Development of a smart laser range finder for an autonomous service robot

Hannes Bistry*, Stephan Pöhlsen*, Daniel Westhoff*, and Jianwei Zhang*†
*Institute Technical Aspects of Multimodal Systems
Dept. of Informatics, University of Hamburg, Germany
†Shenzhen Institute of Advanced Technology, China
{bistry,0poehlse,westhoff,zhang}@informatik.uni-hamburg.de

Abstract— **In this paper we propose an advanced modularization of robotic systems by unification of the connections between different devices. The robot system consists of smart sensors, multiple computers for different control tasks and several actuators that are connected by standard network technologies. The paper will discuss the reasons for a modularized system design and a novel smart sensor will be presented to verify the assumptions made. The smart sensor is an embedded device for connecting laser range finders to a mobile service robot controlled by an on-board industrial PC. On the service robot, the range measurements are used for different parallel tasks like collision prevention, self-localization or people tracking. The laser range finders are equipped with a 500 kBd RS-422 serial interface. It is difficult to integrate them into an existing system because most systems do not provide a serial interface capable of operating at the unusual baudrate of 500 kBd. Therefore a special purpose RS422-to-Ethernet adapter based on a Rabbit-3000-CPU was designed. This unit receives the measurement data from the range finders and sends UDP-packets containing the data over Ethernet. Additionally, automatic control of the laser range finders and pre-processing of the measurement data is done within the Rabbit-3000-CPU. The smart sensor is not only usable on a service robot, it can be used with any device providing an Ethernet interface.**

I. INTRODUCTION

Incorporating sensor measurements in technical systems is becoming more important in many application areas. Industrial production sites become more flexible using advanced processing techniques for sensor data. Some systems like autonomous mobile robots wouldn't be possible without the use of sensor measurements. There, the interpretation of sensor data provides the basic model of the robot's environment, which is in turn the basis for all planning and task execution algorithms that follow a measurement. In the next years, autonomous robots will be applied in many applications. These could be assembly, delivery and cleaning tasks, or security services in dynamic and changing environments.

Still, not only software problems with the interpretation of sensor readings is limiting the development of new systems. The variety of technical connections between the sensors, the control computers and the actuators that build an actual system is a second factor of limitation. Systems become heterogeneous and complex. Extensive knowledge about different technical aspects of a system is required from both the developer and the user. This makes current systems hard to maintain and to extend.

We think that an advanced modularization of robtic systems into smart components and a coupling of these components using only a single network technology is a solution to the above mentioned problems. Therefore, sensors should become stand-alone devices with their own computing power. They should provide one-to-many connections so that sensor readings can be processed by multiple control computers within the whole system in parallel. These smart sensors could become of-the-shelf products which allow faster building of complex applications. The new components may be applicable not only in robotics but also in industrial production, in security applications which monitor a given area using sensors, or smart environments.

The reminder of this paper is organized as follows: In section II we present a brief overview how sensors are used in robotic research and discuss the technical challenges that remain. We introduce the service robot TASER which will be used as an example throughout this paper. It will be shown how the integration of novel smart sensor technologies reduced the complexity of its initial system design. Section III focuses on the problems and limitations of the original connection between the laser range finders and the robot's control computer. Section IV introduces the features and the implementation of the novel smart laser range finder. In section V several experiments analyze the performance and reliability of the newly developed device. Section VI gives a conclusion and an outlook on future research issues.

II. ADVANCED SENSORS IN ROBOTICS

Modern robotic systems are often complex and heterogeneous. Usually they incorporate various technical devices all connected to one or several control computers. Not only the computer systems may vary, e.g. by different operating systems, but also the type of connections between the different devices. One can find RS-232 or RS-422 connections, USB, Firewire, ARCNET, CAN, Ethernet and many more. This requires detailed knowledge by the developer and the user, and makes a system hard to maintain and to extend. Additionally, small and embedded computers are mostly unusable because the do not provide the necessary connections and may not be extendable although they would provide enough computing power.

