test, surface is placed in parallel with the robot's side sensors.

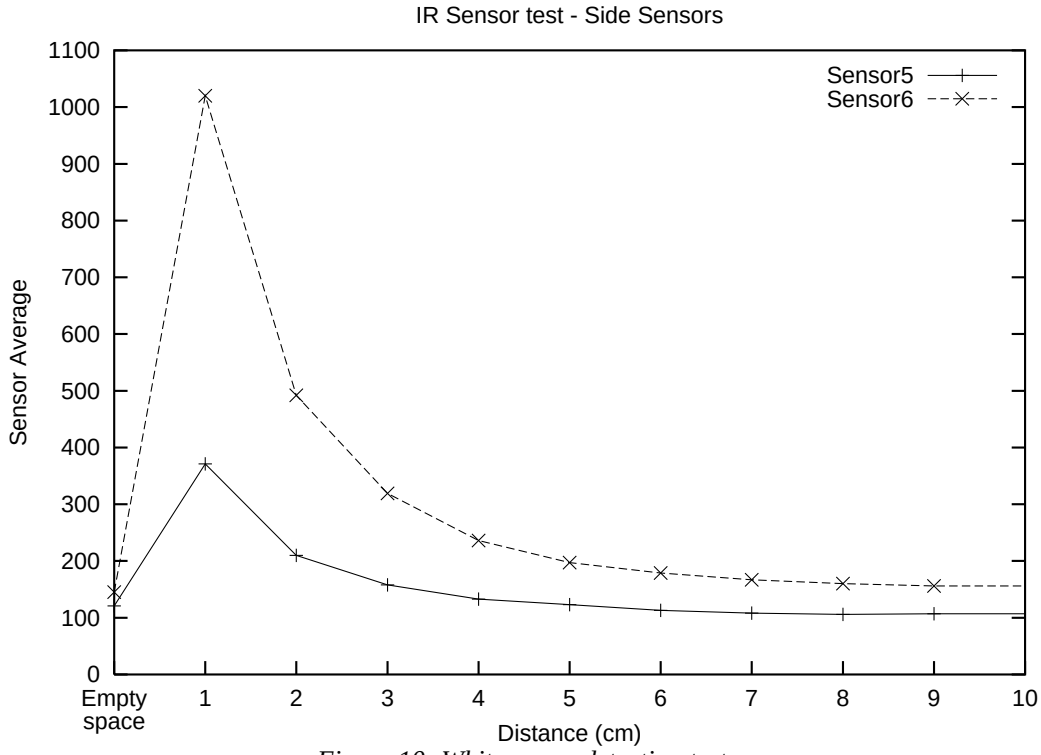


Figure 10: White paper detection test

During other robots detection test, the two Khepera are placed facing each other at the given distance, under the same lighting conditions as before.

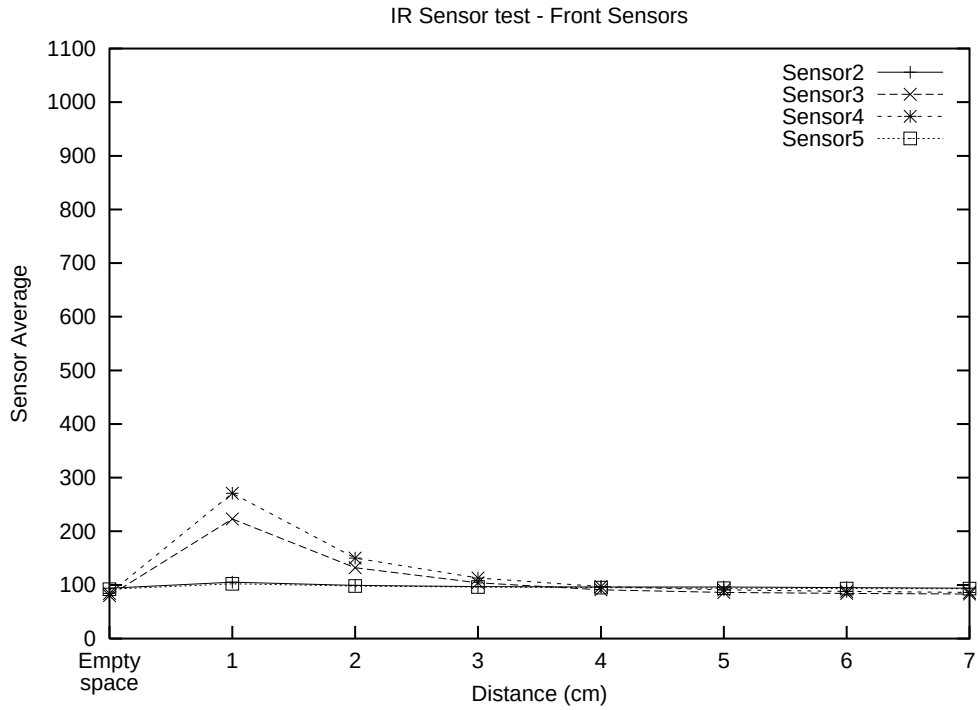


Figure 11: Other robot detection test

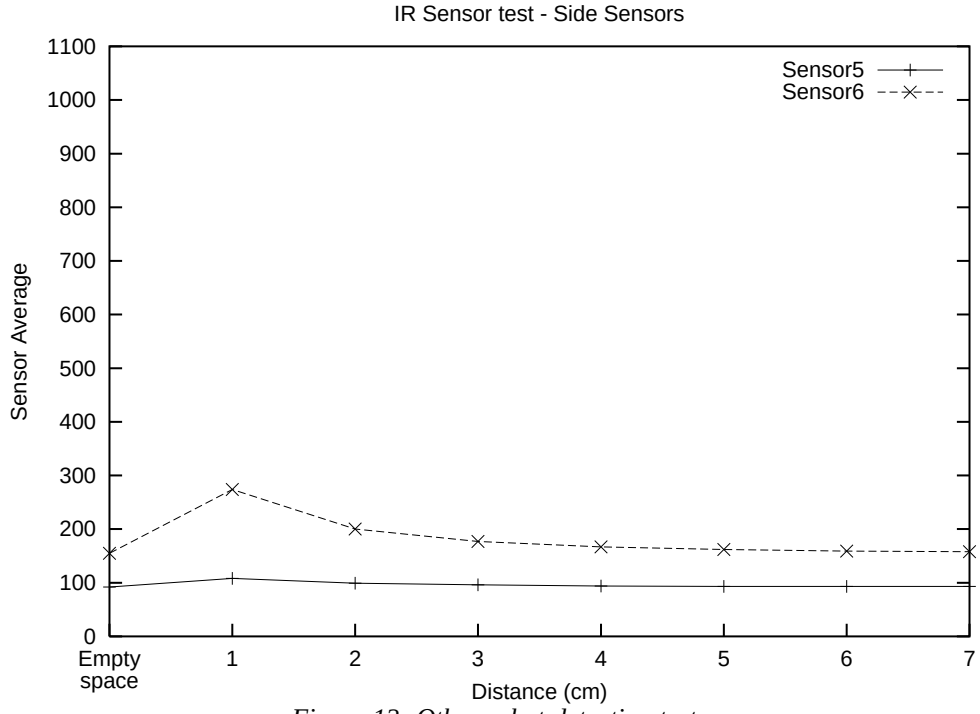


Figure 12: Other robot detection test

3.1.9 Batteries

The Khepera is equipped with four Nickel Metal Hydride batteries providing 250mAh each. Using its embedded power, the robot is able to run completely autonomously during approximatively one hour, running with a basic configuration. When additional equipment are used, the autonomy is reduced as Khepera's turrets usually rely on Khepera's batteries as a power source.

As any batteries, their performances vary over time. When using the Khepera for the first time, autonomy might be shorter than usual. After a couple of full charge and discharge, the batteries should reach their optimum capacity. Carefully respecting the full charge and discharge cycle will enhance batteries lifetime and performances.

There is no specific power management system on the Khepera. When the batteries voltage falls under 4V, the robot's processor will stop working normally. From this moment the robot's behavior is not predictable but it might still be able to move for a few seconds. Users can implement their own software power management system to handle processor shutdown properly.

3.2 Cables and accessories

The S cable (2 m long, with a 6 pins connector) allows the connection between the robot and the interface/charger module to support the communication to the host computer (see section 5.2: Configuration for Robot-Computer communication).

The recharging cable (0.5 m long, with a 4 pins connector) allows the connection between the robot and the interface/charger module to recharge the robot.

4 new tires are also included into the package.

3.3 Power Supply

If an external power source is required or during batteries charge, power is supplied through the interface module. This module should be connected to the main wall socket using the provided AC/DC adaptor only. Depending on the robot's usage, connections between the robot and the interface might change. Please refer to section 5 for detailed description of connections.

If the batteries are switched OFF, the robot will use external power. Do not switch ON the robot if is is connected to an external power source.

SAFETY PRECAUTION: The power supply must be connected to the wall socket after all other connections are already made.

3.4 Interface and Charger Module

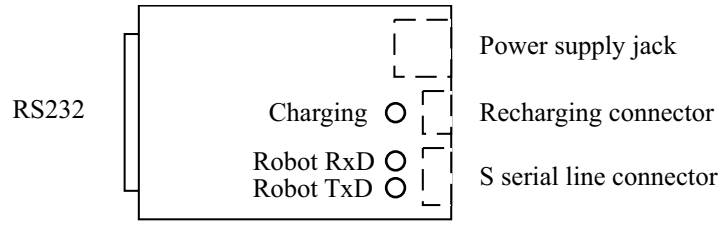


Figure 13: The interface - charger module

This module is used as a interface for serial communication between the robot and a host computer, as an external power source if needed, or as a battery charger. Power supply is to be provided using the AC/DC adaptor, connected to the power supply jack.

This module is providing:

- **The battery charger**: a four pins connector allows the connection to the Khepera for battery charge. **When charging, the robot must be disconnected from all other systems and switched OFF. Avoid to recharge full batteries, this can cause damages!** It would be optimal to completely discharge the batteries (leave the robot switched on for a while) before recharging them! During the charging period, a yellow led indicates the activity. The charging time for an empty battery is about 60 minutes. See section 5.1, for more details.
- **The S - RS232 interface**: this interface allows the connection between the robot (S serial line) and a host computer (over a RS232 port). Two connectors are available: a standard female DB25 (the interface module is a DCE) for the RS232 link toward the host computer and a S six pins connector for the link toward the robot.

The S cable can also be used as a power supply to the robot if the battery switch is OFF. See section 5.2, for more detail on this working configuration.

3.5 K-Team Support CD

This disk contains most of the standard support material and software tools for K-Team products. The disk can be accessed from the following systems:

- MAC OS systems
- Windows 95/98/ME/NT systems
- All UNIX and Linux systems

The Khepera's documentation is usually valid for any system. Specific software tools, cross compiler, LabVIEW® and Matlab® files, are available for MAC OS, Windows, Solaris, and Linux.

LabVIEW® is a product from *National Instruments* and Matlab® is a product from *MathWorks*. These products are NOT included within the package and must be purchased separately.

4 UNPACKING TEST

After unpacking it is important to test the robot's functionality. A test that uses most of the possible functionality is available with the running mode 0: a Braitenberg vehicle (see section 3.1.3 on running modes). Please read the following instruction to before running a demo mode test:

- Put the robot on a flat surface that is safe for the robot. Water, table's edge or metallic objects may cause damage to the robot and must be considered as dangerous. The robot will move rather quickly and cover long distances in a short delay, the safest way to run a demo is to have to robot moving within a limited “arena”.. The robot is normally charged when delivered.
- Set the encoding wheel to position 0, as described in section 3.1.3.
- Switch the robot ON and let it moving freely on the chosen surface.

The robot should move straight forward and avoid obstacles. Some obstacles, such as black surfaces, are particularly hard to detect for IR sensors, but most of them should be avoided nicely. If the robot does not operate properly, check the three steps above, recharge the robot and retry. If the robot does not move, does not correctly avoid obstacles, or is not operating properly in any other way, please contact your local Khepera dealer.

5 CONNECTIONS

There are two standard configurations one is used to charge the robot's batteries and the other is used for any serial communication between the robot and a host computer. The S serial line can also provide an external power supply to the robot.

