ELSEVIER

Microprocessors and Microsystems 21 (1998) 349–361

MICROPROCESSORS AND
MICROSYSTEMS

A modular computing architecture for autonomous robots

Huosheng Hu[a,*], Dongbing Gu[b], Michael Brady[c]

[a]Department of Cybernetics, University of Reading, Whiteknights, Reading RG6 6AY, UK
[b]Department of Electronic Engineering, Changchun Institute of Optics and Fine Machines, Changchun 130022, China
[c]Department of Engineering Science, University of Oxford, Parks Road, Oxford OX1 3PJ, UK

Abstract

This paper presents a modular computing architecture used for intelligent control of autonomous robots. The architecture takes the form of multiple sensing and control layers, based on Locally Intelligent Control Agents (LICAs) in which IBM PowerPC, SIEMENS 80C166, and INMOS Transputers are adopted. The control tasks are distributed among a set of the LICA-based behaviour experts that tightly couple sensing and action, but which are also loosely coupled to each other. It provides a high bandwidth computing platform for real-time navigation and control tasks embedded in real-world applications. Based on this modular design, autonomous mobile robots have been built successfully to implement both indoor and outdoor navigation. © 1998 Elsevier Science B.V.

Keywords: Modular architecture; Autonomous robots; Sensor integration

1. Introduction

To achieve the new heights of performance needed for routine and flexible deployment in the real world, autonomous robots must be able to navigate their environment [1], build or update maps [2], plan and execute actions [3], and adapt their behaviour [4] to environmental changes. In other words, they must be able to operate reactively in the real world rather than to blindly follow the orders and plans being prepared in advance. Therefore, the integration of multiple sensors and the co-ordination of navigation, planning, and execution are key for success [5]. To perform these complex tasks in real time demands huge computer power and naturally requires multi-processor architectures to bear the burden of processing loads. Such computer architectures must simultaneously incorporate all operations mentioned above, providing a fast connection at low level between sensing and action while meeting the needs of high-level planning and reasoning processes.

In general, multi-processor (or distributed) computing architectures offer a number of advantages to deal with the significant design and implementation complexity inherent in sophisticated autonomous robotic systems. First, multiple processors provide the opportunity to take advantage of parallelism for improved throughput [6]. Second, possibly redundant processors provide the capability for fault tolerance [7]. Third, a distributed implementation often simplifies the difficulties of complex hardware and software design as well as debugging [8]. Finally, the most important aspect of a system design is the need to meet real-time constraints, which is considerably easier given the concurrency inherent in distributed computing solutions. However, problems of multi-processor topologies, communications, and the optimisation of how tasks should be shared out are still challenging issues today.

A modular computing architecture used for intelligent control of autonomous robots has been developed over the last few years [6,9,10]. This paper describes such an architecture which combines different features of many currently proposed intelligent control architectures, typically hierarchical [11,12] and subsumption [4] approaches. This architecture takes the form of a network of sensing and control nodes based on a novel module, which we call LICA (Locally Intelligent Control Agent). Each LICA has its own processing units, knowledge base, decision-maker, communication links, and facilities to access external devices. A network of LICAs does not require common buses or a central communication facility. Computation is performed locally and communication occurs between any two nodes via high speed serial links. Similar to the Real-time Control System (RCS) approach [13], a LICA concept can be implemented in many different ways. For example, a LICA architecture can be implemented with FPGAs, neural nets, Texas C40s, PowerPCs, DEC Alpha, and SIEMENS 80C166.

We begin by describing how to build real-time systems to satisfy real world applications in Section 2, in which the

* Corresponding author.

system performance and architecture are discussed. In Section 3, a modular approach to build a sensor-based control architecture for our autonomous robots based on INMOS transputers, IBM PowerPC and SIEMENS 80C166 is presented. Section 4 describes the hardware configuration, software configuration, and fault tolerance in our LICA design. The experimental results obtained from both indoor and outdoor operations of the robot are given in Section 5 to demonstrate the system performance. Finally, conclusions are briefly summarised in Section 6.

2. Building real-time systems

The control of autonomous robots operating in the real world is a difficult problem since autonomous robots: (i) operate over large-scale and dynamic environments, (ii) encounter unpredictable or partially predictable large disturbances, (iii) are controlled by noisy sensors and non-perfect actuators, and (iv) are subject to real-time constraints. That is why most autonomous robots that are built today in the world are still in laboratories, hardly being used in the real world. In this section, we briefly introduce how we build a control system for autonomous robotic systems to deal with difficulties listed above.

2.1. System performance

To meet the challenge in real applications, the control system of autonomous robots should have:

- Autonomy—to be able to accomplish various tasks and manage sensors and actuators by itself.
- Adaptiveness—to be able to modify its behaviour to cope with dynamic changes and unexpected events in its environment.
- Robustness—to cope with the inaccuracies and uncertainties of sensor data as well as with hardware errors like communication failures.
- Modularity—to build up the entire system using a modular approach, in a stepwise manner from low-level behaviours such as obstacle avoidance, to high-level behaviours such as map building.
- High bandwidth—to achieve real-time performance in navigation, planning and control. It should be a multi-rate control system with response times typically from 0.001 to 1 s.
- High intelligence—to have the ability to: (i) generate a solution in unexpected situations; (ii) learn something new from its travels; (iii) communicate with other robots and humans.

2.2. Autonomous robotic systems

The key to achieve routine deployment of advanced autonomous robots in industry is the use of multiple sensors, sensor fusion methods, as well as planning and control algorithms. Since all sensors are subject to limitations and no single sensor suffices for all situations, we have developed multiple sensors in the last few years to reduce sensory noise and improve accuracy, including a vision system [6], a laser sensor [14], a sonar array [15], a lateral effect photodiode (LEP) range sensor [16], and a stereo vision head [17]. A laser scanner has been used to locate the robot in a structured environment at relative high accuracy [7]. Fig. 1 shows the typical configuration of sensors and actuators for our new autonomous robots. As we can see, it is a fully modular configuration from sensing to action in such a way that any sensor and motor drivers are easily remove or added on the robot. The modularity holds at all levels: mechanical, electrical, and computational. On the mechanical side, the top of the vehicle platform can be easily taken off by releasing few screws. On the electrical side, the power to control racks and sensor racks can be flexibly plugged in or unplugged. The emphasis in this paper is on the computational aspect of modularity, which we describe later.

2.3. Hybrid control architecture

To achieve the performance specified above, the control system of autonomous robots should include different strategies for each individual operating condition, adaptive behaviours to react under uncertain or unexpected situations, and capabilities to co-ordinate distributed sensors and controllers to implement multiple tasks in a coherent way. A distributed or decentralised control design becomes essential, and a hybrid control architecture is needed.

Fig. 2 shows the hybrid control architecture we designed for our autonomous robots. It consists of low-level feedback control loops and high level meta-controllers. Both have been designed separately, and then integrated to form a unified control system. As we can see from Fig. 2, constraints and commands from upper layers are passed

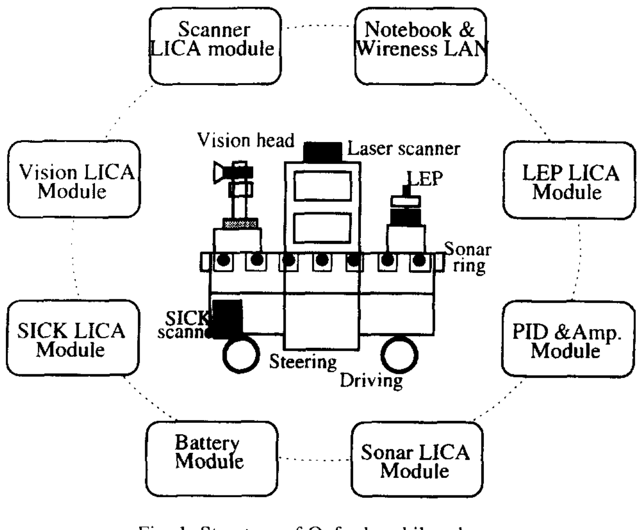


Fig. 1. Structure of Oxford mobile robots.