An autonomous service robot is an excellent example for such a system. Figure 1 presents the initial interface design

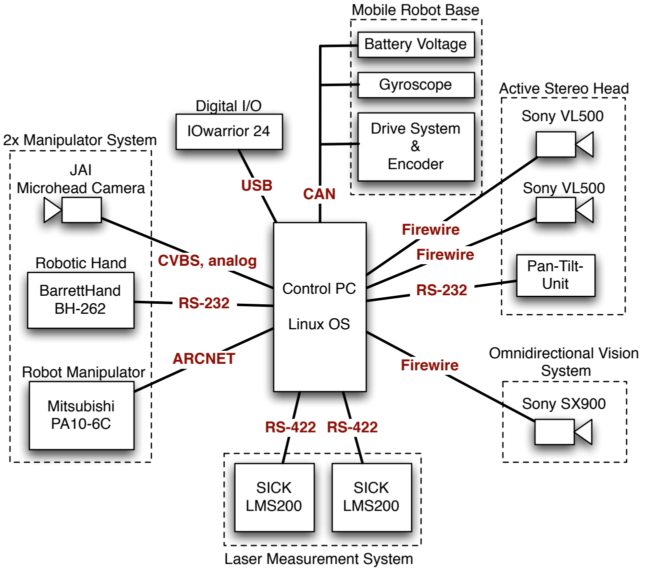


Fig. 1. This figure shows the initial setup of TASER. The control PC has to provide a variety of connection types.

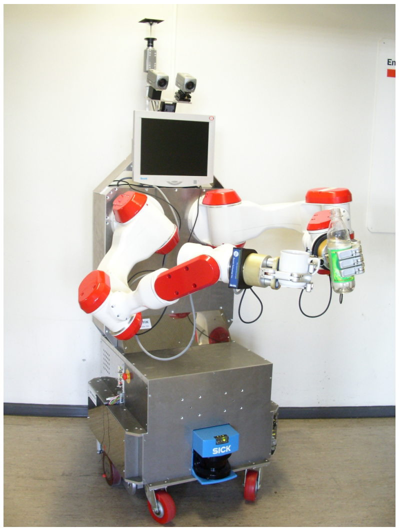


Fig. 2. TAMS SErvice Robot (TASER)

of TASER, the service robot of the TAMS institute at the University of Hamburg. The different sensors and actuators of the robot and how they are connected to the control computer are shown. A photograph of the whole robot system is shown in figure 2. The robot features a mobile platform with differential drive and wheel encoders, two laser range finders, a Pentium IV 2.4 GHz industrial computer running Linux OS, as well as two PA10-6C manipulators. The aim of the service robot project is to carry out several interactive tasks in an office environment.

The initial system design had several drawbacks. The control computer had a high work load due to the different control and processing tasks. The control application became a monolithic program, where new algorithms and sub-tasks had to be incorporated whenever new functionality was required. If real-time capabilities are required, more limitations exist due to the timings of the real-time tasks.

Since TASER has five cameras, one can imagine that the amount of data is too large to process everything in parallel on one computer. Because of that, the pre-processing of the image data was outsourced onto the cameras. Therefore, each camera must provide computing power. This idea is not new and such cameras have become available recently, e.g. the Basler EXCITE or the Sony XCI series. They are called smart or intelligent cameras and provide access via Ethernet. This enabled us to modify the system architecture of TASER and to reduce the number of connection types used on the robot. Additionally, the cameras can now be accessed in parallel by several computers that are connected to the local Ethernet on the robot.

This idea of standalone network devices has become popular in consumer electronics, too. An example are network printers and network accessed storage devices (NAS), which have become available in the low price segment in the last two years.

In the sector of robotics we propose to generally speak of smart or advanced sensors if a sensor is combined with computing power and an Ethernet interface. In the following we will present and discuss such a solution for laser range finders, one of the most popular and successful sensor devices in robotics.

Laser range finders have been used over the last decade for security applications or other high-level applications like robot self localization [1], map building [2], [3] and people tracking [4], [5], [6]. Although the use of cameras in robotics is growing, laser range finders will remain popular in the next years. With a smart laser range finder pre-processing algorithms, like those presented in [7] and feature extraction algorithms like line or edge detection, could be computed on the smart sensor. Comparisons of line extraction algorithms can be found in [8], [9].

For TASER the use of smart sensors simplified the system architecture (see Figure 3). New applications like the people tracking presented in [5] became possible. In this paper we will lay out, how the smart laser range finder relieved the system design from the technical layer up to the application design layer.

III. INTEGRATION OF THE LASER RANGE FINDERS

This section is about the problems and limitations that occur when the SICK laser range finders are connected to the service robot with a high speed MOXA PCI card. The drawbacks of the serial connection are discussed in detail.