5.1 Configuration for batteries charge

Warning: Batteries should be fully discharged before charging to improve performances. DO NOT start charging fully charged batteries as this can cause damage to the batteries themselves and to the Khepera.

To charge robot's batteries, make sure the following connections are correct:

- Between the robot and the interface/charger module using the charging cable (4 pins).
- Warning: the robot battery switch must be on the OFF position.

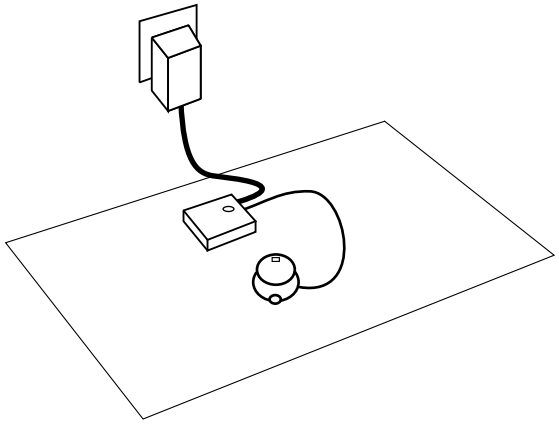


Figure 14: Connections to charge robot's batteries

- Between the interface/charger module and the wall socket using the AC/DC adaptor to the power supply jack.
- Plug the power supply to the wall socket only after all these connections are set.

Before charging batteries, the charger performs a connection checking and measurement, these operations can last up to 10 minutes when batteries are hot (after a long use) or too discharged. When charging, the “charging” indication led is ON. If the led does not switch ON after 10 minutes, check all the connections then unplug and re-plug the power supply.

The led is switched OFF at the end of the charging process. The charging time for an empty battery is about 60 minutes. At this moment the power supply can be unplugged and the charger cable removed. When charging, the battery can be as hot as 50°C. This is normal.

5.2 Configuration for Robot-Computer Communication

This configuration allows the communication between the robot and a host computer through a serial link. The host computer is linked to the interface module using a standard RS232 line, while the interface module converts RS232 signal into S serial signal to communicate with the robot.

To use the serial communication mode, please make sure the following connections are correct:

- The robot must be connected to the interface/charger module using the S serial cable. This cable is also used as a power source if needed. This external power supply is used when the general battery switch is OFF. If the switch is ON, the robot uses its own batteries as a power supply.
- The interface module must be connected to the host computer using a standard RS232 cable. As many different connectors are used, depending on the computer, this cable is not included with the package. You can easily purchase this cable from your host computer dealer. If your host computer serial port is a DB25 male connector you can directly plug the interface module without using a cable.

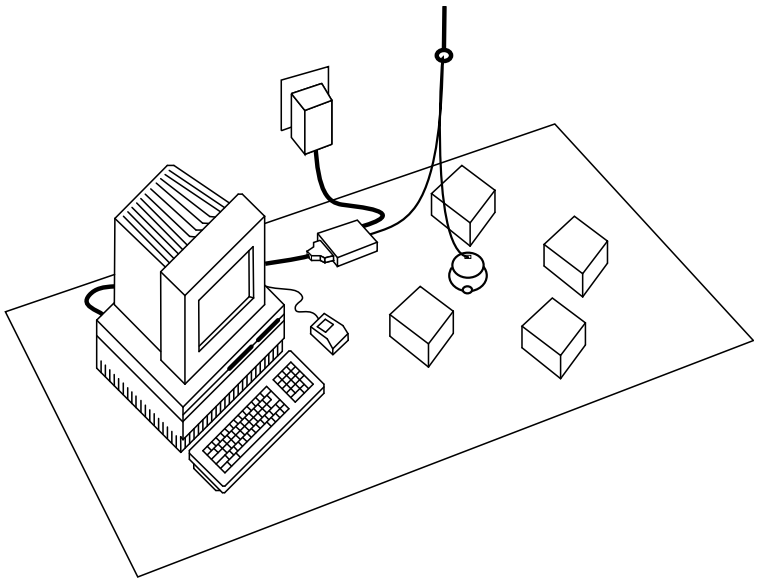


Figure 15: Configuration for communication between the Robot and a host computer

- The interface/charger module must be connected to the AC/DC adaptor using power supply jack.
- Set the encoding wheel to the desired running mode (see section 3.1.3).
- Plug the AC/DC adaptor to the wall socket.

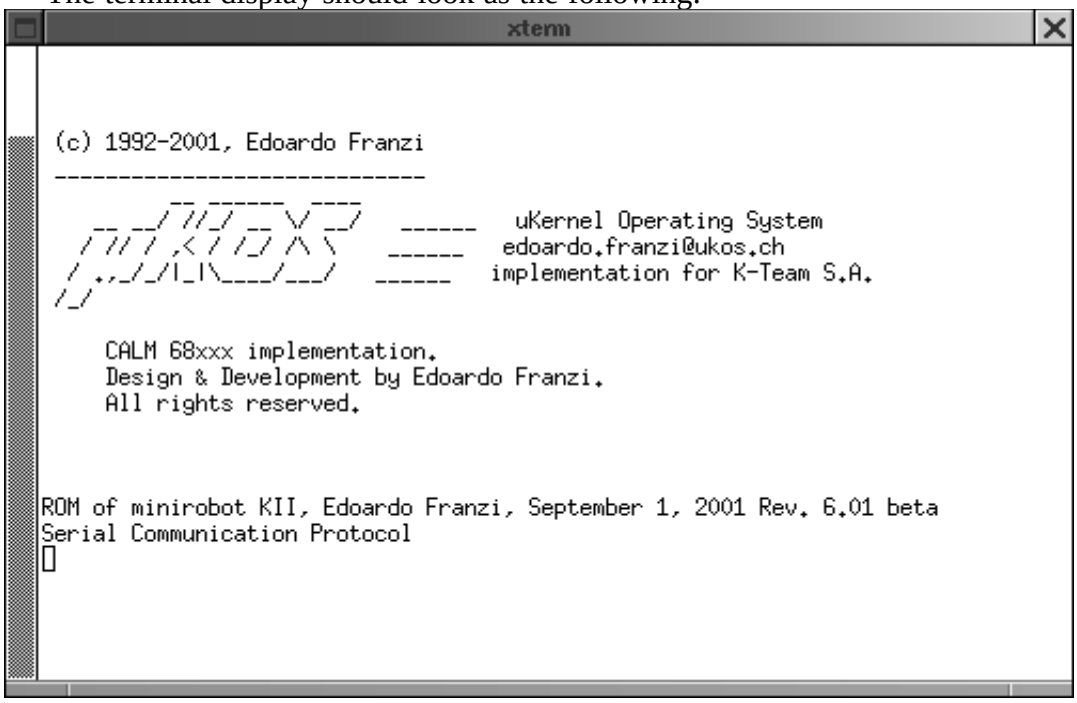
6 SERIAL COMMUNICATION MODE

6.1 Testing the serial link

Before any further operations, the serial link between the host computer and the robot should be tested. Please read the following instructions to test the serial communication mode:

- Make sure all connections are correct (see section 5 for details).
- The robot should be placed on a flat and safe surface. The battery switch must be OFF.
- Select running mode 1 using the encoding wheel as described in section 3.1.
- A terminal emulator should be running on the host computer. Make sure the terminal is connected to the correct serial port. **The terminal configuration must be set to 9600 Baud, 8 bit, 1 start bit, 2 stop bit, no parity.**
- Plug in the power supply, or, if already connected, **reset the robot** using the reset switch.

The terminal display should look as the following:



The transmit data (Robot TxD on the interface module) green led should blink after a reset. If the robot does not respond as indicated (the green led does not blink after reset), check the above steps and retry. If the green led blinks but the displayed messages do not match the screenshot, check the serial port configuration and terminal configuration, as well as the serial connection between interface/charger module and host computer.

6.2 Serial communication protocol

The serial communication protocol is designed to control all Khepera's functions using a RS232 serial line. Any serial communication running mode can be used (see section 3.1.3: Running modes). The connection configuration is described in section 5 of this manual. The serial line configuration (baudrate as well as data, start, stop and parity bits) for the host computer must match the robot's configuration according to the running mode. The communication speed is the important parameter to check as all other parameters are always 8 bit, 1 start bit, 2 stop bit and no parity.

The host computer and the Khepera robot are communicating with ASCII messages. Each single interaction is composed by:

- A command, sent from the host computer to the Khepera robot and followed by a carriage return or a line feed.
- When needed, a response, sent by the Khepera to the host computer.

During the entire communication, the host computer is acting as a master and the Khepera as slave. All communications are initiated by the master.

Two different types of interactions are possible. The first set of interactions is used to set up the robot configuration from the host computer (set up serial line, changing controllers configuration,...), the second set of interactions is used to control the robot (controlling motors, reading sensors value,...).

Some other tools can also be used from the terminal, these are described in next section.

6.2.1 Tools

Here is the description of some basic tools:

"run"	Starts a function stored in ROM. It has to be followed by the function name. Available functions can be listed using the *list* tool, the functions identification string begins with "FU". Some of the functions correspond to running modes presented in the section 3.1.3. Any mode, such as the demo mode, can be started using the *run* tool. (typing "run demo" and return)
"serial"	Sets the serial channel to the given speed in baud. Typing "serial 38400" sets the communication speed to 38400 baud.
"help"	Display any available message for a ROM module. An optional, module name can be set so that help on a particular module is displayed. Help message for the list tool is displayed by typing"help list".
"list"	List of all the tools, functions, protocol commands and other modules available in ROM is displayed. An ID, name, description and version is associated with each listed item. The ID is a four letters string. The first two letters define the module's family: "TA":tasks running on Khepera "FU " functions that can be executed using the *run* command. "PR" protocol commands "TO" tools such as this list "BI" BIOS components.
"k-team"	Gives a short description of the K-Team founders.
"net"	Gives an information about intelligent extension turrets installed on the robot. A name, ID (to be used when addressing a turret, see also the command 'T' in appendix), description and revision is associated with each listed item.
"memory"	Information about system's memory usage is displayed
"restart"	The robot is reset (equivalent to hardware reset).
"process"	All processes running on Khepera are listed in parallel with the serial communication protocol management.
"sfill"	Motorola S format loader is started. This loader does not start the execution of the loaded code when completed. To download and execute code, please use the sloader function, started by the command "run sloader".

6.2.2 The control protocol

The control protocol is used to send control messages to the robot. A set of commands is available as detailed in appendix. As the Khepera may need to send an answer message to the host, ASCII messages are used to communicate between the two. Each interaction is composed by:

- A command, beginning with one or two ASCII capital letters and followed, if necessary, by numerical or literal parameters separated by a comma and terminated by a carriage return or a line feed, sent by the host computer to

the Khepera robot.

- A response, beginning with the same one or two ASCII letters of the command but in lower-case and followed, if necessary, by numerical or literal parameters separated by a comma and terminated by a carriage return and a line feed, sent by the Khepera to the host computer.

During the entire communication, the host computer is acting as a master and the Khepera as a slave. All communications are initiated by the master.

Please refer to appendix for a complete description of available commands.

6.2.3 Testing a simple interaction

Testing some basic commands is the best method to understand the serial protocol and tools available on the Khepera. Using a properly configured serial link between the robot and a computer, please follow the instructions bellow:

- Type the capital letter **B** followed by a carriage return or a line feed.
- The robot must respond with **b** followed by an indication of the version of software running on the robot and terminated by a line feed.
- Type the capital letter **N** followed by a carriage return or a line feed.
- The robot must respond with **n** followed by 8 numbers separated by a comma and terminated by a line feed. These numbers are the values of the robot proximity sensors presented in section 3.1.6.
- Retry the same command (N) putting some obstacles in front of the robot. The response must change.
- Type the protocol command **D,5,-5** followed by a carriage return or a line feed.
- The robot must start turning on place and respond with **d** and a line feed.
- To stop the robot type the protocol command **D,0,0** followed by a carriage return or a line feed.
- Type the protocol command **H** followed by a carriage return or a line feed.
- The robot must respond with **h** followed by 2 numbers separated by a comma and terminated by a line feed. These numbers are the values of the position counters of each wheel.
- Type the protocol command **G,0,0** followed by a carriage return or a line feed.
- This command set the position counters to the 2 values given as parameters. The answer is composed by a **g** and a line feed.
- Retry the protocol command **H** to verify that the G command has been executed.
- Type the protocol command **C,1000,1000** followed by a carriage return or a line feed.
- The robot respond with **c** and goes forward 80 mm.
- Retry the protocol command **H** to verify the final position.
- Try other commands following the description given in Appendix A.