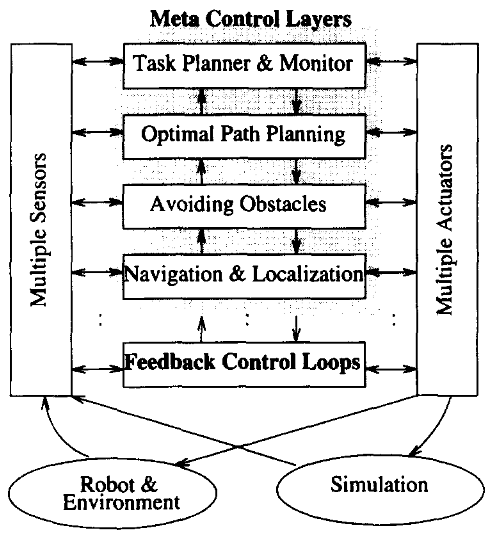


Fig. 2. Overall control architecture.

downwards, and status and data from lower layers are passed upwards. A virtual autonomous robot and its virtual environment have been simulated on a SUN workstation to test different planning and control algorithms before they can be safely used on a real robot in the real world.

The distinguishing feature of this system is its high flexibility and ease of construction. A layer-by-layer addition and a step-by-step modification can be carried out without reconfiguring the whole system each time. Different layers have different response times, typically from 0.001 to 1 s, to perform different tasks and achieve specified behaviours. For example, we are, at present, running the inner motion control loops (steering and traction) at 500 Hz to achieve the required performance. Different control approaches may be realised by changing the communication links. This modular system structure facilitates the development and testing of individual modules as well as their integration in the system.

3. Embedded controller

The fundamental idea of our design is to distribute the complex tasks of an autonomous robot into different sensing and control modules, and then combine them into several layered control loops to form different task-achieving behaviours in a bottom-up manner. Both synchronous and asynchronous communication schemes are adopted in a complementary manner. In this section, we further address how to implement this idea on a network of LICAs which will satisfy the performance requirements that were described in Section 2.1.

3.1. Modular approach

A modular design is the natural way to build a complex control system in terms of flexibility, expandability, robustness and the ease of implementation. Following this idea, we propose a single generic module, called a locally intelligent control agent (LICA), as a building block to implement our distributed sensing and control architecture.

It was decided that all necessary hardware functions are incorporated in LICA modules, compatible with a full range of sensors and actuators commonly used in autonomous robots. Each LICA is built on a 3U standard Eurocard, fitted with a few functional components: a standard TRAM (TRAnsputer Module), an interface, and a driver, as shown in Fig. 3(a). It should be noted that the number and the size of the interfaces and drivers in an LICA are completely flexible, depending on which task this LICA is to perform. The board size of an LICA is also flexible to different applications.

A typical LICA board includes the following components:

- One or more TRAMs (T2, T4, T8) with 0.5 or 1 MB of memory.
- Four high-speed serial communication data links (20 Mbit/s), two UP and DOWN control links.
- Two RS422 differential drivers for long-distance communication between LICAs.
- One or more interfacing TRAMs to sensors or actuators.
- A DIN41612 backplane connector compatible with a DIN/Eurocard standard racking system.

In general, each LICA is an expert in its own localised function and contains its own localised knowledge source. Fig. 3(b) shows its software model which includes typically six processes. These processes play different roles and cooperate with each other. They can run on a single transputer in a pseudo-concurrent manner or several transputers, depending on the complexity and time constraints of a specific task.

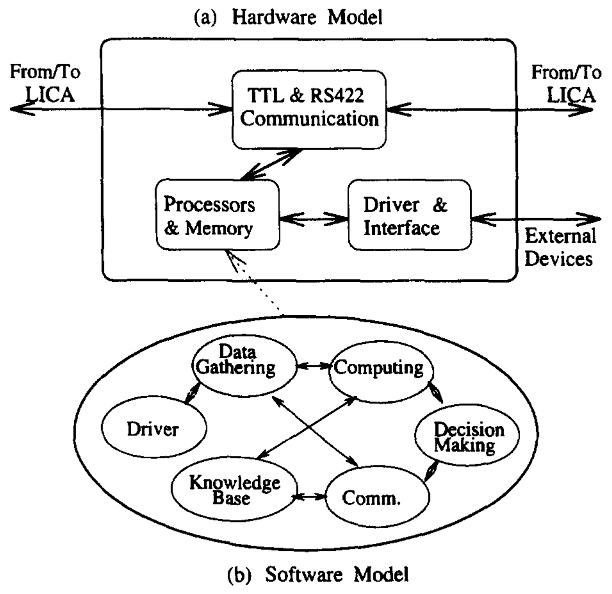


Fig. 3. Locally Intelligent Control Agent.

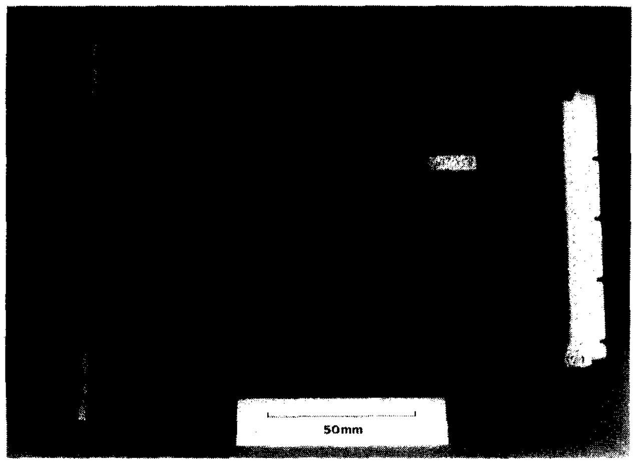


Fig. 4. A photograph of a typical LICA.

Each LICA uses high-speed INMOS serial links to communicate directly with other LICAs. There is no shared memory. The LICAs run independently, monitoring their input lines, and sending messages on their output lines. Once a LICA has input data to work on, the data are processed and the results are held for a certain predefined time for other LICAs to read. If no valid data are available, the LICA gives up its input process and continues with other processing. Two sets of RS422 differential drivers (data link and control signal) have been included on-board for long-distance communication with other LICAs or a host computer. This makes the development of distributed systems of autonomous robots much easier. Fig. 4 presents the photograph of a typical LICA, in which there is a sonar interface TRAM (SIZE 3) on the left and a T800 computing TRAM (SIZE 1) on the right.

3.2. Evolution of LICAs

The transputer was one of fastest microprocessors on the market when it first appeared in 1983. Since it is specially designed for dedicated use in embedded real-time systems, it has outstanding capability for input and output, and very efficient response to interrupts. These features are supported by a timesharing (multi-programming) operating system which is built into the chip. An application program can therefore be structured naturally as a collection of loosely coupled processes, modelling the parallel structure of the environment in which they are embedded. Another exciting feature is that if an application requires great computing power, more transputers can be simply added via two wire links to execute the processes of the program which has been written without change. These are the main reasons why we adopt transputers in our LICAs.

However, some sensors in robotic systems demand high computing power. For example, a stereo vision system we used has two black/white cameras with pixel resolutions. Each camera typically delivers approximately 6.5 MBytes/s (25 frames/s) at a digitisation of 8 bit/pixel. Processing and transferring of these data are time-consuming. For example, we have to use more than 30 T800 transputers to run some stereo matching algorithms. Therefore, we need a new generation of LICAs which could provide such computing power needed for highly demanding tasks. Fortunately, IBM PowerPC and other fast processors available on the market provide us with the ability to do so. As shown in Fig. 5, a PowerPC has been introduced as core processing engine for LICAs. Currently, we use Transtech TTM600 in our stereo vision system, which has a PowerPC 603 chip to produce more than 53 MFlops performance [18]. A T805 transputer is used as communication processor with approximately 6.5 MBytes/s transmission rate. A photograph of the PowerPC LICA is given in Fig. 6.