A. *Problems of the serial connection*

During the operation of the service robot several problems occurred: When the laser range finders were set to their real-time mode, the system load of the PC was about 75 % in its basic working mode. About 40 % of the measurement data got lost. The stability of the system was also affected. Occasionally, the system even crashed when the laser range finders were operating.

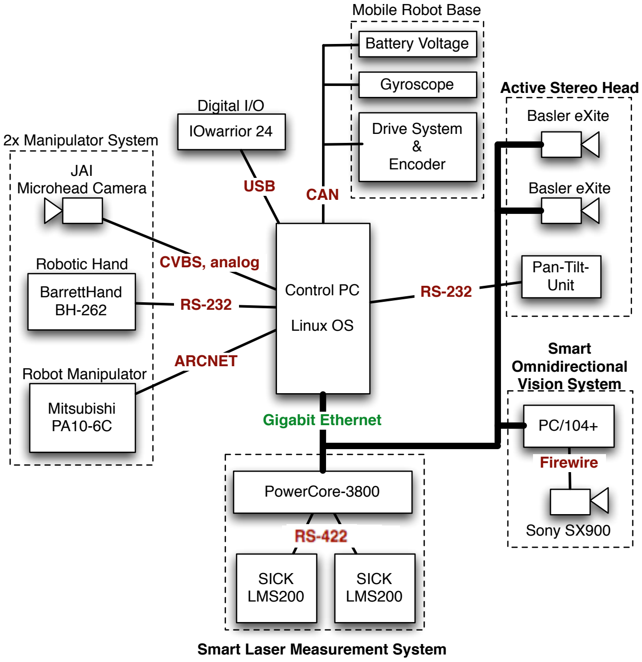


Fig. 3. This figure shows the new setup of the service robot. In the future even more sensors and actuators will be replaced by smart devices. Additional computing power can be added by connecting more computers via Ethernet.

The high system load is unwanted, because the PC has to perform further tasks like analyzing the images from the cameras or controlling the actuators (motors, arms, hands). So, the remaining resources for the higher-level tasks are very low. In most cases, switching off the laser range finder devices is not possible, because crucial tasks like collision detection are based on their data. Therefore, some analysis has been done to examine the reason for the high system load. The Linux kernel module of the MOXA PCI card has been modified so that timestamps can be set. With these timestamps the period of time that passes while reading data from the MOXA card can be analyzed. Every second the sum of the time the system was busy because of the MOXA kernel module is displayed. Additionally the amount of data actually received from the laser range finders is calculated.

During one second, an average value of 54816 bytes is read in 144 ms. This implies a system load of 14,4 % only for the kernel module. The reason for this high load is the small 16 byte FIFO of the Moxa card. The system has to perform a whole interrupt service routine for fetching a few bytes. Devices like hard disks and network interfaces provide the capability of using DMA techniques, so that the data is written directly to memory without interaction of the processor. The MOXA card does not provide this feature.

Furthermore, the small FIFO also causes the high loss rate. Everytime the system does not run the interrupt service routine soon enough after the interrupt occurs, the FIFO will overflow and drop bytes.

Theoretically, exactly 54900 bytes should be received from the laser range finders during one second. The actual loss rate is $1 - \frac{54816}{54900} = 0,0015$. According to the communication protocol of the laser range finders, one measurement data telegram consists of 732 bytes. If only one of these bytes is lost, the whole telegram is not valid anymore and has to be dumped. The 0.15 % lost bytes lead to a telegram loss of 40 %, which can be measured, when debug output is enabled in the higher level software of the service robot.

There are some additional technical drawbacks when the laser range finders are driven by the MOXA PCI card. The PC of the service robot has to provide a free PCI slot. The choice of system architecture and operation systems is therefore restricted.

B. *The smart sensor solution*

Similar problems may be found with other sensors or actuators too, and show the limitations of the service robot's initial system architecture. Too many devices have to be served from one single computer. If every device needs its own connection standard, it is difficult to configure the robot's computer in a way that every device is served properly. There are limitations for the choice of the computer because of the need for a number of PCI slots, serial ports and firewire ports. There may not be the possibility to build a robot with a small laptop computer as the control unit, because some sensors or actors are not connectable due to the lack of PCI slots.

Therefore we are considering a solution where every sensor or actuator is replaced by or advanced to a smart device. The term smart device means that these units have autonomous computing power. All these units communicate over an Ethernet connection like gigabit or fast Ethernet. The sensor data is analyzed directly on the smart sensors, and the relevant information is extracted and transmitted. With this, only already processed data has to be transferred to the control unit. The more computing power each unit provides, the less the load on the control unit will be.