- We continue testing some tools:
  - Type the command **help** followed by a carriage return or a line feed.
  - The robot must respond with the list of all tools available.
  - Type the command **help serial** followed by a carriage return or a line feed.
  - The robot must respond with the description of the “serial” tool.
  - Type the command **help D** followed by a carriage return or a line feed.
  - The robot must respond with the description of the “D” protocol command.
  - Type the command **list** followed by a carriage return or a line feed.
  - The robot must respond with the list of all software modules present in the ROM. In addition to the “tools” (characterized by a “TOXX” ID) and the “protocol” commands (characterized by a “PRXX” ID) you can find on the list “functions” (characterized by a “FUXX” ID) and BIOS modules (characterized by a “BIXX” ID). On every module you can have an help message.

7 S LOADER MODE

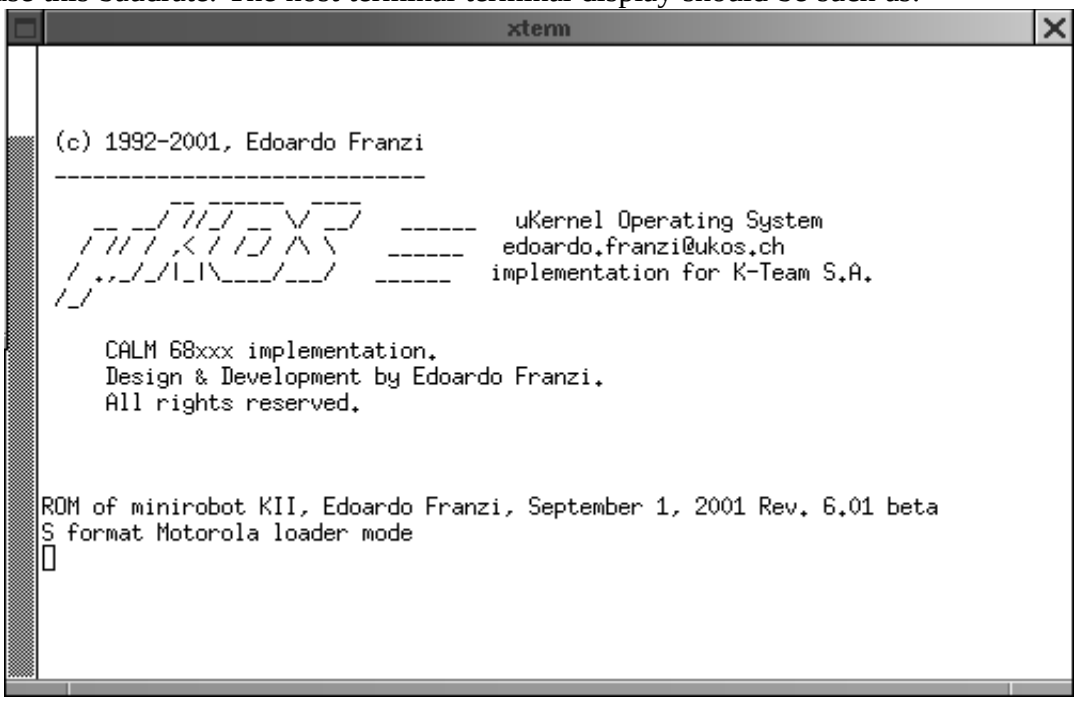
The S loader mode is used to load in RAM and execute any user application. Instructions to obtain a Motorola 68000 executable file for the Khepera are available in a separated documentation. Such a file is, of course, required to execute an user application.

7.1 Serial link configuration

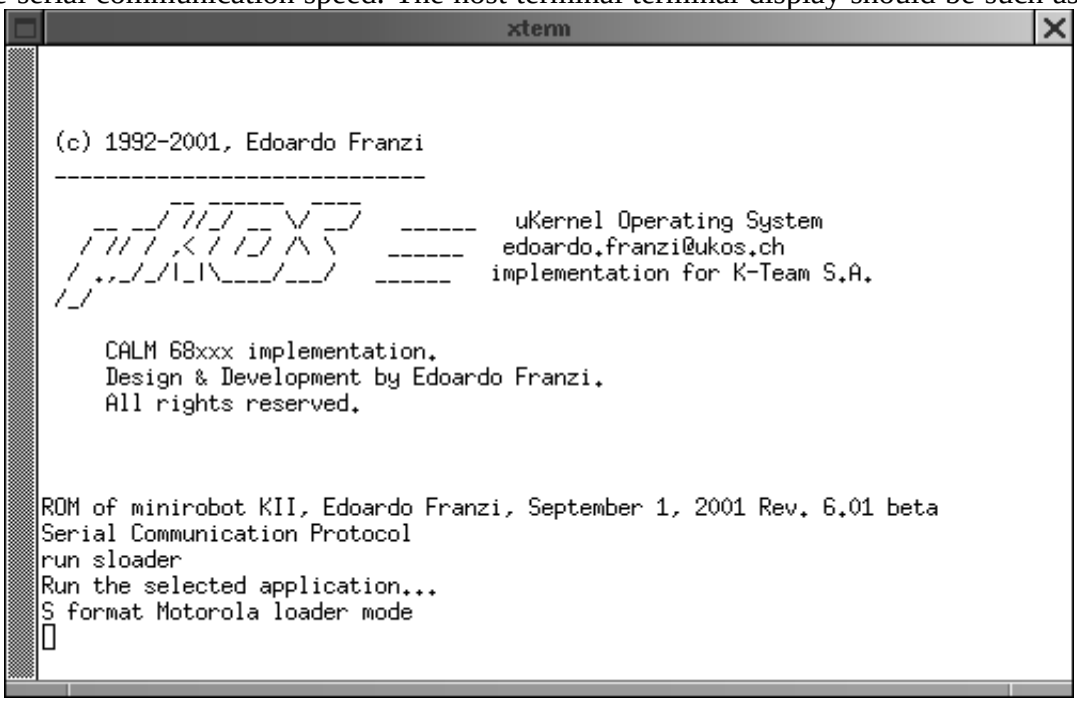
The serial link has to be properly configured to enable a robot computer communication. Please refer to section 5 for a detailed description of necessary connections and configuration. When booting in S loader mode, the serial communication speed is set at boot time, and the host terminal speed must be set according to the chosen mode. Other parameters are set to the usual value.

7.2 Starting the S loader

Two different methods are available to start the S loader. First, when booting the Khepera in one of the S loader modes (5,6 or A), the loader is automatically started. The communication speed is set according to the chosen mode, and the entire loading process will use this baudrate. The host terminal terminal display should be such as:



Second, the S loader can be started using the serial communication protocol. The command “run sloader” is used to start the loader, and the transfer speed will stay the same as the serial communication speed. The host terminal terminal display should be such as:



Important Note: Running the S loader using a 115200 bit/s communication speed may cause loading failures. The maximum recommended baudrate is 57600 bit/s until an upgraded BIOS is available.

7.3 Loading an application file

Once started, the S loader simply waits for an executable file to be transferred through the serial line. Depending on your system and on the terminal emulator you are using, several methods are available to send a file. The most common is to use a “send file” command from the terminal emulator.

As soon as the loading process is initiated, one of the Khepera's led indicator is switched on. The indicator should stay on during the entire loading process and turned off when the download is completed. The downloaded application is executed as soon as the transfer is achieved and the following message is displayed:

S: download terminated

In case of problem loading an application, check all the connections and configuration and try to use the serial communication protocol using the same baudrate.

8 USER APPLICATION MODE

This mode is used for a completely autonomous execution of an user application. The application has to be uploaded in the robot's non volatile memory, using a serial link, and can be executed at boot time.

8.1 Uploading an application in non volatile memory

Before proceeding, a serial communication link must be properly set up (see section 5: Connections) between the robot and your computer. The communication speed is setting the file transfer rate. Then the instructions bellow have to be followed:

- Use the “sfill” command to start the loader. The following message should be displayed:
  “S format Motorola loader mode”
- Send the application file to the serial line, using your favorite method. As soon as the transfer is in progress, an led indicator is switched on, and will stay on until the transfer is completed. The application is transferred into the robot's RAM, and still needs to be flashed. When completed, the following message should be displayed:
  “S: download terminated”
- Erase the robot's non volatile memory using the “flash E” command. When completed the following message should be displayed:
  “FLASH user segments erased”
- Write the application into the flash memory using the “flash W” command. When completed the following message should be displayed:
  “FLASH user segments written”

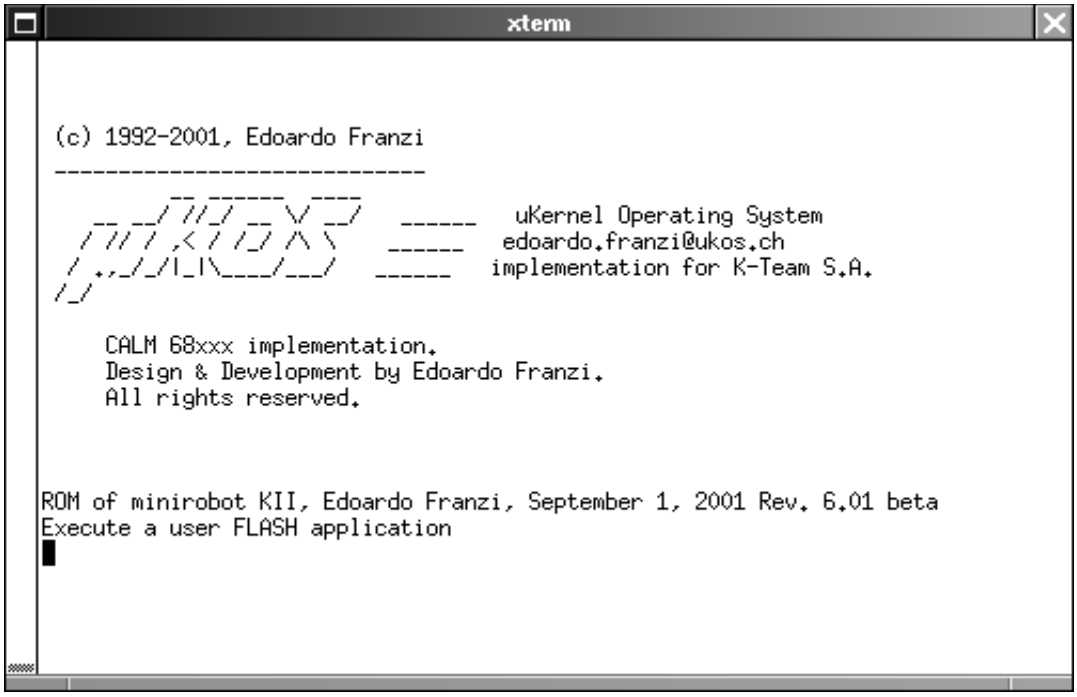
The robot is now able to execute autonomously the loaded application. The only required intervention is to switch it on.