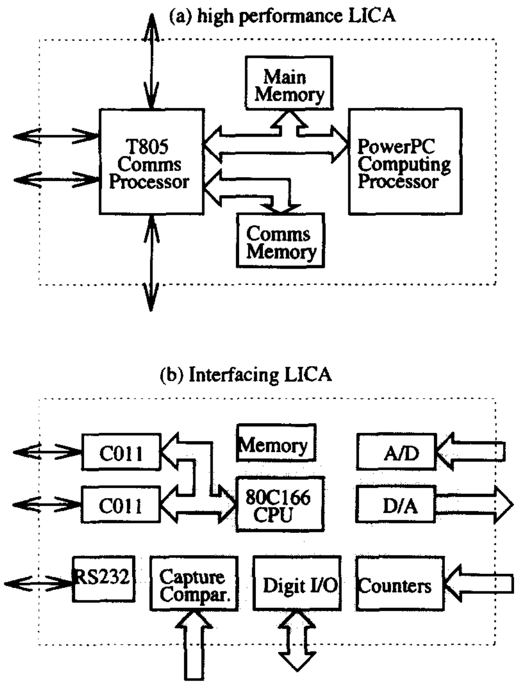


Fig. 5. Evolution of LICAs.

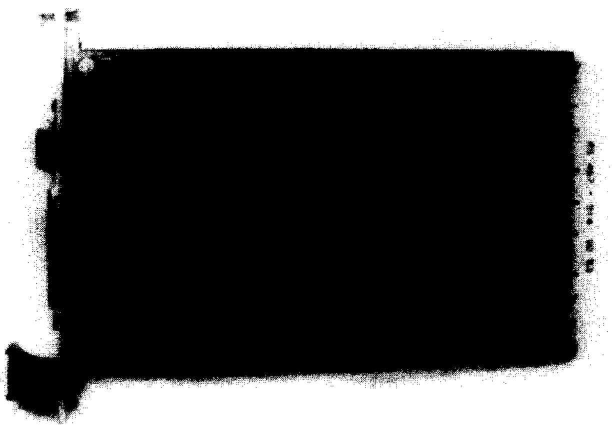


Fig. 6. A photograph of the PowerPC LICA.

To increase the interfacing capability of LICAs to external devices such as sensors and actuators in our applications, we have introduced a low-cost microcontroller, SIEMENS 80C166 [19], into our LICA architecture. It has a high-performance 16-bit CPU with four-stage pipeline and instruction cycle time at 40 MHz CPU clock. It includes a wealth of control and interfacing functions such as 10 channel 10 bit A/D converters, 16 channel capture/compare units (which could be used as D/A converters), two timer units, two serial channels, and up to 76 digit I/O lines. There are two INMOS link adaptors used for communication with other LICAs. This version of LICAs is typically used in low-level motion control of robots or various sensors.

3.3. LICA-based sensor units

The LICA design provides us with a convenient way to build intelligent systems. In this section, two LICA-based sensor units are presented to demonstrate such a flexibility, namely the LEP LICA module and the Vision LICA module as shown in Fig. 1. Note that the principle of both sensors will not be addressed here, but can be found in [20,21].

3.4. LICA-based LEP sensor

The LEP scanner has been fully integrated into the on-board transputer network, based on four LICA boards. Fig. 7 presents its hardware configuration. On LICA 1, there is an RS422 TRAM for long-distance communication to a remote host PC and three T805 transputers. Another four T805 transputers are located on LICA 2. LICA 3 has been fitted with an ADC TRAM, and LICA 4 has a digital I/O TRAM and a T2 TRAM. The whole network runs at 20 MHz communication speed, compatible with the whole control system of the AGV. The scanner interface is a 3U rack including power supply, stepper driver, and signal generator.

Fig. 8 shows the software structure for the LICA-based LEP sensor. It is written using INMOS Parallel C. There are nine processes altogether which are mapping into seven T805 transputers, running in parallel. More specifically, the feature abstraction process has a least squares estimator which abstracts a set of line segments and passes it down to the feature tracking process. The feature tracking process reads the scanner position for every scan and transform the extracted features and their associated variances from a sensor frame to the vehicle frame, based on recursive Kalman filtering.

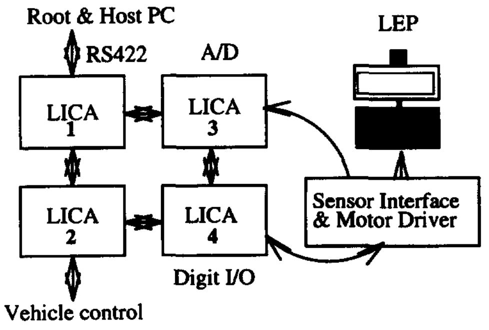


Fig. 7. The LICA-based LEP scanner.

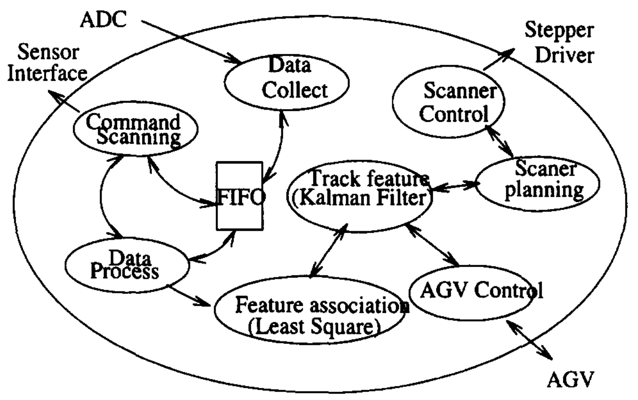


Fig. 8. The LEP software structure.

3.5. LICA-based vision sensor

An active vision system to implement visual navigation is also built using the LICA computing modules, including T800 transputers and a PowerPC processor [20]. Fig. 9 shows the system hardware configuration and Fig. 10 shows the software mapping. The whole system works as follows:

- Stereo images are grabbed through two cameras;
- Each image (240 × 240) is divided into two parts and sent to a PowerPC processor's local transputer T805 via two communication channels. This local transputer is responsible for all channel communications between the PowerPC and other transputers in the network;

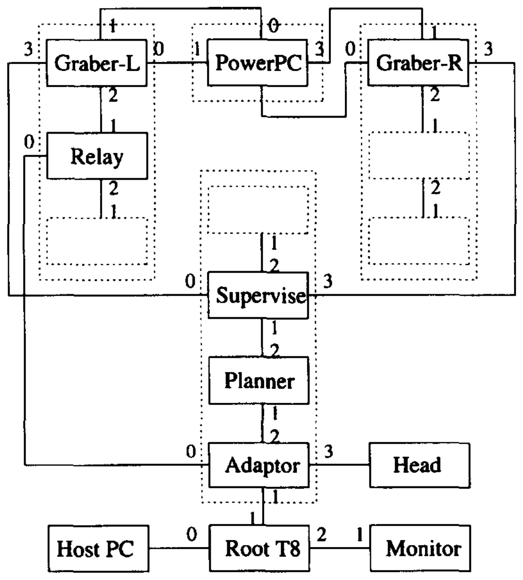


Fig. 9. Hardware configuration of the vision head.

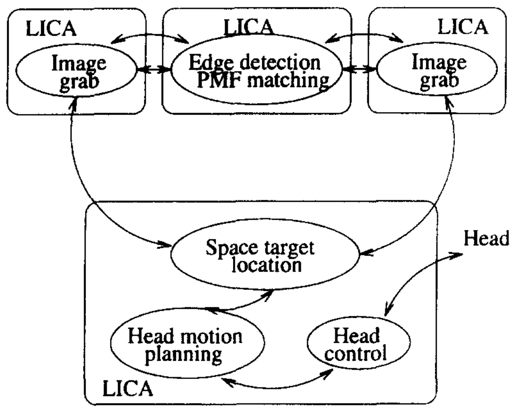


Fig. 10. Software mapping of the vision head.

- The interest part of this stereo image pair is sent to the PowerPC processor and processed there. Two kinds of processing are performed, i.e. Sobel edge detection and PMF stereo matching [20]. The resulted image position of the tracked object is sent back to the transputer *Supervise* for head motion planning via PowerPC's local transputer and image grabber transputers;
- Virtual space target position is computed according to equations in transputer *Supervise*. Kalman filters are used here to get a smooth motion of the tracked object;
- The updated information of the tracked object is then sent to the transputer *Planner* for head motion planning;
- The planned head motion is sent to transputer *Adaptor* and converted to head joint motion commands used to control DC motors;
- Head feedback state is sent back to every transputer in the network for closed-loop control.