The concept has another advantage: If more computing power is required, more control units can be added to the robot system, and a task can be assigned to any of them. By applying this concept, it is possible to reconfigure the robot system as needed for a special task. For example, the camera system could easily be replaced by a newer one with additional features, higher resolution and framerate.

IV. IMPLEMENTATION OF THE SMART RS422-TO-ETHERNET CONVERTER

There are several solutions on the market for connecting serial devices to Ethernet. These devices cannot be used to connect SICK laser range finders, because there is no one that supports the baudrate of 500 kBd. Most of the devices support up to 230 kBd, some support up to 460 kBd. In order to receive real-time measurement data, the communication partner has to support 500 kBd.

An asynchronous serial connection has no clock signal that is transmitted additionally to the data signal. The transmission is stopped after each 8 data bits and then resynchronized. Therefore some tolerance between transmitter clock and receiver clock is allowed. This tolerance is theoretically about 5 %, but in practice it should not exceed 3 %.

Few microcontrollers support asynchronous serial connections at high data rates. Working through the data sheets of several development boards, we found out that the Rabbit Powercore 3800 is intended to work with baudrates that are divisors of 6450 kBd. Beside the usual baudrates of 460 kBd and 230 kBd, the unit can be configured to communicate with 496 kBd. The difference to 500 kBd is under 1 % and therefore not critical.

The Rabbit Powercore is a small-size industrial PC featuring a Rabbit 3000 8-bit microprocessor, running at 51,6 MHz, 10/100BaseT compatible Ethernet, six asynchronous serial ports, 1 MB flash ROM and 1 MB RAM. Further components can be added on a user-designed motherboard, which is connected to the Powercore 3800 by a 50-pin connector. The development kit including a programming interface, an integrated development environment and a power supply is available for about $200. Part of the development tools is a royalty free TCP/IP stack. The manual gives assembler code examples for asynchronous serial communication. By analysis of the timing of some demonstration programs we figured out, that the computing power of the unit is sufficient for receiving and forwarding the data and doing some pre-processing.

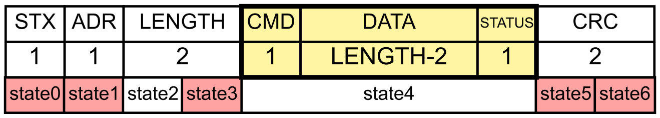
A. *Hardware features*

We developed a motherboard that is connected to the Powercore containing additional parts required for the desired application. Because the serial input and output ports have CMOS level, a dual RS-422 transceiver chip has to be added to convert the levels. Two SUB-D male connectors are soldered on the motherboard to connect the laser range finders. The power can also be supplied to the Powercore by the 50-pin motherboard connector, so on the motherboard there is a jack for an external power source. Some results of the processing of measurement data shall be visualized with LEDs in the housing of the device, so driver chips are added that connect the LEDs to general purpose I/Os of the Powercore.

B. *Software features*

The basic functionality of the device is forwarding of measurement data. Several additional functions have been added to the device to make the integration into the host system as easy as possible and to unload the processor of the host system.

1) Telegram level synchronization: One important function is the telegram level synchronization. The SICK LMS200 sends status and measurement data in a special telegram format. The task of the software is to locate the start and end of each telegram in the stream of serial data. One complete telegram has a maximum size of 732 byte, so it fits into one UDP-packet. If each packet contains one telegram, no telegram level synchronization has to be done on the host system. To achieve a solid synchronization, several criteria of a validly received telegram have to be checked for:



<table>
<tr><td>STX</td><td>ADR</td><td colspan="2">LENGTH</td><td>CMD</td><td>DATA</td><td>STATUS</td><td colspan="2">CRC</td></tr>
<tr><td>1</td><td>1</td><td colspan="2">2</td><td>1</td><td>LENGTH-2</td><td>1</td><td colspan="2">2</td></tr>
<tr><td>state0</td><td>state1</td><td>state2</td><td>state3</td><td colspan="3">state4</td><td>state5</td><td>state6</td></tr>
</table>

Fig. 5. The structure of a SICK LMS200 telegram and the corresponding state machine. The red colored states perform a check of the received data. The LSB of the length is not checked because the value is not meaningful without knowing the MSB