8.2 Executing user application

An application stored in the robot's non volatile memory will stay there until it is erased, using the “flash E” command, or until the memory chip is damaged. This application can be executed at any time using one the two following methods:

- If a serial communication link is already up and running between the robot and your host computer, simply use the command “run user-flash” to execute the user flash segment. The following messages should be displayed:
  “Run the selected application...”
  “Execute a user FLASH application”

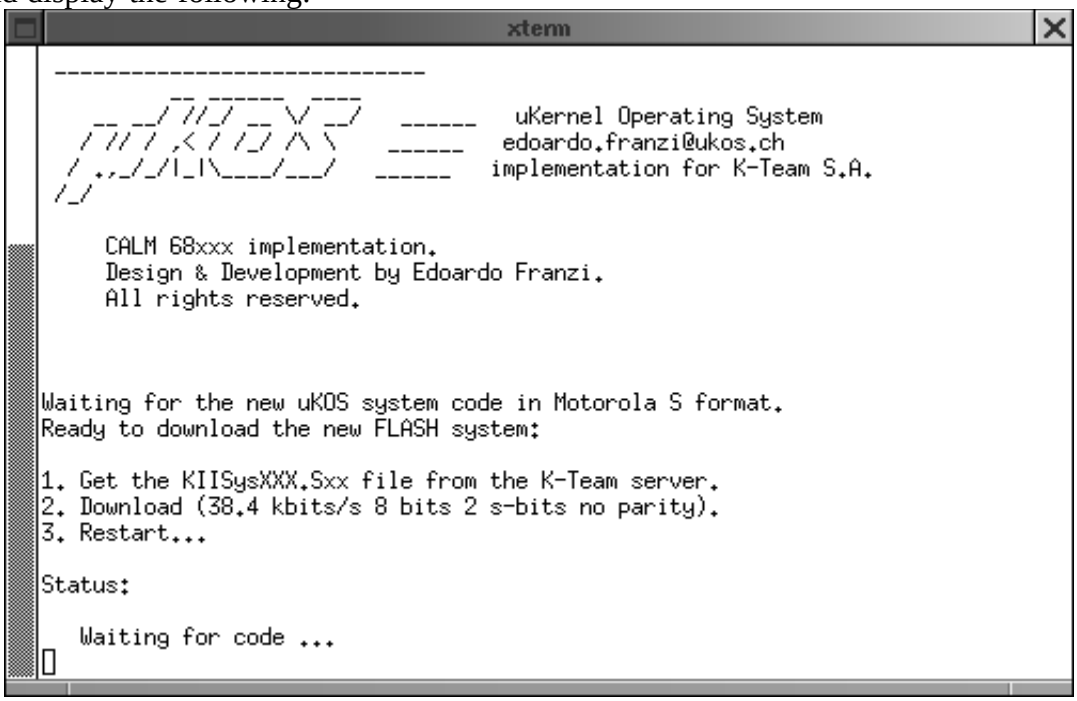
- For a completely autonomous execution, set the encoding wheel position to one of the user application mode (4 or B) and reset the robot. The user flash segment is automatically executed at boot time and the terminal display should look as:



9 UKOS UPGRADE MODE

The Khepera operating system can be upgraded using the serial link. The Khepera is provided using the last available uKos version, but new upgraded releases are available for free download from K-Team website (www.k-team.com). For latest information on new uKos versions, K-Team tools, documentation and other K-Team products, please subscribe to Knews, K-Team public mailing list.

To upgrade the Khepera's operating system, the serial communication link must be properly set up between the robot and your host computer (using 38400 bits/s speed). Set the encoding wheel position to the uKos upgrade mode (F) and reset the robot. The terminal should display the following:



Next, the new uKos binary file must be sent to the serial line, using a send file command. Uploading the new system might last for a few minutes, and the robot's led indicators stay off. When the loading process is completed, the following messages are displayed while the corresponding operations are performed:

“Downloading OK ...”
“Erasing the FLASH ...”
“Programming the FLASH ...”
“Programming terminate; restart the system.”

If the above operations are successful, a simple reset will boot using the upgraded operating system. Check for release notes and documentation on K-Team website for all information concerning new features and improvements.

10 USING LABVIEW®

This chapter is introducing the LabVIEW® environment for Khepera usage. As LabVIEW® is a *National Instruments* product (www.ni.com), this tool can be purchased separately. To this end, the examples are presented in an increasing order of complexity. Our advice is to follow the chronological order of presentation. Please refer to the LabVIEW manuals for more information about this software.

The following examples and the files distributed with this product are based on LabVIEW® version 5.

LabVIEW® runs on your PC, Macintosh® or SUN® workstations, and can control the functionality of the Khepera robot using the serial communication protocol described in section 6.

10.1 Hardware Configuration

Set your environment as illustrated in section 5.2: Configuration for Robot-Computer communication. The encoding wheel should be set to boot the Khepera using mode 2.

10.2 Setting up the Serial Link

To enable the exchange of information between your computer and the robot, you have to configure the serial link of your host computer, according to the setting chosen on the Khepera robot.

Be sure that the connection cable is connected at both ends (Khepera and interface), that the robot is powered (power adaptor), then start LabVIEW® and open the Set-up virtual instrument (called “VI”) present in your floppy disk.

The panel illustrated in figure 22 should appear.

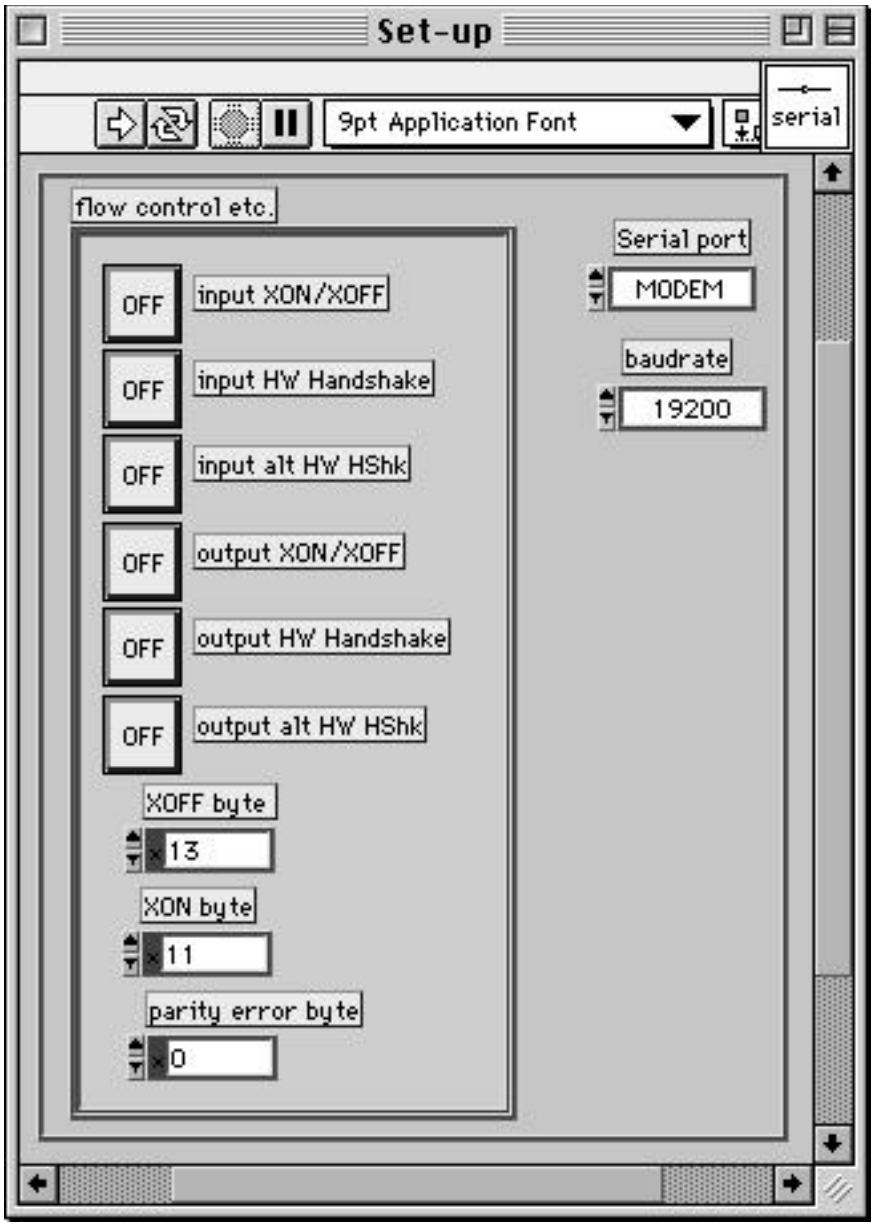


Figure 16: Serial link configuration panel

Now, select the serial port on which the robot is connected. This selection depend on which port you use and the type of computer you have. This choice must be made for every module that you will use.

Then click once on the run arrow  at the top of the window.

A stop icon  appears for a few seconds, after what the front panel returns to its initial state.

That's all! The serial link with Khepera is set to 19200 baud. It will remain so until you quit LabVIEW®.

10.3 Using Motors

We will now control the displacement of the robot. Be sure that the serial link has been correctly installed, then open the Motors VI present on the floppy disk. Now your screen displays the following panel:

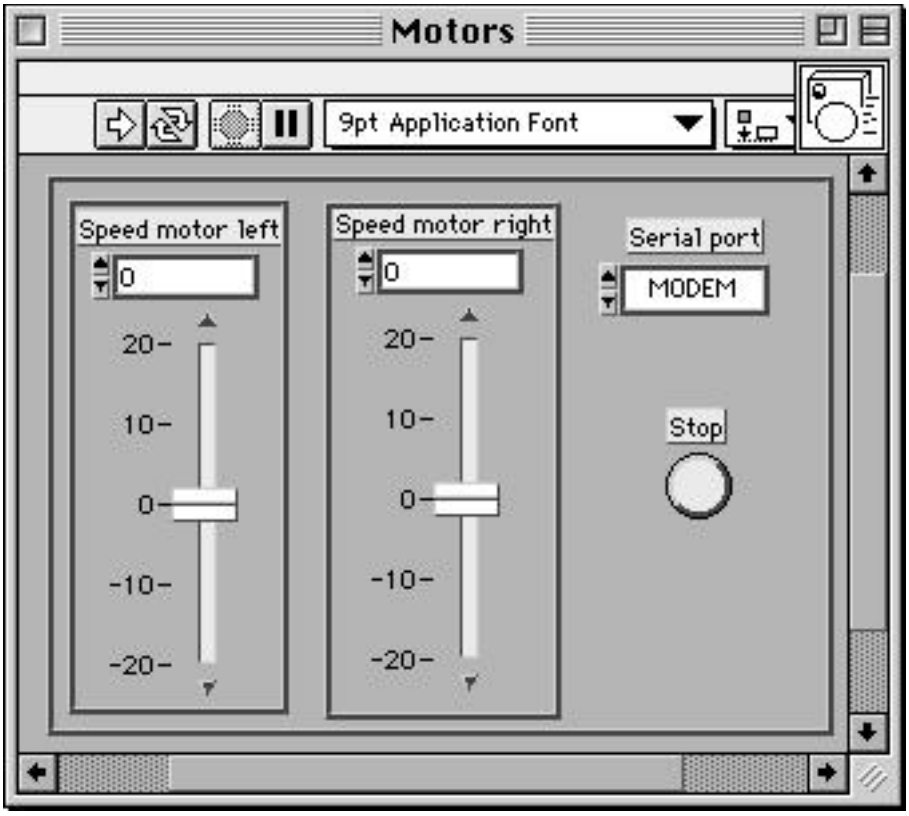


Figure 17: Motor panel - sliders controlling each wheel speed.

Before moving the robot, you must learn how to stop it. Different means are available. Starting from the most efficient:

- Press the reset button on the robot once.
- Put the value 0 for each speed using the sliders. Don't forget that these new values will only be taken into account at the next execution (i.e., click on the arrow).
- Click on the button labeled Stop. You have also to click on the arrow again so that the robot takes your last decision into account. Before trying to give another values to the motors, click again on the button (to de-select this option). This last option is the best way to stop the robot.

You control directly each one of the motors by simply putting the desired speed values in the corresponding slider. This can be made moving the slider or entering the desired speed in the digital display placed between the sliders and their names.

Possible values are constrained (only on the sliders) between -20 and +20 so to take care of the mechanics. To transmit your order to the robot, just click once on the arrow. You can change the values and click on the arrow again to validate your choice. You see that the

robot continues moving at the same speed until new values are send.

If you are getting bored with clicking on the arrow, try one click on the double arrow  . Click on the stop icon to stop the execution.

The robot has two incremental encoders on the wheels. Using these sensors it is possible to measure the displacement and the speed of each wheel at every moment. The VI "Get_position" ask for the content of the increment counter, which represent the displacement of the wheel. The unit of displacement correspond to 0.08mm.