Eight transputers and one PowerPC processor are used in the system, which enables it to run at a frame rate of 25 Hz for a window size of 100 × 15 pixels.

4. System integration

To implement the control system designed for our robots shown in Fig. 2, we have adopted the LICA-based approach for sensors, actuators, and decision-makers. One or more LICAs incorporate a task-achieving behaviour. They

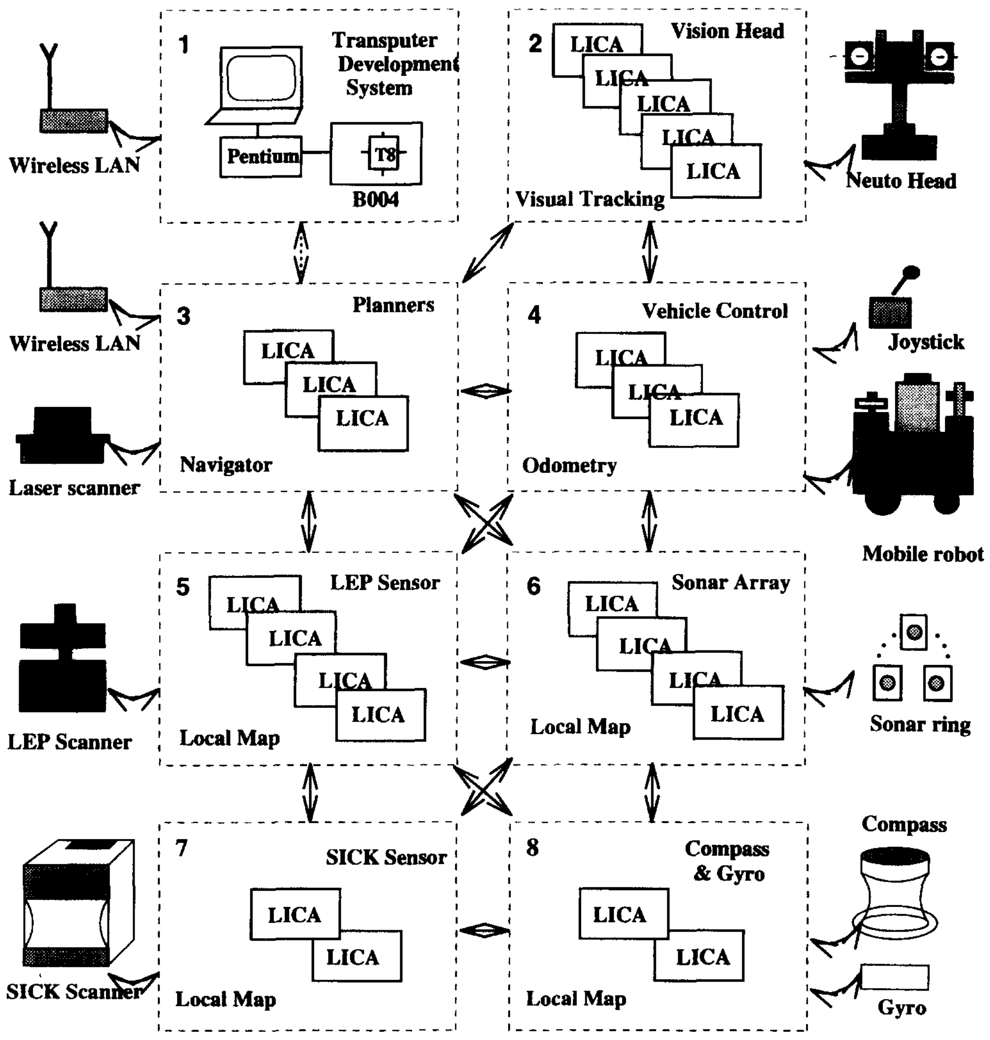


Fig. 11. An LICA-based system architecture.

monitor their input data from the sensors and other LICAs, process that data, and output final decisions. Communication between behavioural layers is loosely coupled, using asynchronous communication. In this section, we present some details on how to build a real-time control system for our mobile robot. Reliability of the system is also considered.

4.1. Hardware configuration

Fig. 11 shows that the control system of our mobile robot can be built using total 23 LICA boards. They are grouped into eight main function blocks for distributed sensing, planning, and control tasks. In particular, Block 1 is a transputer development system which is based on a Pentium PC. It downloads the control code via a communication link and collects data via a wireless LAN (5 Mb/s). Block 2 is an active vision head which sits at the back of the robot to do visual navigation. A new generation PowerPC LICA has been used to replace 12 T800 transputers to do real-time image processing, which we describe in the next section. Block 3 integrates two task-achieving behaviours: planner and navigator. Block 4 is a low-level motion control system to maintain accurate path tracking and vehicle stability.

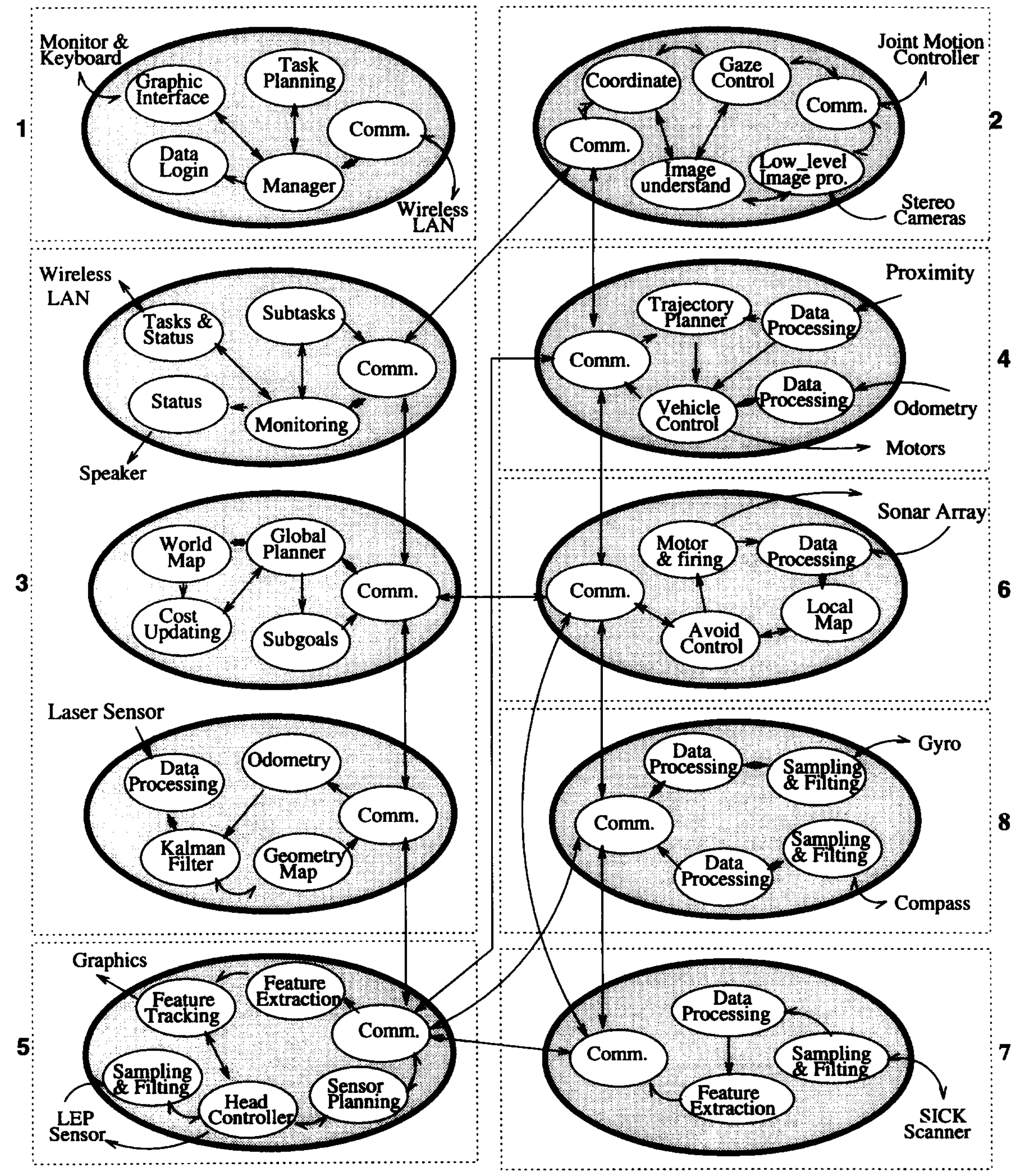


Fig. 12. A network of communication processes.