- first byte has to be 0X02 (start byte)
- second telegram 0X80 (address byte)
- third and fourth telegram containing a valid length
- last two byte containing a valid CRC

The unit has to be tolerant against all possible errors, and should resynchronize to the stream of data as fast as possible. For example, the Powercore 3800 could be connected to a LMS200 that is already sending data, or that bytes get lost during transmission due to interference. In this case the CRC byte will be false and the system has to drop the current telegram. All these checks are implemented into the interrupt service routine of the serial ports. This is done because one telegram should be saved in memory without fragmentation. The routine for sending UDP packets takes one start address and the number of bytes to send. If the partitioning of the received data into telegrams would be done outside the interrupt service routine, an additional copy operation would be necessary to arrange the data. A six state automaton is implemented to provide the described functionality (see fig. 5) .

2) Control of the laser range finders: The laser range finders are configured by telegrams sent by the host system. These telegrams can contain commands to change the operation mode or simple status requests. After receiving such a command, the laser range finder responds with a telegram that indicates whether the change of the operation mode was successful.

At start up, the laser range finders could be set to one of the four possible baudrates. These are 9.6 kBd, 19.2 kBd, 38.4 kBd of 500 kBd, but it is unknown which is set. Therefore the developed unit tries to determine the actual speed the following way. It sends a status request telegram with each of the possible speeds and waits for the response. If no response arrives, the next baudrate is tried. After the response has been received, the commands are sent to change the speed to 500 kBd and to enter the mode to transmit all measurement values as a stream of data. The data is received by the unit and forwarded to the host PC. If no measurement data is received for one second, it is assumed that a connection problem has occurred. In this case, the system tries to resynchronize to the laser range finders by repeating the whole process of synchronization described above.

This process is also implemented as a state machine that is programmed in the C language. The code is called periodically to check timeouts and it is called each time a telegram is received.

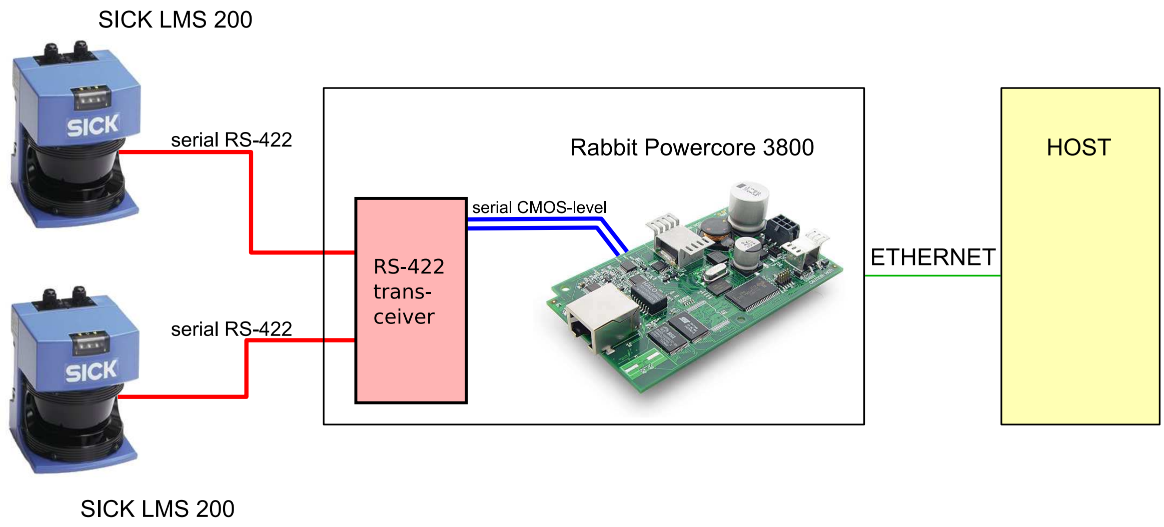


Fig. 4. This figure shows the structure of the whole system. The two SICK LMS200 are connected to the Powercore 3800 via a dual RS-422 transceiver-chip. Forwarding of the measurement data to the host PC of the service robot is done by UDP packets over Ethernet.