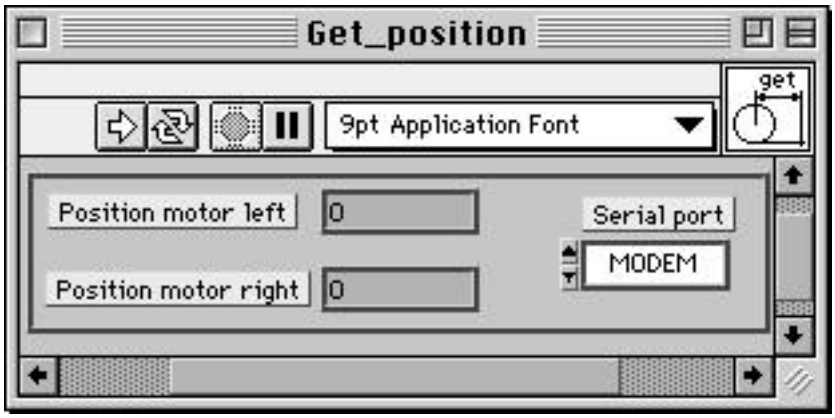


Figure 18: Get_position panel - 2 indicators showing each wheel position

To test the functionality of this module just click on the double arrow to start the recurrent running mode. At this moment the VI will show you the actual position of each wheel. To change the position of the wheel use the "Motors" VI as described above, and observe the result on the "Get_position" VI. To set the position counter to a given value you can use the "Set_position" VI.

The "Get_speed" VI display the speed of each wheel. This value is computed on the robot, based on the information of the position and the time.

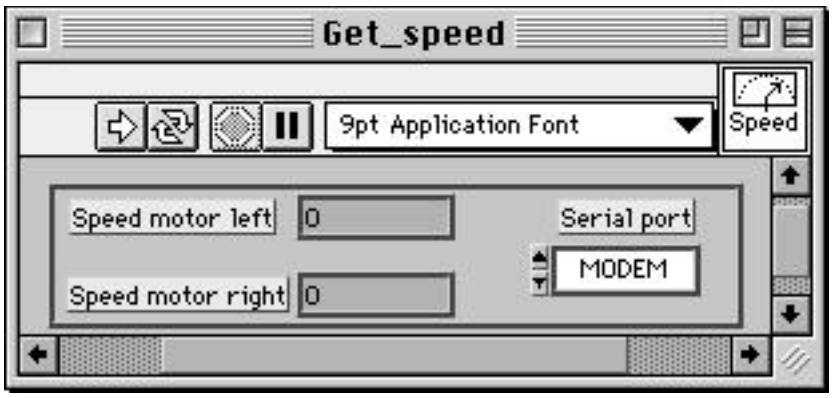


Figure 19: Get_speed panel - 2 indicators showing each wheel speed

To test the functionality of this module just click on the double arrow to start the recurrent running mode. At this moment the VI will show you the actual speed of each motor. To change the speed of the motors use the "Motors" VI as described above. You can also try to set the motor speed to 5 using the "Motors" VI, then slow down the wheels with your fingers and look to the result on the "Get_speed" VI.

It is also possible to give to the robot a position to reach, expressed using the position of the two wheels as described above. The position given as target will be reached using a trapezoidal speed profile (as described in section 3.1.5 of this manual). The target position can be given to the robot with the “Control_position” VI, illustrated in figure 20. Also here the commands can be made moving the slider or entering the desired position in the digital display placed between the sliders and their names.

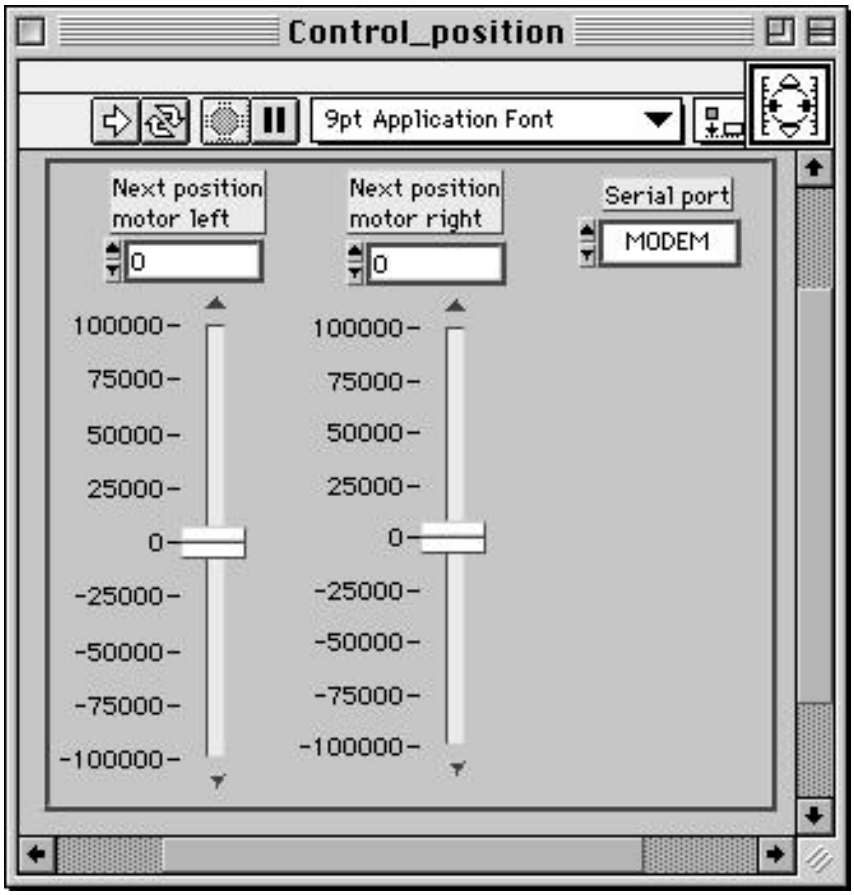


Figure 20: Control_position panel - 2 sliders controlling target positions for each wheel

As described in section 3.1.5 of this manual, every displacement in position control mode will be made following a precise speed profile. The parameters of this speed profile can be configured using the VI “Conf_pos_param”, showed in figure 21. For each motor you can set acceleration and maximal speed. Deceleration will be set identical to acceleration, figure 21 shows default values.

To test the position control, please start setting to 0 the two position counters of the two wheels, using the “Set_position” VI or resetting the robot. At this moment you can try the “Control_position” VI setting a distance of 1000 on both sliders and running the VI once, clicking on the run arrow. The robot should move forward showing an acceleration, a fixed speed and then a deceleration. Test other movements keeping both left and right displacements identical to move on a line.

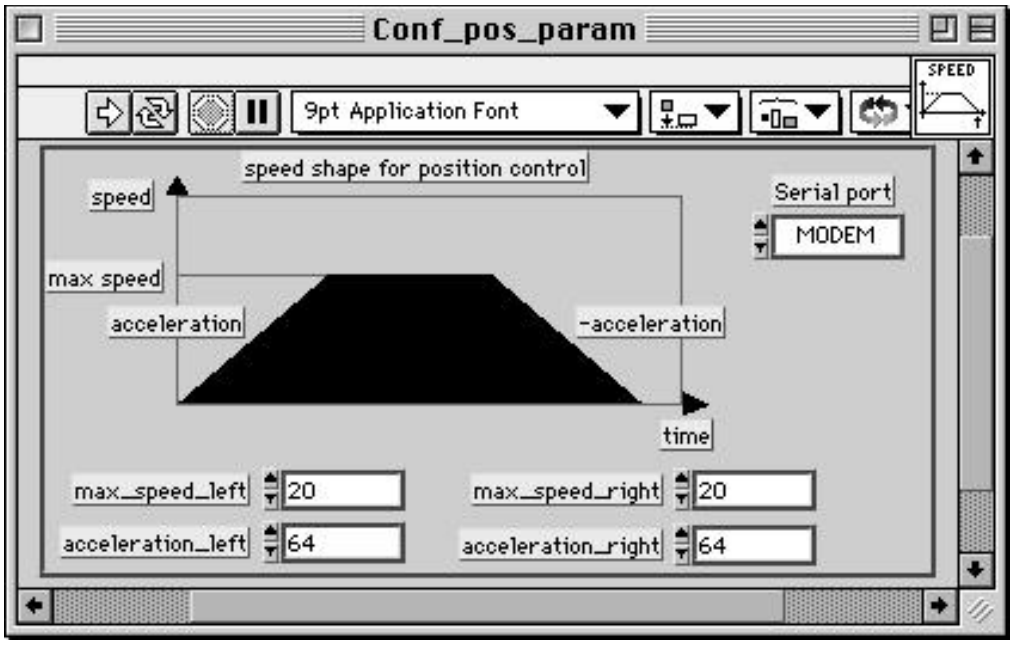


Figure 21: Conf_pos_param panel - Speed profile parameters configuration

To start other kinds of trajectory, bring back the robot to the 0 position or use the "Set_position" VI to reset the counter. Then use the "Conf_pos_param" VI to set the left maximal speed to 10, the left acceleration to 32, and keeping the values of the right trajectory to the default value indicated in figure 21. Run the VI once to make these values effective. Then set, on the "Control_position" VI, the goal position of the left wheel to 1000 and the goal position of the right wheel to 2000. Run the VI once and observe the trajectory of the robot. The robot should make a circular trajectory.

10.4 Using Sensors

In its basic version, Khepera has eight infra-red sensors, as described in section 3.1.6. You will now easily understand their characteristics. Be sure of the set-up of the serial link, then open the Sensors VI that you have on the floppy disk. You should see on your screen the panel illustrated in figure 22.

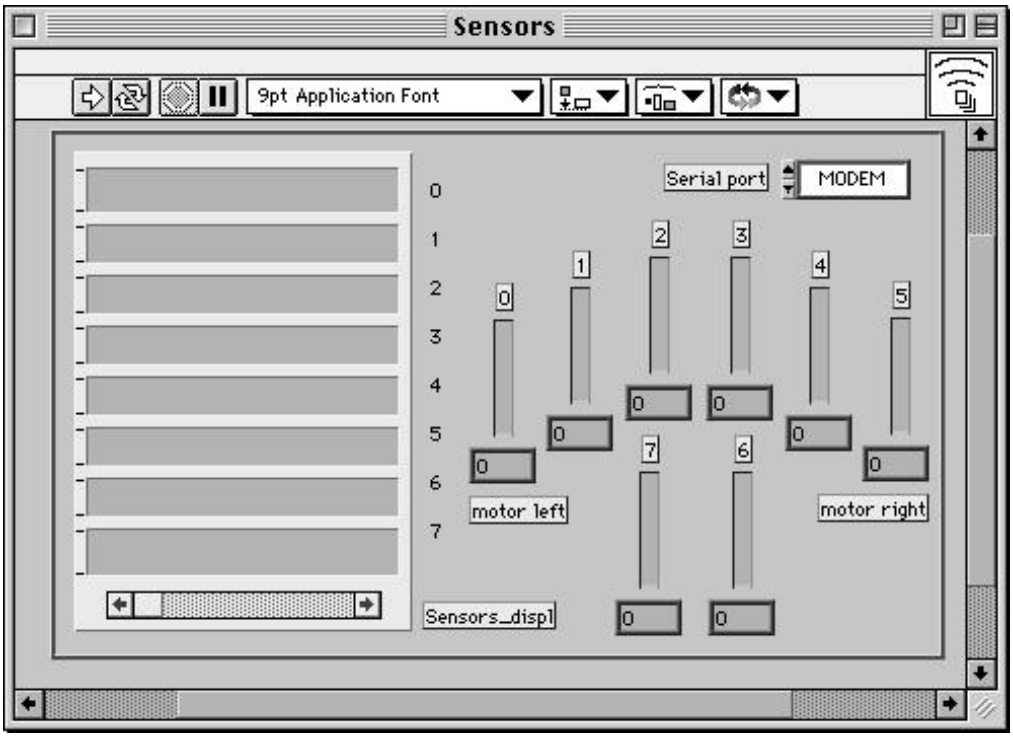


Figure 22: Sensors panel - 8 gauges displaying the infra red values.

Each proximity sensor value is displayed as a gauge. The gauges are placed on the panel like the corresponding sensors on the robot. The exact value received is written underneath. Values are between 0 and 1023. Start the acquisition as before by clicking on the double arrow (to stop the execution, click on the stop icon that will appear). Now you are free to test the response of the sensors. In particular, some materials reflect better than others the infra-red light emitted by the sensors. You are also able to state difference in the individual response of the proximity sensors.

The graph on the left of the panel shows the values for each sensor against time.