Block 5 is an LEP scanning system to monitor a 2 m range in front of the robot. Block 6 is a ring of 16 sonars to detect and avoid unexpected obstacles. A SICK laser scanner has recently been added into the system at block 7. Finally, a digital compass and a gyro sensor are also integrated into the system to provide additional orientation and the rate of rotation of the robot.

4.2. Software configuration

The LICA design enables us to achieve flexibility, expandability, and ease of implementation. However, this also very much depends upon how the software is designed. Fig. 12 illustrates a network of communication processes for the LICA-based control system shown in Fig. 11. It is written in INMOS C language. There are 10 main processes which are in turn divided into eight blocks corresponding to the hardware configuration in Fig. 11. Different sensors have been used to gather data from the real world. Then fusion algorithms such as Kalman filtering are used to track features in the real world by integrating data from different physical sensors. Meta-controllers operate at different rates and in parallel, based on maps with different formats and resolutions [9,10]. An obstacle avoidance algorithm has been developed to handle unexpected obstacles based on decision theory [15]. A trajectory planning algorithm and a GPC tracking algorithm have been used to generate and track a smooth trajectory using environmental geometry information and the robot's kinematics [22]. In contrast, the path planner works on a topological map, and searches for an optimal path consisting of a set of intermediate points [23].

It should be noted that the number of LICAs for each task-achieving behaviour such as path planning and obstacle avoidance is flexible, and depends upon the particular sensor and the system time constraints. When more transputers are needed, the processes currently residing on a single transputer can easily be mapped onto separate transputers in the LICAs. The only change we have to make is that the software channels need to be mapped to the corresponding transputer links. Therefore, LICA-based modularity offers flexibility in changing the system configuration between the different applications by exploiting transputer link communications.

4.3. Fault tolerance

As robotic systems grow in complexity and performance requirements become more demanding, the reliability and fault tolerance of such systems are primary concern for their safety. This is because the sudden shutdown of a controller may result in a catastrophe disaster in safe–critical applications, which should be absolutely prevented. Therefore, the LICA design which exploits parallel processing in a real-time control system is not only to reduce execution time, but also to improve reliability of the systems. As with many other fault-tolerant techniques, we implemented both hardware and software redundancy in LICAs to achieve fault-tolerant performance. In other words, extra resources have been added than are necessary for normal system operation.

In our design, there are three forms of hardware redundancy:

- Redundant sensors [24]—Multiple sensors have been used for the robot to navigate autonomously in a dynamic environment. The robot should remain safe in operation when some of sensors stop working, and degrade gradually.
- Redundant processors—Spare processors are used to replace the fault processors which have been detected.
- Redundant links—Extra communication links between LICAs are used to replace the fault links which cease functioning.

Software redundancy uses error detection and error correction codes to detect and rectify errors introduced during the operation of the LICAs. We use two main forms of error detection methods. One of them is Watchdog timers which are used to detect processor failures by checking for the failure of communications within a specified time period. Another is Validity checks which are used to inspect whether sensor data are within their specified ranges.

5. Real-time implementation

The proposed modular computing architecture has been implemented in our new mobile robot, LICAR-II, to navigate autonomously in both indoor and outdoor environments. In this section, we first describe our experimental

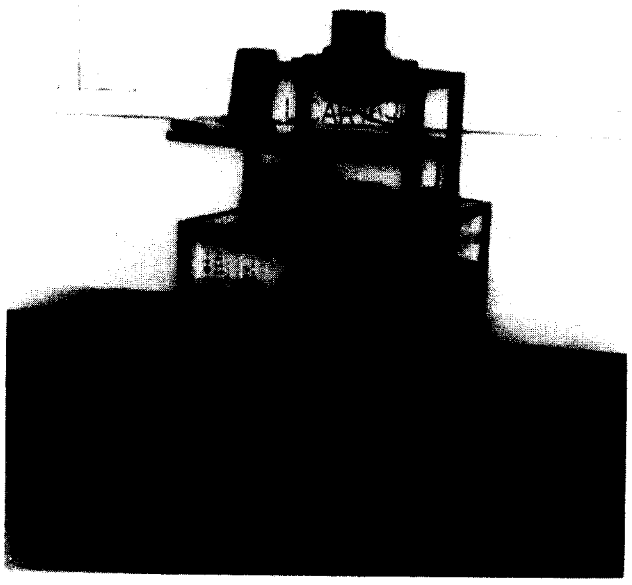


Fig. 13. A new mobile robot LICAR-II.

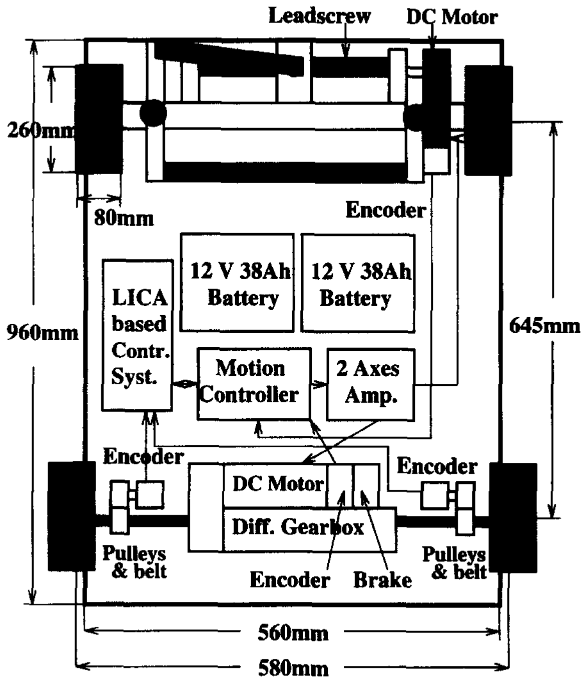


Fig. 14. Schematic diagram of the robot base.

vehicle and the navigation system being adopted. Then some experimental results are presented to show their performance.

5.1. Experimental vehicle

As shown in Fig. 13, the mobile robot LICAR-II has a complete modular structure so that it can be readily extended and easily reconfigured without a major redesign, allowing the use of a range of sensor or actuator modules. The robot has four tyred wheels with a diameter of 260 mm. The front wheels are steered together using a leadscrew driven by a DC motor, which is equivalent to a single steering wheel. The rear wheels are driven by a DC motor with a differential gearbox, as shown in Fig. 14. There are two 12 V 38 Ah sealed batteries on board to provide power. Two 500 count/revolution rotary encoders are fixed onto motor shafts for servo control by a two-axes PID motion controller. Another two 500 count/revolution rotary encoders have been connected to the rear wheels using a 3:1 ratio pulley and belts. They have been connected to the LICA-based control system to implement position control and trajectory tracking. The robot has a weight of 100 kg and a maximum speed 1 m/s.

5.2. Navigation system

The navigation system of the LICAR-II robot consists of two main feedback loops, as shown in Fig. 15. The inner loop is a two-axis motion controller to maintain stable traction speed and steering angle, based on data from encoders. In contrast, the outer loop is a position control loop to guide the robot to follow the planned trajectory accordingly, relying on position estimation from a Kalman filter. It compensates for any errors caused by disturbances from the floor and non-perfect actuators. The path generator is used to produce a continuous curvature trajectory for the robot to travel.

In our robot, the odometry calculation is based on two optical encoders mounted on two rear wheels. We assume that Δs_l is the distance travelled by the left wheel, and Δs_r is the distance travelled by the right wheel at the cycle time Δt. Let the guide point of the robot be the middle point of the rear axle. Then, the position and orientation changes of the robot are calculated using:

$$\Delta s = \tfrac{1}{2}(\Delta s_r + \Delta s_l), \quad \Delta\theta = \tfrac{1}{H}(\Delta s_r - \Delta s_l) \tag{1}$$

where $H = 580$ mm is the distance between the two rear wheels, as shown in Fig. 14.