3) Pre-processing of measurement data: The Powercore 3800 provides enough computing power to do some easy processing tasks. We implemented a search for reflectors and a check if any object penetrates minimal distance around the center of the robot. The SICK LMS200 checks the density of the reflection of each measurement value. If you attach special reflectors in a room at the appropriate height, the laser range finder will be able to recognize them. For each 13-bit distance value, the system outputs a 3-bit density value that is typically 0 if no reflector mark is hit. Knowing the position and the arrangement of the marks in a room, a self-localization algorithm for the robot can easily be implemented. Therefore, there is the need for generating a list that contains the position of all recognized marks. One problem that had to be solved is that one mark can appear in several adjacent measurement values. In this case the position of the mark has to be approximated. This is done by averaging the measurement values weighted with the 3-bit density value.

Data of the laser range finders is used on the service robot to prevent collisions. If any obstacle comes to close to the robot, the motors are stopped. In the previous implementation, for every set of measurement data the nearest distance is calculated. This distance is calculated towards the center of the robot and not towards the laser range finders itself. Because of this, trigonometric calculations are performed to obtain the distance. The Rabbit 3000 processor does not feature a floating point unit, so these operations would be too slow to be done in real-time. For this reason the way of collision prevention on the robot system is modified. The Powercore calculates if two circular areas around the center of the robot are penetrated by any object. This can be done by comparing each value with previously calculated thresholds. The thresholds depend on the actual angle of the measurement. At system start up a list of these values is calculated. In the current configuration the two areas are 1.0 m and 0.6 m around the center of the robot. If no object violates the area of 1.0 m, no further calculation has to be performed on the host system of the service robot because the next obstacle is far away. For the case that both areas are violated, the robot has to stop its motion, and there is no need for additional calculations on the robot, either. Only for the case that the 1.0 m area is violated but the 0.6 m one is not, there have to be calculations on the robot itself, which are mainly adjustments of the driving speed of the robot. The result of the minimal distance check is visualized by LEDs in the housing of the device.

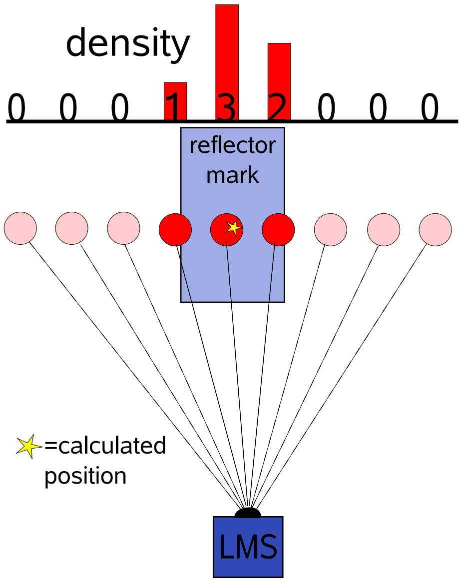


Fig. 6. If one mark is hit bay multiple laser rays, the software will calculate the probable position of the mark by averaging the weighted values.

4) Data transmission: The processed data is sent to the host system as UDP packets. The UDP protocol was chosen due to several reasons. There is less overhead compared to a TCP connection, which would cause an additional load on the Powercore and on the host system. The retransmission of lost packets is unwanted, because they would contain old data. The TCP connection would have to be handled as a stream of data, so the previously determined apportionment to telegrams would be lost. A further advantage of UDP packets is the ability to send them to a multicast address. Every device in the local area network, which subscribes to the multicast address, receives the packets. The measurement data can be used by many devices.

This feature is already used by a project that uses the measurement data from the laser range finders on an external computer. The control PC of the service robot is configured to route the packets from the wired LAN to the wireless LAN. Aim of the project is to track the motion of people based on laser range finder data and on images from several digital video cameras (see [5]). To achieve a solid tracking algorithm, both data are included in the tracking process. With the smart laser range finders, this tracking is done in parallel to the normal operations of the robot e.g. self-localization.

There are several other possible applications where sensor data has to be processed by multiple programs on multiple computers. This will be a research topic in coming projects.

V. BENCHMARKS OF THE SMART INTERFACE

The main goal of the smart interface was to reduce the system load on the service robot and to avoid the loss of telegrams. To verify if this goal is achieved, several tests were done. The software of the Powercore counts the telegrams that had to be dropped due to interference on the transmission line or due to lost bytes because of an interrupt service routine that was executed too late. The system was operating for some hours and the values have been checked regularly. Not one telegram got lost during the test period. On the robot system, the received telegrams were also counted, because there could be a loss of UDP packets. In our case we received telegrams with a constant rate of 37.5 telegrams per second from each SICK laser range finder. This matches the telegram rate of these devices. On the test setup, the Powercore was connected to the PC directly with a crossed Ethernet cable, so no collisions could occur.