10.5 Braintenberg's Vehicle

At this stage, you have a good understanding of motors and sensors functionality. In this section we will combine these two modules, as sub-VIs of a main sensory-motor loop VI. Let's open both panels (Motors and Sensors), but do not start them. This way, you will be able to watch, at the same time, data issued from the sensors and those send to the motors. Open the instrument called KheBraitenberg3c. You should see on your screen the panel illustrated in figure 23.

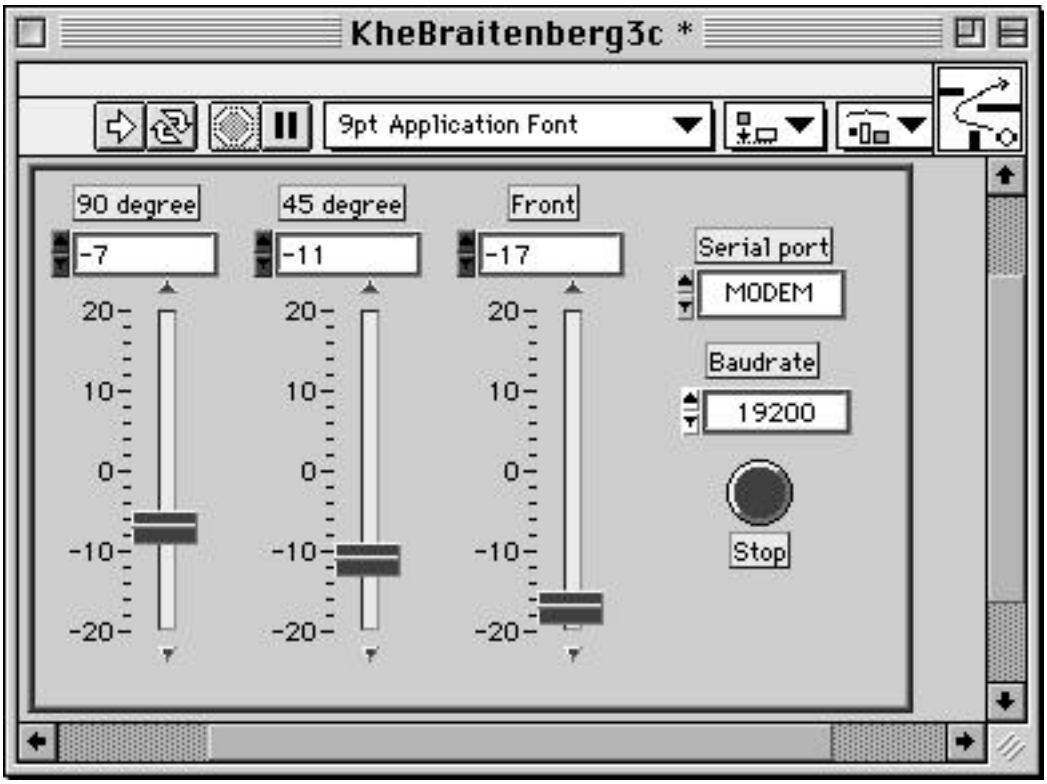


Figure 23: KheBraitenberg panel 3 sliders defining the link weights.

Each of these three sliders defines the sensibility of the motors reaction to the obstacles of a given group of sensors:

- "Front" corresponds to the two central front sensors (2 and 3),
- "45 degrees" corresponds to the two lateral front sensors (1 and 4),
- "90 degrees" corresponds to the two sensors on the side of the robot (0 and 5).

Set carefully the serial port and the baudrate according to the settings of your Khepera robot. Start the application by clicking on the arrow. It is not necessary to click on the double arrow in the present case. The sensibility can be modified by moving the cursors or writing directly the desired values. Khepera now moves, avoiding bumping into obstacles. In a parallel way you can observe the motor commands and the sensor readings from the "Motors" and "Sensors" panels. Test different sensibilities on its behavior. This control structure is inspired from the work of V. Braitenberg [Braitenberg84].

Note that this VI uses "Motors" and "Sensors" as sub-VIs. One of the advantages of LabVIEW® is to allow a context-free use of the building modules. This fact is particularly

interesting for creating programs in a structured way and for debugging.

The button (inside the panel) labeled “stop” stops the robot and the execution of the VI. It is a much better way than using the stop icon (above rubber), because the stop button stops the robot before stopping the VI.

10.6 Advanced Programming

Now that we have executed different manipulations using LabVIEW® and Khepera, it becomes interesting to present how it has been programmed. First, it is important to be able to manipulate LabVIEW® and its rolling menus.

Select the Motors panel and open the diagram using the option ‘Show diagram’ of the menu “Windows”. Your screen now displays the following schema:

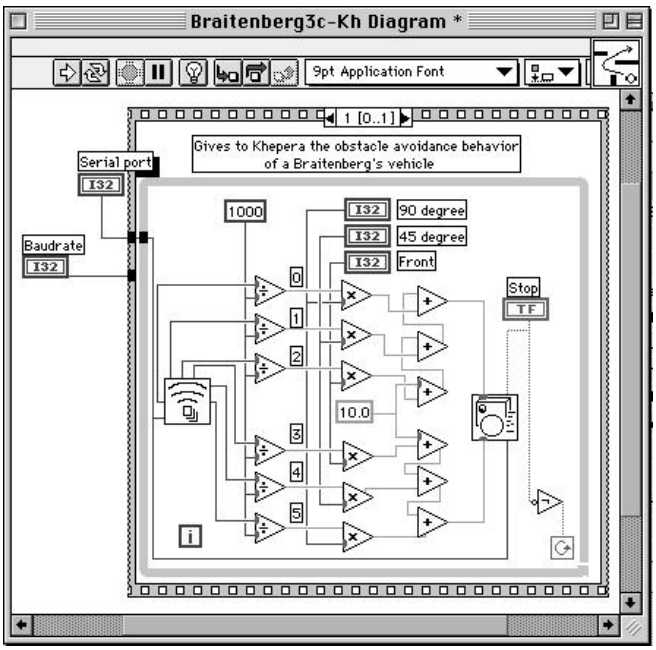


Figure 24: Motor diagram corresponding to the Motors panel.

Each element of the panel used for displaying or getting data corresponds to an icon. So, the box I32 under ‘Speed motor right’ is the getting variable of the corresponding slider on the front panel (of type 32 bits Integer). The same is true for the icon TF, labeled Stop on the left, which is a Boolean variable (type True False). This variable allows to stop the motors by sending to each one a null speed value. The triangle represents an indirection controlled by the Boolean variable “Stop”: If “Stop” is true, then the output value (on the right) will be 0, if “Stop” is false, then the output value is the Speed variable. These values are formatted by the next two icons to a string of ASCII characters. The character D is placed at the beginning of the string. Then, successively, the two speed values are added and the string is terminated by a carriage return (\n). This string is send to the robot using the serial link.

To control the speed of the wheels from another VI, the “Motors” VI can be used as sub-VI. This use is demonstrated with the KheBraitenberg3c VI..
The help window (figure 25) displays positions and semantics associated to the icon.

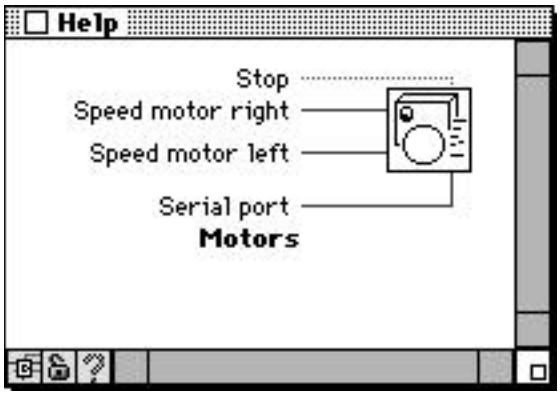


Figure 25: Semantic of the Motors connectors.

10.6.1 Sensors

Select the Sensors panel and open the diagram using the option Show diagram of the menu Windows. Your screen now displays the following schema:

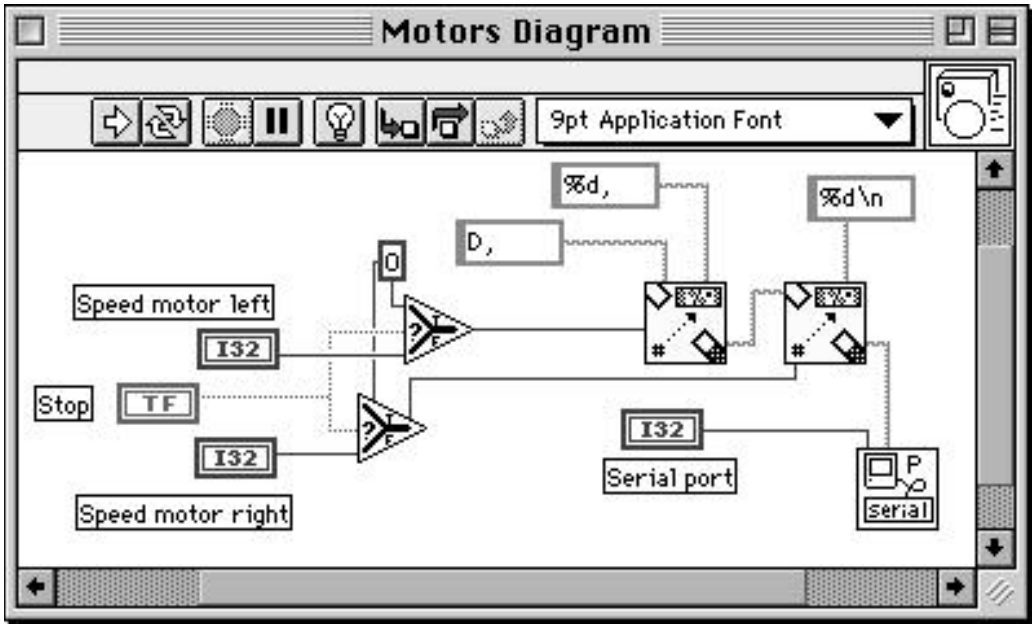


Figure 26: Diagram corresponding to the Sensors panel.

On the contrary to the “Motors” panel which sends values, this panel receives values from the serial link. Eight values are extracted in four steps from the string. These values are put into the variables (type Unsigned 16 bits) corresponding to the panel gauges. They are also put into a vector so to be displayed by Sensors_displ.

These 8 values are transmitted to others modules through the icon used as a connector. This use is demonstrated with the experiment on Braitenberg's vehicle. The help window (figure 27) displays positions and semantics associated to the icon.

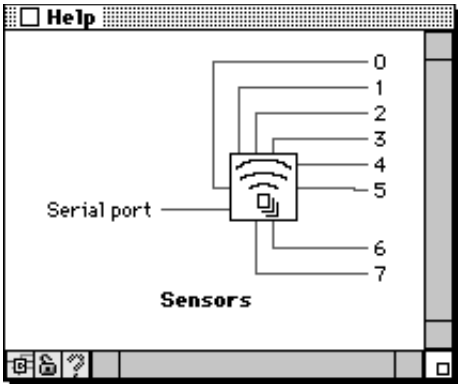


Figure 27: Semantic of the sensor connector.

10.6.2 Example of Braitenberg's vehicle.

Select the Braitenberg panel and open the diagram using the option Show diagram of the menu Windows. Your screen now displays the following schema:This schema may appear complex at first glance. But, as we will see, it takes back elements already studied. It can be decomposed into two parts: serial link initialization and avoiding behavior. LabVIEW¨ is a data-flow controlled language, so the execution order of non-dependent control structures is not fixed. Serial link initialization and avoidance behavior are independent; but initialization must occur first. The sequential structure allows the definition of an execution order, so initialization is the first element of the sequence. The second element contains the while loop where the avoidance behavior is executed until the Boolean variable Stop becomes True (note the presence of a logical inversion). This Boolean variable is also transmitted to the “motors” VI through its icon. Its action will be to stop both motors. In the same way, the two speeds computed here are sent to the “motors” VI to be sent through the serial link to Khepera.