The odometry calculation is as follows:

$$\boldsymbol{x}_r(k+1) = \begin{bmatrix} x_r(k) + \Delta s(k)\cos(\theta_r(k)) \\ y_r(k) + \Delta s(k)\sin(\theta_r(k)) \\ \theta_r(k) + \Delta\theta(k) \end{bmatrix} \tag{2}$$

where $\boldsymbol{x}_r$ is the posture of the robot. k and $k+1$ are discrete time steps.

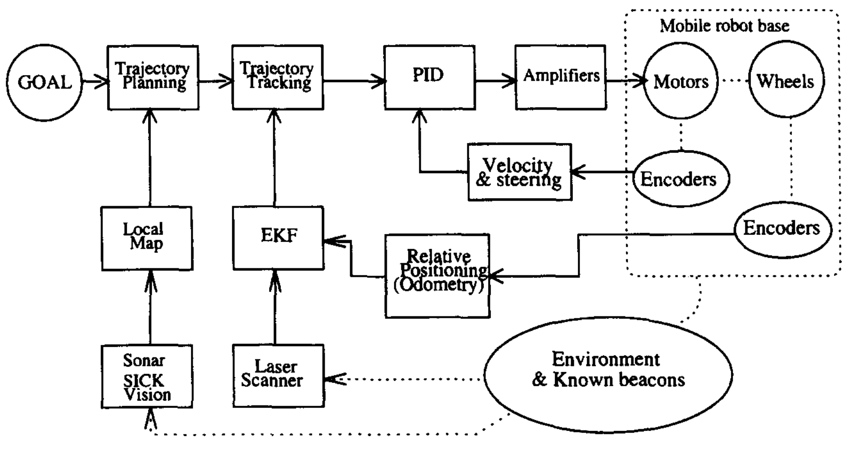


Fig. 15. The configuration of the navigation system.

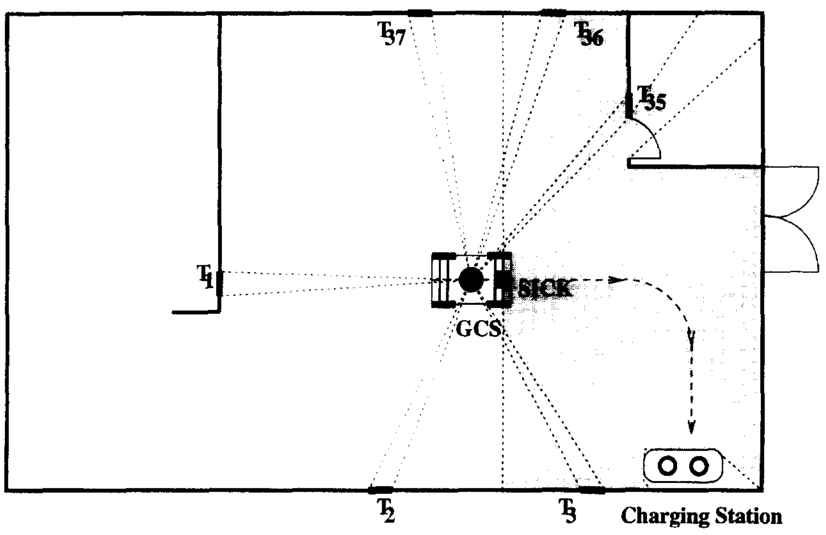


Fig. 16. The robot docks to a charging station.

The problems for odometry are: (i) surface roughness and undulation may cause the distance to be overestimated; (ii) wheel slippage can cause distance to be underestimated; (iii) variations in load can distort the odometry wheels and introduce additional errors. Because it leads to unbounded accumulation of errors, it is clear that absolute position estimation would be necessary. Therefore, a laser scanner is equipped on the top centre of the robot while the artificial beacons mounted vertically on the operating space. The laser scanner rotates at 2 Hz to measure the bearing angle from the falling edge of the detected beacon to the central axis of the robot, ranging from 0 to 360°. Prior information about the beacons can be viewed as a form of world model. With this model, an Extended Kalman Filter (EKF) is used to estimate the position and orientation of the robot by integrating the information from odometry and the laser scanner under the assumption that the errors of measurement follow a Gaussian distribution.

Suppose that the centre of the scanner is denoted by $\boldsymbol{x}(k) = [x(k), y(k), \theta(k)]^T$ and C is the distance between $\boldsymbol{x}_r(k)$ and $\boldsymbol{x}(k)$. We have

$$\begin{bmatrix} x_r(k) \\ y_r(k) \\ \theta_r(k) \end{bmatrix} = \begin{bmatrix} x(k) \\ y(k) \\ \theta(k) \end{bmatrix} - C \begin{bmatrix} \cos\theta(k) \\ \sin\theta(k) \\ 0 \end{bmatrix} \tag{3}$$

The discrete system model and observation model of our robot system are:

$$\boldsymbol{x}(k+1) = \boldsymbol{F}(\boldsymbol{x}(k), \boldsymbol{u}(k)) + \boldsymbol{w}(k) \tag{4}$$

$$z(k+1) = \boldsymbol{h}(B_i, \boldsymbol{x}(k)) + \boldsymbol{v}(k) \tag{5}$$

where $\boldsymbol{F}(\boldsymbol{x}(k), \boldsymbol{u}(k))$ is the state transition function. The notation $\boldsymbol{w}(k)$ indicates a Gaussian noise with zero mean and covariance $\boldsymbol{Q}(k)$. $\boldsymbol{v}(k)$ is a Gaussian noise with zero mean and covariance $\boldsymbol{R}(k)$. B_i is the location of the beacon i which is known in advance.

The EKF algorithm is recursively executed in four steps: prediction, observation, matching, and updating. More details can be seen in [25].

5.3. Indoor navigation

We first tested the modular computing architecture for our new mobile robot in an indoor environment whose sketch diagram is shown in Fig. 16. As we can see, there are six beacons being fixed on the wall, namely T_1, T_2, T_3, T_{35}, T_{36}, and T_{37}. Note that the shadow part is the scanning area for the SICK scanner. The robot is first commanded to dock at a charging station from the start position. An arc turn trajectory is planned by the trajectory planner to achieve such a goal. Fig. 17 shows the tracking results gathered from the docking process of the robot. We can see that the robot position estimated by the EKF was improved over the one made by odometry. In contrast, Fig. 18 shows that the robot travels at a speed of 0.3 m/s along a closed route with reasonable accuracy. The position estimates from the odometry and the EKF are presented, and

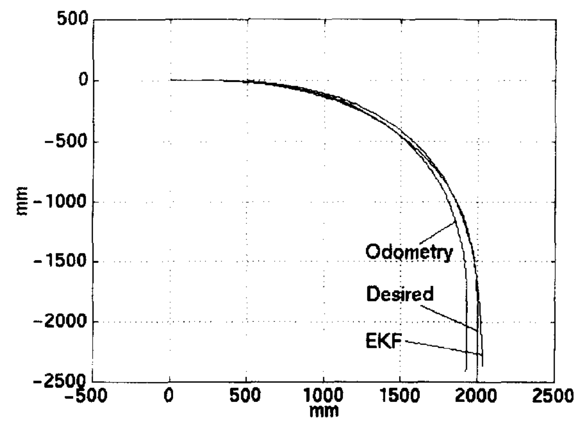


Fig. 17. Tracking results during docking.