In another test, the system load of the control PC of the service robot was measured while reading and processing the measurement data. This was done over a period of one minute by analyzing the file /proc/stat. Results of this test are shown in fig. 7. The load is divided into system load (drivers, Linux kernel) and user load (running programs). You can see a strong reduction of the system load from 23,4 % to 3,9 % compared to the MOXA interface. The user load increases slightly. This happens because of the fact that 69 % more telegrams have to be processed by the complex localization algorithm. (100 % instead of 59.2 % lead to 69 %) The computing time per telegram instead has been reduced to 64.5% (100% = connection with MOXA) due to the pre-processing of the data.

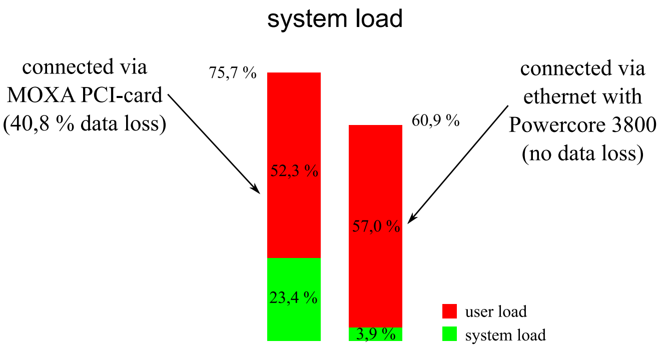


Fig. 7. The smart sensor reduces the system load on the control PC while slightly increasing the user load, because more packages are served.

VI. CONCLUSION

In this paper we presented the advantages of modularization of a robot system. Several devices were coupled via standard network technology. This provided a unified architecture for the robot and decreased the system load on the control PC. As an example, the development of a smart laser range finder was given and its advantages were discussed. Through the use of smart sensors novel applications like in [5] became possible and can be executed in parallel to the basic control program of the robot. In the future, the smart sensor technology enables us to do further tasks, which were not possible with the initial system architecture. Additionally, the smart sensors are applicable to other application fields.

REFERENCES

[1] Sebastian Thrun, Dieter Fox, and Wolfram Burgard. *Probabilistic Robotics*. MIT Press, 2005.

[2] Jose A. Castellanos and Juan T. Tardos. *Mobile Robot Localisation and Map Building: A Multisensor Fusion Approach*. Kluwer Academic Publishers, 2000.

[3] Hartmut Surmann, Andreas Nüchter, and Joachim Hertzberg. An autonomous mobile robot with a 3d laser range finder for 3d exploration and digitalization of indoor environments. *Robotics and Autonomous Systems*, 45(3–4):181-198, 2003.

[4] Maren Bennewitz, Wolfram Burgard, Grzegorz Cielniak, and Sebastian Thrun. Learning motion patterns of people for compliant robot motion. *The International Journal of Robotics Research*, 24(1):31–48, 2005.

[5] Martin Weser, Daniel Westhoff, Markus Hüser, and Jianwei Zhang. Real-time fusion of multimodal tracking data and generalization of motion patterns for trajectory prediction. In *Proc. of the IEEE Int. Conf. on Information Acquisition (ICIA)*, Shandong, China, August 2006.

[6] Katsuyuki Nakamura, Huijing Zhao, Ryosuke Shibasaki, Kiyoshi Sakamoto, Tomowo Ooga, and Naoki Suzukawa. Tracking pedestrian by using multiple laser range scanners. In *Proc. of the XXth ISPRS Congress, Commision IV, IAPRS Vol. XXXV*, Istanbul, Turkey, 2004.

[7] Steffen Gutmann. *Robust Navigation for Autonomous Mobile Systems (in German)*. PhD thesis, University of Freiburg, Akademische Verlagsgesellschaft Aka, Berlin, 2000.

[8] Daniel Sack and Wolfram Burgard. A comparison of methods for line extraction from range data. In *Proc. of the 5th IFAC Symposium on Intelligent Autonomous Vehicles (IAV2004)*, Lissabon, Portugal, 2004.

[9] Viet Nguyen, Agostino Martinelli, Nicola Tomatis, and Roland Siegwart. A comparison of line extraction algorithms using 2d laser rangefinder for indoor mobile robotics. In *Proc. of the 2005 Int. Conference on Intelligent Robots and Systems (IROS2005)*, Edmonton, Canada, 2005.