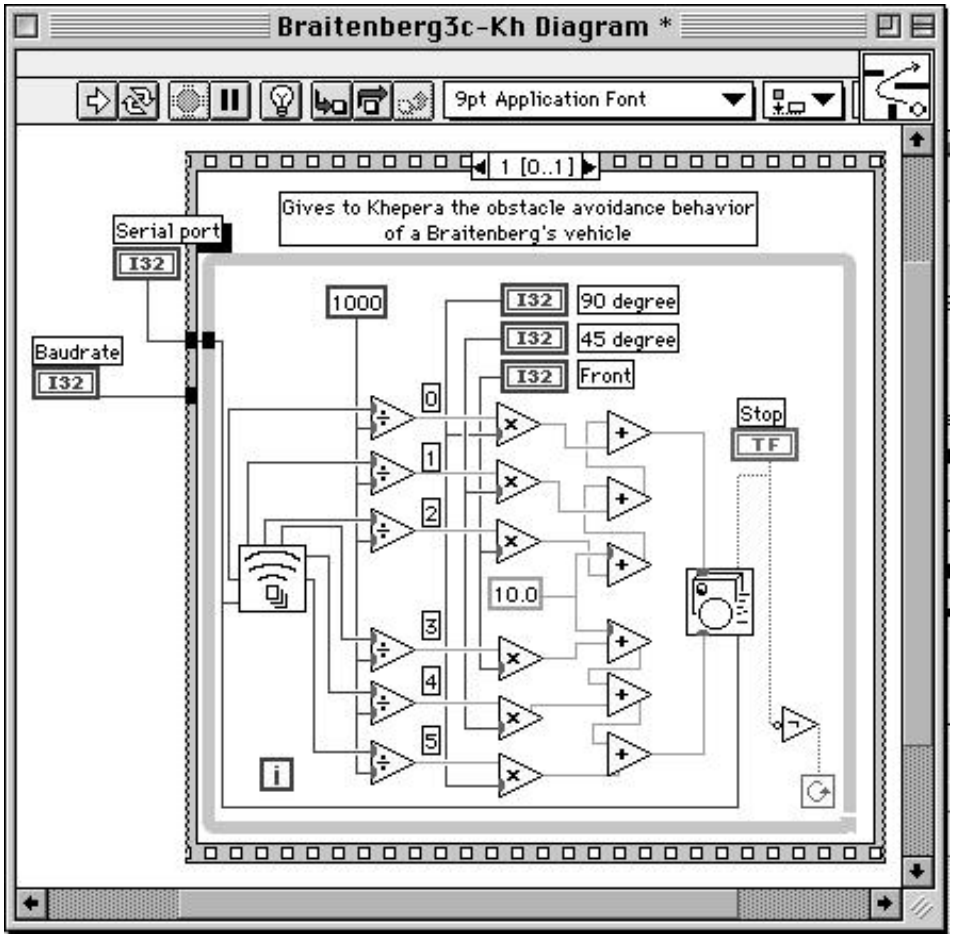


Figure 28: Diagram corresponding to the KheBraitenberg panel

Sensors values are received from the “sensors” icon. They are normalized (/1000) and multiplied by the corresponding sensibility value (type real Single). Front corresponds to the two central sensors (numbers 2 and 3), 45 degree corresponds to the two oblique sensors (numbers 1 and 4) and 90 degree corresponds to the two side sensors (numbers 0 and 5). The result (type single) is added to the value 10,0. The sum of the left sensors (3, 4 and 5) corresponds to motor right (0). The sum of the right sensors (0, 1 and 2) corresponds to motor left (1). These two values are then formatted (I32) and send to the motors. The computation needed is simple and fast enough to control Khepera in real-time without big delays. However, displaying “Motors” and “Sensors” panels is a computational expensive operation. If you want Khepera to move faster, close these panels but don't close the Braitenberg panel!

11 REFERENCE

[Braitenberg84] Braitenberg V., “Vehicles: Experiments in Synthetic Psychology“, MIT Press, 1984

[Mondada93b] Mondada F., Franzi E. and Ienne P., “Mobile robot miniaturisation: a tool for investigation in control algorithms.” , ISER3, Kyoto, Japan, 1993.

[National91] National Instruments, LabVIEW manuals for the release 5, January 1998.

APPENDIX A COMMUNICATION PROTOCOL

This communication protocol allows complete control of the robot functions through a RS232 serial line. The required configuration is presented in section 5.2. The serial line set-up on your host computer must match the one set on the robot according to the chosen running mode.

The protocol is made of commands and responses, all in standard ASCII codes. A command is sent from the host computer to the robot: it is starting with by an upper case alpha character and followed, if necessary, with numerical or literal parameters separated with comma and terminated by a line feed. The response is sent by the robot to the host computer: it is starting with the same character that was initiating the command but using lower case, and followed, if necessary, with numerical or literal parameters separated with comma and terminated by a line feed.

To better understand this protocol, please refer to the following simple test:

- Set the encoding wheel to running mode 1 (see section 3.1.3).
- Set the connection configuration presented in section 5.2.
- Start a terminal emulator on your host computer with the serial line set to 9600 Baud, 8 bit data, 1 start bit, 2 stop bits, no parity.
- Type the capital letter **B** followed by a carriage return or a line feed.
- The robot must respond with **b** followed by an indication of the version of software running on the robot and terminated by a line feed.
- Type the capital letter **N** followed by a carriage return or a line feed.
- The robot must respond with **n** followed by 8 numbers separated by a comma and terminated by a line feed. These numbers are the values of the proximity sensors presents on the robots.
- Retry the same command (N) putting some obstacles on the front of the robot. The response must change.
- Try other commands:

List of Available Commands

(Π indicates a CR(cariage return) **or** LF(line feed), ¶ indicates CD **and** LF)

A Configure

Format of the command: A, Kp, Ki, KΠ

Format of the response: a¶

Effect: Set the proportional (Kp), integral (Ki) and derivative (Kd) parameters of the speed controller. At reset, these parameters are set to standard values. Kp to 3800, Ki to 800, Kd to 100.

B Read software version

Format of the command: BΠ

Format of the response: b, version_of_BIOS, version_of_protocol¶

Effect: Give the software version stored in the robot's EPROM.

C Set a position to be reached

Format of the command: C, pos_left, pos_rightΠ

Format of the response: c¶

Effect: Indicate to the wheel position controller an absolute position to be reached. The motion control perform the movement using the three control phases of a trapezoidal speed shape: an acceleration, a constant speed and a deceleration period. These phases are performed according to the parameters selected for the trapezoidal speed controller (command J). The maximum distance that can be given by this command is (2**23)-2 pulses that correspond to 670m. The unit is the pulse that corresponds to 0.08mm. The movement is done immediately after the command is sent. In the case another command is under execution (speed or position control) the last command replaces the precedent one. Any replacement transition follows acceleration and maximal speed constraints.

D Set speed

Format of the command: D, speed_motor_left, speed_motor_rightΠ

Format of the response: d¶

Effect: Set the speed of the two motors. The unit is the pulse/10 ms that corresponds to 8 millimetres per second. The maximum speed is 127 pulses/10ms that correspond to 1m/s.

E Read speed

Format of the command: EΠ

Format of the response: e, speed_motor_left, speed_motor_right¶

Effect: Read the instantaneous speed of the two motors. The unit is the pulse/10 ms that corresponds to 8 millimetres per second.

F Configure the position PID controler

Format of the command: F, Kp, Ki, KdΠ

Format of the response: f¶

Effect: Set the proportional (Kp), the integral (Ki) and the derivative (Kd) parameters of the position regulator. At the reset, these parameters are set to standard values: Kp to 3000, Ki to 20, Kd to 4000.

G Set position to the position counter

Format of the command: G, position_motor_left, position_motor_rightΠ

Format of the response: g¶

Effect: Set the 32 bit position counter of the two motors. The unit is the pulse, that corresponds to 0,08 mm.

H Read position

Format of the command: HΠ

Format of the response: h, position_motor_left, position_motor_right¶

Effect: Read the 32 bit position counter of the two motors. The unit is the pulse, that corresponds to 0,08 mm.

I Read A/D input

Format of the command: I, channel_numberΠ

Format of the response: i, analog_value¶

Effect: Read the 10 bit value corresponding to the channel_number analog input. The value 1024 corresponds to an analog value of 4,09 Volts.

Ch 0: Used to detect the battery charger
Ch 1: Instantaneous reflected light measurement
Ch 2: Instantaneous ambient light measurement
Ch 3,4,5: Free channels corresponding to analog inputs 36, 37 and 38 on the Kbus.
Ch 6: Khepera's current consumption in mA

J Configure the speed profile controller

Format of the command: J, max_speed_left, acc_left, max_speed_right, acc_rightΠ

Format of the response: j¶

Effect: Set the speed and the acceleration for the trapezoidal speed shape of the position controller. The max_speed parameter indicates the maximal speed reached during the displacement. The unit for the speed is the pulse/10ms that corresponds to 8 mm/s. The unit for the acceleration is the ((pulse/256) /10 ms)/10 ms, that correspond to 3,125 mm/s2. At the reset, these parameters are set to standard values: max_speed to 20, acc to 64.

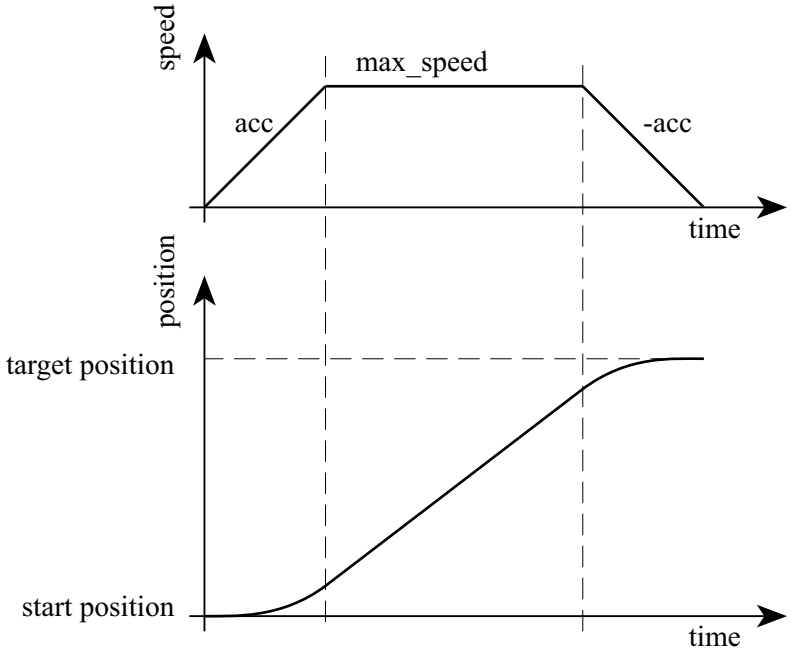


Figure 29: speed profile

K Read the status of the motion controller

Format of the command: KΠ

Format of the response: k, T_left, M_left, E_left, T_right, M_right, E_right¶

Effect: Read the status of the motion controller. The status is given by three numbers for every motor: T (target), M (mode) and E (error). T=0 means that the robot is still on movement. T=1 means that the robot is on the target position. M=0 means that the motor control is in the speed mode. M=1 means that the control is in position mode. M=2 means that the control is in PWM mode. E indicates controller position or speed error.

L Change LED state

Format of the command: L, LED_number, action_numberΠ

Format of the response: l¶

Effect: Perform an action on one of the two LEDs of the robot. Possible actions are: 0: turn OFF, 1: turn ON, 2: change status. The LED number 0 is the lateral one, the LED number 1 is the frontal one.

M Unused

N Read proximity sensors

Format of the command: NΠ

Format of the response: n,val_sens_left_90°,val_sens_left_45°,val_sens_left_10°, val_sens_right_10°,val_sens_right_45°,val_sens_right_90°, val_sens_back_right,val_sens_back_left¶

Effect: Read the 10 bit values of the 8 proximity sensors (section 3.1.6), from the front sensor situated at the left of the robot, turning clockwise to the back-left sensor.

O Read ambient light sensors

Format of the command: OΠ

Format of the response: n,val_sens_left_90°,val_sens_left_45°,val_sens_left_10°, val_sens_right_10°,val_sens_right_45°,val_sens_right_90°, val_sens_back_right,val_sens_back_left¶

Effect: Read the 10 bit values of the 8 light sensors (section3.1.6), from the front left sensor turning clockwise to the back-left sensor.