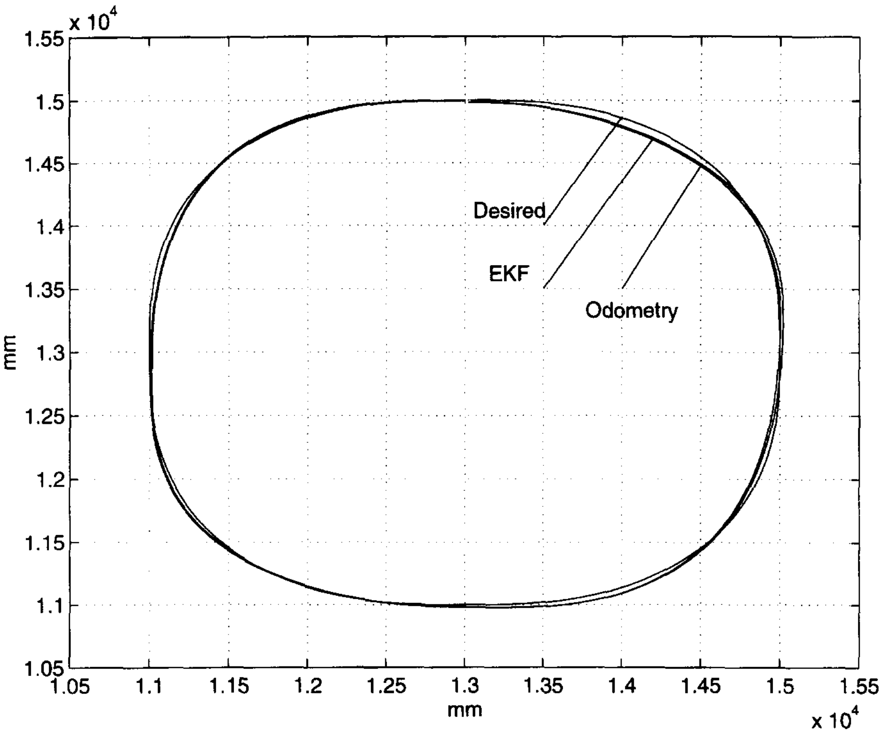


Fig. 18. Tracking a closed route indoors.

are very close. Therefore, the parameters of the EKF need to be adjusted in order to see more clear improvement made by the EKF.

5.4. Outdoor navigation

After the indoor testing, we then conducted further experiments in an outdoor environment. Fig. 19 shows a photograph of the scene of our robot travelling outdoors. The experimental results gathered from the outdoor navigation of the robot are presented in Figs. 20 and 21, in which beacons are represented by #. More specifically, Fig. 20 shows that the robot travels at a speed of 0.3 m/s along a closed route. The robot vehicle travelled about 20 m and had four-corner turning with good accuracy. In Fig. 21, the robot

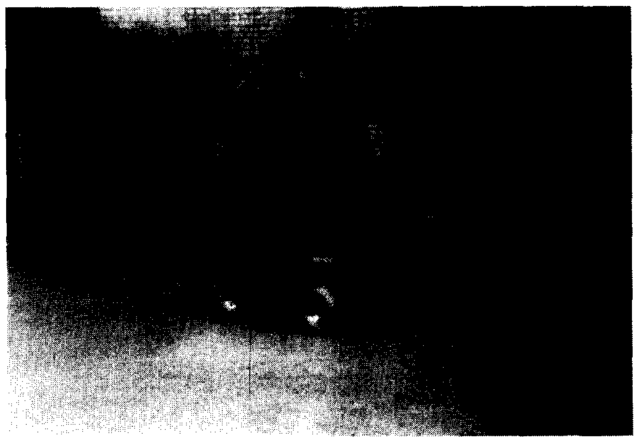


Fig. 19. A photograph of the mobile robot operating outdoors.

travelled along a relatively complex pre-planned route. The robot position estimated by the EKF is good for most of the time. However, it got worse when the robot approached too close to the target (less than 1 m). This suggested that the laser scanner should be used to detect targets at a distance of more than 1 m.

6. Conclusions

A modular computing architecture used for intelligent control of autonomous robots has been developed over the last few years. This architecture takes the form of a network of sensing and control layers based on a generic module, LICA. The control tasks of an autonomous robot are distributed among a set of behaviour experts that tightly couple sensing and action (subsumption), but which are also loosely coupled to each other (hierarchical). In other words, it combines real-time motion planning, navigation, and obstacle avoidance with high-level task planning and reasoning.

The design of the modular computing architecture presented in this paper is heavily influenced by the need for real-time sensing, control and decision-making in order for autonomous robots to interact intelligently with a dynamic environment. The proposed system has been designed to maximise the amount of parallel information flow from sensing to action so as to provide minimal delay in responding to a constantly changing environment. It is a high-bandwidth, robust and extendible control and sensing architecture. The LICA-based approach enjoys many desirable properties including modularity of sensing and

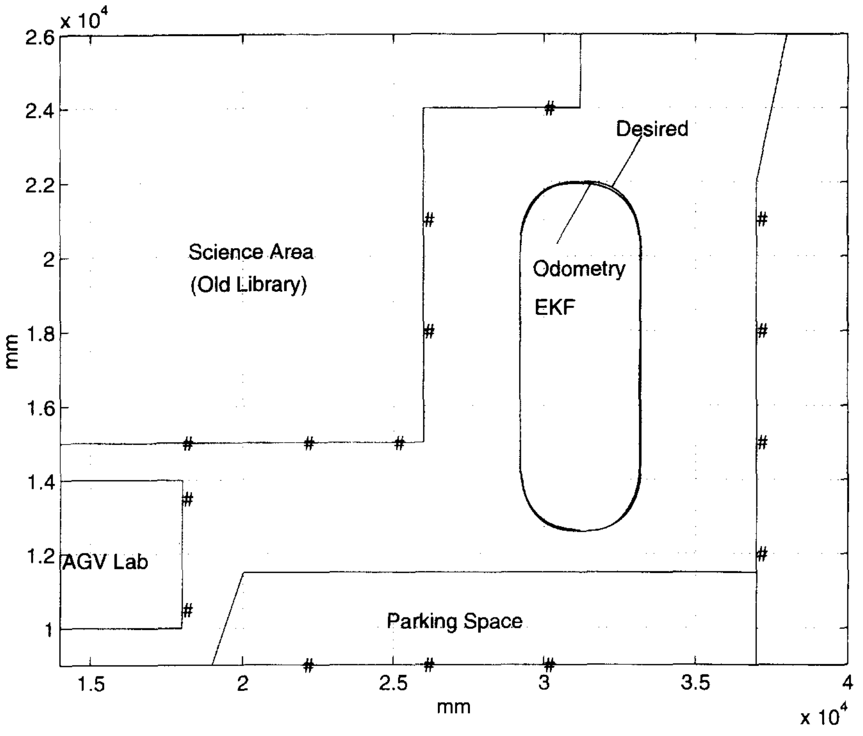


Fig. 20. Tracking a closed route outdoors.

control devices, robustness to sensor or control failure, and flexibility to the addition of more sensing and control nodes when necessary. Based on this architecture, different sensing and control algorithms have been used to achieve autonomous navigation capabilities.

The LICA architecture has proved to be a flexible, powerful platform to build complex robotic systems in real-time applications. Experimental results for both indoor navigation and outdoor navigation have demonstrated the successful implementation of such a design.

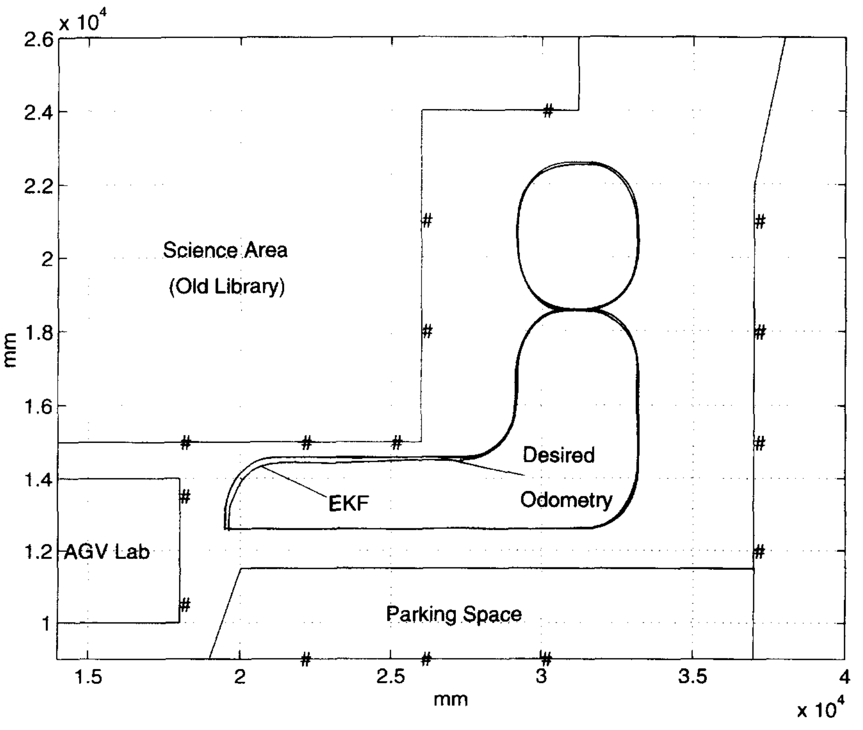


Fig. 21. Tracking a complex route outdoors.