P Set PWM (pulse width modulation)

Format of the command: P, pwm_motor_left, pwm_motor_rightΠ

Format of the response: p¶

Effect: Set the desired PWM amplitude (see “Motors and motor control” on page 6 for more details) on the two motors. The minimum PWM ratio is 0 (0%). The maximal forward ratio (100%) correspond to a value of 255. The maximal backwards ratio (100%) correspond to a value of -255.

Q Unused

R Read a byte on the extension bus

Format of the command: P, relative_addressΠ

Format of the response: r, data¶

Effect: Read the data byte available at the relative_address (0...63) of the extension bus.

S Unused

T Send a message to an extension turret

Format of the command: T, turret_ID, commandΠ

Format of the response: t, response¶

Effect: Send a command and return the response of the intelligent extension turret with turret_ID. The list of turrets connected and their ID can be requested with the tool “net”. The command parameter takes the same format as a standard command, including an identification capital letter followed, if necessary, by numerical parameters separated by commas and terminated by a line feed. The response takes the same format, starting with the same letter but in lower case, followed, if necessary, by numerical parameters separated by commas and terminated by a line feed. The command and response formats are specific for every module.

U Unused

V Unused

W Write a byte on the extension bus

Format of the command: W, data, relative_addressΠ

Format of the response: w¶

Effect: Write the data byte at the relative_address (0...63) of the extension bus.

X Unused

Y Unused

Z Unused

Female connector

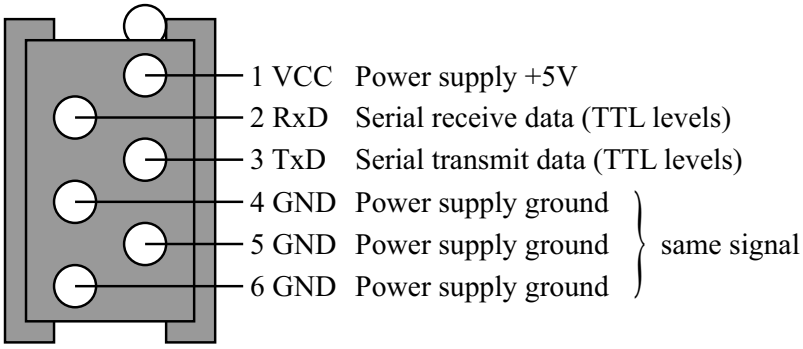


Figure 30: Serial line connector (Red), on the top.

Female connector

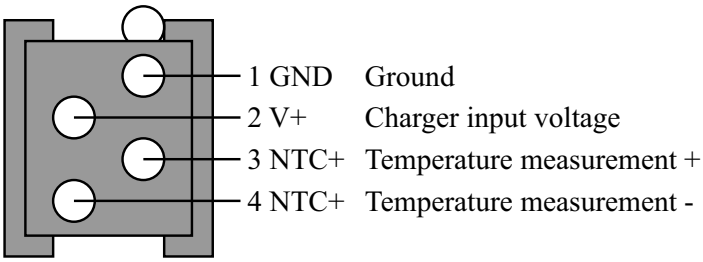
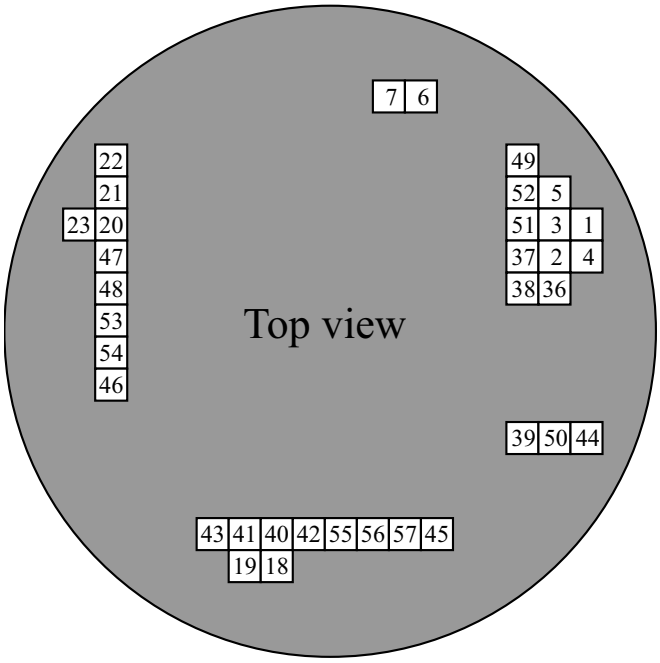


Figure 31: Battery charger connector, on the base.



Pin	Signal	Pin	Signal
1	/Reset	41	D15
2	VCC_Ext	42	A0
3	GND	43	A1
4	VCC	44	A2
5	GND	45	R/W
6	VRef	46	/CSExt
7	GNA	47	F7
18	D8	48	/IRQ6
19	D9	49	MISO
20	D10	50	MOSI
21	D11	51	SCK
22	D12	52	/CSCOM
23	D13	53	TxD
36	CH3	54	RxD
37	CH4	55	A3
38	CH5	56	A4
39	PAI	57	A5
40	D14		

Figure 32: K-Extension bus pinning

DB25 female connector

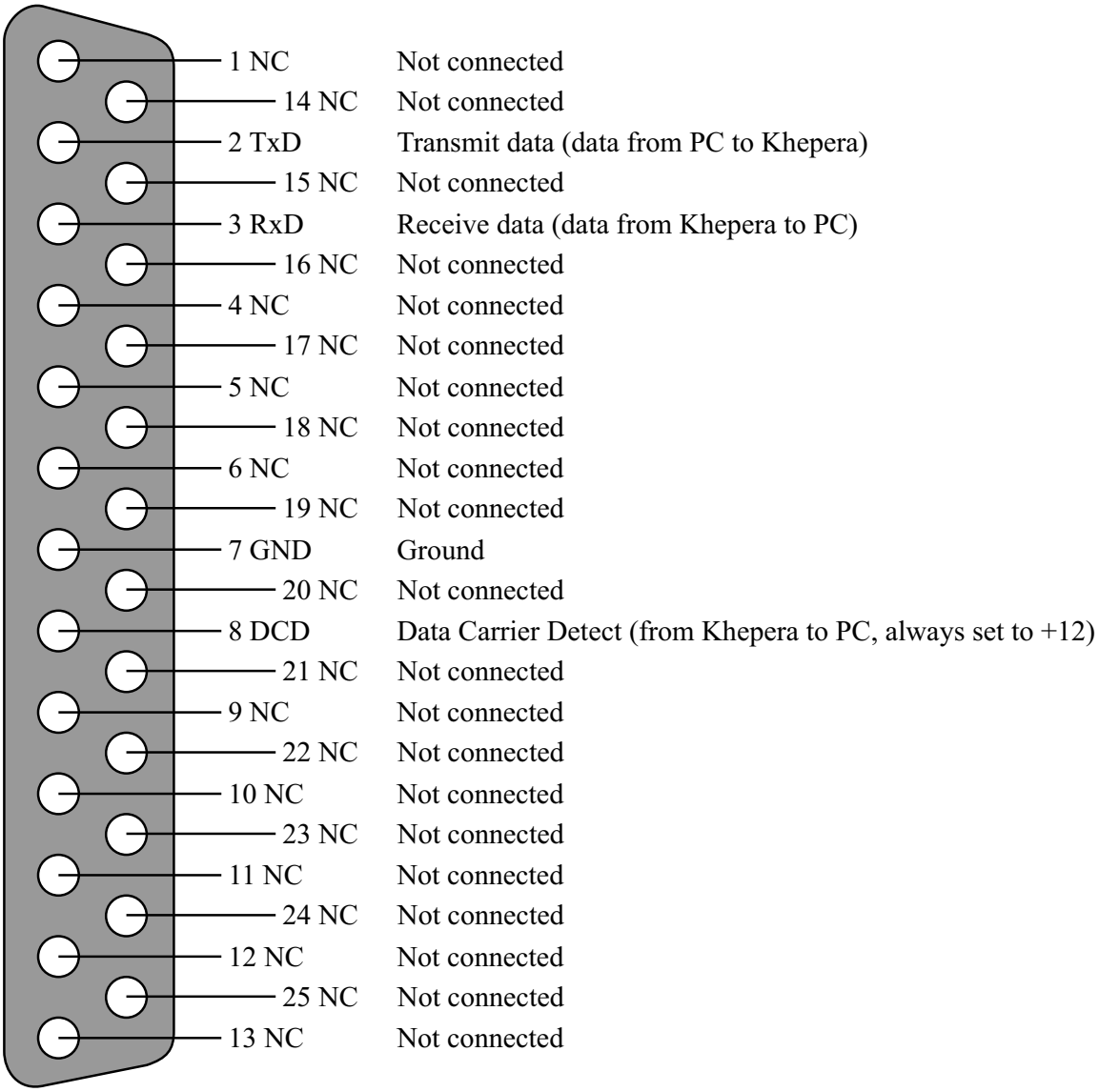


Figure 33: RS232 connector, on the charger module.

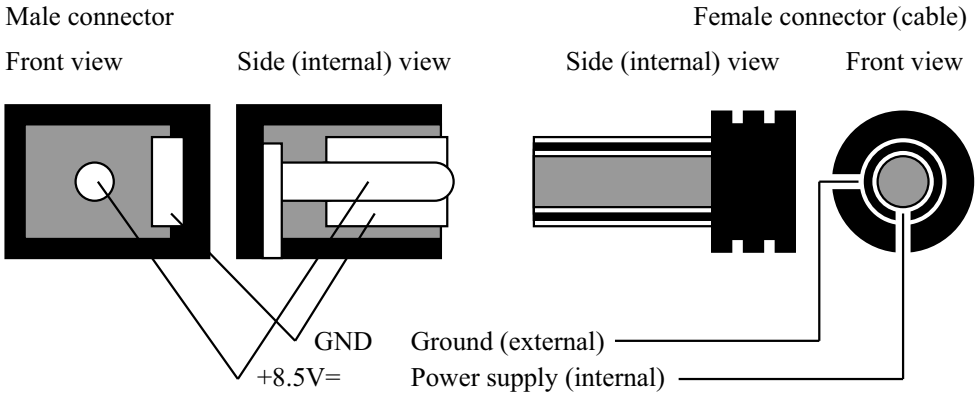


Figure 34: Power supply connector, on the charger module

APPENDIX C RUNNING MODES

Depending on your use of the robot (remote control, downloading, test, demo etc.) you can select a specific module by setting the correspondent running mode. The encoding wheel on the top of the Khepera is used to change mode (see section 3.1.1: Overview) The running mode is set according to the wheel position at boot time. One of the following sixteen modes can be selected:

0. Demonstration mode (9600 bits/s): Braitenberg vehicle algorithm (number 3 according to the “Vehicle” book [Braitenberg84]) for obstacle avoidance.
1. Serial communication mode (9600 bits/s): Mode to control the robot using the Serial communication protocol. The robot should be connected to a terminal using the S cable.
2. Serial communication mode (19200 bits/s): Same as mode 1.
3. Serial communication mode (38400 bits/s): Same as mode 1.
4. User application mode (9600 bits/s): Start an application stored in the robot's non volatile memory. The application should be flashed first using the S loader (see section 7 and 8 for details).
5. S loader mode (9600 bits/s): Robot waits for an application to be transferred in RAM and executes it when fully uploaded.
6. S loader mode (38400 bits/s): Same as mode 5.
7. Test mode (9600 bits/s): Successive tests are performed and the results are displayed using the serial link.
8. Serial communication mode (57600 bits/s): Same as mode 1.
9. Serial communication mode (115200 bits/s): Same as mode 1.

A. S loader mode (57600 bits/s): Same as mode 5.
B. User application mode (57600 bits/s): Same as mode 4.
C. Reserved
D. Reserved
E. Flash Erasing mode (38400 bits/s): The user segment of the non-volatile memory is erased.
F. uKos upgrade mode (38400 bits/s): Khepera BIOS upgrade mode.

The serial link set-up is always 8 bit, 1 start bit, 2 stop bit, no parity. Only the baud rate changes. The encoding wheel position can be changed at any time. **If the robot is running, a reset is necessary for the set-up to be effective.**

The reset button can be used at any time to reset the robot.

K-TEAM SA
1028 PRÉVERENGES
CH DE VUASSET, CP 111
SWITZERLAND