Acknowledgements

We wish to thank Fuxing Li and Fenglei Du (now at Amherst Systems Inc.) for contributing towards the LICA-based vision system, Robert Taylor and Nick Pears (now at the University of Cambridge) for the LICA-based LEP scanner, Penny Probert, Hugh Durrant-Whyte (now at the University of Sydney) and Alan Pugh (University of Hull) for insightful conversations. The anonymous referees made many valuable comments which improved the paper. Thanks also to the EPSRC DIP for the financial support under grant GR/K39844.

References

[1] R.C. Arkin, Motor schema-based mobile robot navigation, International Journal of Robotics Research 8 (4) (1989) 92–112.

[2] J. Borenstein, Y. Koren, Real-time map-building for fast mobile robots obstacle avoidance, in: Proc. SPIE, Vol. 1338, Mobile Robot V, 1990, pp. 74–81.

[3] J. Barraquand, P. Ferbach, Path planning through variational dynamic programming, in: Proc. IEEE Int. Conf. Robotics and Automation, San Diego, California, USA, 8–13 May 1994, pp. 1839–1846.

[4] R.A. Brooks, A robust layered control system for a mobile robot, IEEE J. Robotics and Automation 2 (1986) 14.

[5] H.F. Durrant-Whyte, Integration, Coordination, and Control of Multi-Sensor Robot Systems, Kluwer Academic Press, Boston, MA, 1987.

[6] J.M. Brady, H. Hu, H. Wang, S. Udall, Parallel algorithms in vision and robotics, Parallel Computation, (1992) 335–366.

[7] H.A. Thompson, P.J. Fleming, Fault-tolerant transputer-based controller configurations for gas-turbine engines, in: IEE Proceedings, Vol. 37, Part D, No. 4, 1990, pp. 253–260.

[8] A. Elfes, S.N. Talukdar, A distributed control system for the cmu rover, in: Int. Joint Conf. Artificial Intelligence, 1983, pp. 830.

[9] H. Hu, J.M. Brady, F. Du, P.J. Probert, Distributed real-time control of a mobile robot, The International Journal of Intelligent Automation and Soft Computing 1 (1) (1995) 63–83.

[10] H. Hu, J.M. Brady, P.J. Probert, F. Li, Concurrent sensing and action framework for intelligent mobile robots, in: Proc. Int. Conf. on Intelligent Autonomous Systems, Karslruhe, Germany, 28–30 March 1995, pp. 1049–1054.

[11] J.S. Albus, Brains, Behaviour and Robotics, McGraw-Hill, 1981.

[12] R.P. Paul, H.F. Durrant-Whyte, M. Mintz, A robust distributed robot control system, in: Proc. 3rd Int. Symp. Robotics Research, France, 1985, pp. 93–100.

[13] J.S. Albus, W.G. Rippey, RCS: a reference model architecture for intelligent control, in: Proc. from Perception to Action Conference, Lausanne, Switzerland, 7–9 September 1994, pp. 218–229.

[14] P. Stevens, M. Robins, M. Roberts, Truck location using retroreflective strips and triangulation with laser equipment (turtle), in: Proc. 2nd European Conf. on Automated Manufacturing, Birmingham, UK, 1983.

[15] H. Hu, J.M. Brady, A bayesian approach to real-time obstacle avoidance for an intelligent mobile robot, The International Journal of Autonomous Robots 1 (1) (1994) 67–102.

[16] I. Reid, J.M. Brady, Model-based recognition and range imaging for a guided vehicle, Image and Vision Computing 10 (3) (1992) 197–207.

[17] F. Du, Fundamentals of an active vision system, PhD thesis, Department of Engineering Science, University of Oxford, 1994.

[18] Transtech Limited, PowerPC Toolset Manual. Transtech Parallel Systems Limited, High Wycombe, UK, 1996.

[19] Siemens AG, Data book: Microcontrollers Semiconductor Group, Siemens, Germany, 1993.

[20] F. Li, M. Brady, H. Hu. Visual guidance of an AGV, in: Proc. 7th Int. Symposium of Robotics Research, Munich, Germany, 21–24 October 1995.

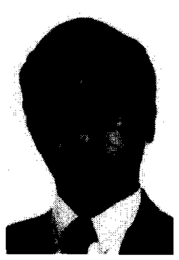
[21] N.E. Pears, P.J. Probert, An optical range sensor for mobile robot guidance, in: Proc. IEEE Conference on Robotics and Automation, 1993.

[22] H. Hu, J.M. Brady, P.J. Probert, Trajectory planning and optimal tracking for an industrial mobile robot, in: Proc. SPIE's Int. Symp. on Mobile Robots VIII, Vol. 2058, Boston, USA, September 1993, pp. 152–163.

[23] H. Hu, M. Brady, Planning with uncertainty for a mobile robot, in: Proc. 2nd Int. Conf. on Automation, Robotics and Computer Vision, Hyatt Regency, Singapore, 16–18 September 1992, pp. RO-13.4.1–7.

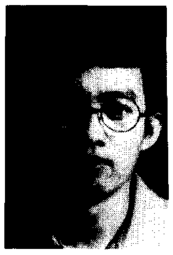
[24] A.M. Flynn, Redundant sensors for mobile robot navigation, International Journal of Robotics Research, 7(6) (1988), pp. 5–14.

[25] H. Hu, M. Brady, Dynamic global path planning with uncertainty for mobile robots in manufacturing, IEEE Trans. Robotics and Automation, 13(5) (October 1997), pp. 760–767.



Huosheng Hu received the MSc degree in Control Engineering from the Central-South University of Technology, China, and a PhD degree in Robotics from the University of Oxford, UK. He served as a lecturer at the Central-South University of Technology from 1982 to 1987, and was a visiting research fellow at the University of Birmingham, UK, during May 1987 and September 1988. From 1988 to 1997 he was a research scientist in the Department of Engineering Science, University of Oxford.

Dr. Hu is now a lecturer in the Department of Cybernetics, University of Reading, UK. He has published more than 50 papers in journals, books and conferences. His research interests include planning under uncertainty, real-time systems, intelligent robots, and computer architectures. He is a Chartered Engineer and a member of IEE, IEEE, and the Intelligent Autonomous Society.



Dongbin Gu received BSc and MSc degrees in Control Engineering from Beijing Institute of Technique, China in 1985 and 1988. He joined the Department of Electronic Engineering, Changchun Institute of Optics and Fine Mechanics, China as a lecturer in 1988, and became an Associate Professor in 1993. He was a visiting scholar in the Department of Engineering Science, University of Oxford, UK during October 1996 and September 1997, financially supported by a scholarship of SBFSS from the British Council.

Professor Gu has published 20 papers in journals and conferences, and received several awards from the Ministry in China. His research interests include real-time systems, microprocessors, optical instruments, and autonomous robots.

Michael Brady is Professor of Information Engineering at the University of Oxford. In 1980 he was appointed Senior Research Scientist in the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology, Cambridge, MA. In August 1985, he left MIT to take up the newly created chair in Information Engineering at Oxford and to found the Oxford Robotics Research Group. He became Department head in July 1989.

Professor Brady is founding father of the International Journal of Robotics Research and is Associate Editor of the Artificial Intelligence journal. His published books include Robot Motion, First International Symposium on Robotics Research, and Robotics Science. His areas of expertise include computer vision, robotics, and artificial intelligence. He is a fellow of the Royal Society, the Royal Academy of Engineering, IEE and AAAI.