Robotica (2000) volume 18, pp. 287–297. Printed in the United Kingdom © 2000 Cambridge University Press

# A climbing autonomous robot for inspection applications in 3D complex environments

C. Balaguer, A. Giménez, J.M. Pastor, V.M. Padrón and M. Abderrahim

*Departamento de Ingeniería Eléctrica, Electrónica y Automática (DIEEA), University Carlos III of Madrid, c/Butarque, 15, 28911 Leganés (Madrid) (Spain)*
*E-mail: balaguer@ing.uc3m.es*

**SUMMARY**
Often inspection and maintenance work involve a large number of highly dangerous manual operations, especially within industrial fields such as shipbuilding and construction. This paper deals with the autonomous climbing robot which uses the "caterpillar" concept to climb in complex 3D metallic-based structures. During its motion the robot generates in real-time the path and grasp planning in order to ensure stable self-support to avoid the environment obstacles, and to optimise the robot consumption during the inspection. The control and monitoring of the robot is achieved through an advanced Graphical User Interface to allow an effective and user friendly operation of the robot. The experiments confirm its advantages in executing the inspection operations.

KEYWORDS: Climbing robots; Autonomous systems; Robot path planning; Optimum planning.

## 1. INTRODUCTION
The development of special climbing, walking and mobile robots for non-traditional sectors and service application increases every day. The most active industrial sectors are: a) Construction, with application to wall erection, brick assembly, etc.;[1] b) Electric power transportation hot-line maintenance;[2] c) Shipbuilding for welding tasks in ship erection processes;[3] and d) Aeronautic application for aircraft inspection,[4] etc.

The construction industry has become one of the most appropriate for robotization, due to its current low level of automation. Another important reason is the high complexity of construction sites and environments, such as buildings, bridges, towers, etc. That is why the construction industry demands autonomous climbing robots with a high level of mobility in their type of environments.

During the last years a few well known climbing robots have been developed.[5–9] Nevertheless, these robots are mainly non-autonomous or semi-autonomous in two senses: a) In the control system of the robot, which is usually placed on the "ground"; and b) In the power supply source which is also placed in the "ground". The "ground" equipment is umbilically linked to the climbing robot using heavy wires which susbstantially reduce their mobility. Their control systems work at the actuator level only, but not in the locomotion or inspection ones.

There are many highly dangerous manual operations related to periodical inspection at construction sites. An important part of these operations is performed in large size environments formed by metallic structures with difficult and dangerous access, even for skilled workers. The most relevant examples are: Inspection of screwed/welded unions of building metallic skeletons, or inspection of the painting of the metal-based bridges with a complex structure (Figure 1). The possibility of using autonomous robots for these applications will present a very important advantage with regard to safety and quality.[10]

The main objective of the ROMA robot is the development of multifunctional autonomous self-supported climbing robots able to travel into complex metal-based environments. The navigation is performed by the robot CPU in an autonomous way without other help. The robot is able to self-support its locomotion system for 3D movements and it has the possibility of autonomous power supply using the on-board batteries or, if it is necessary, be umbilically connected to a "ground" power supply to increase the robot's autonomy. The effectiveness of the ROMA robot in the inspection of large structures, like bridges, directly depends on its autonomy.

The robot is equipped with two types of on-board sensors for inspection operations:

a) Colour cameras, and
b) Laser telemeters.

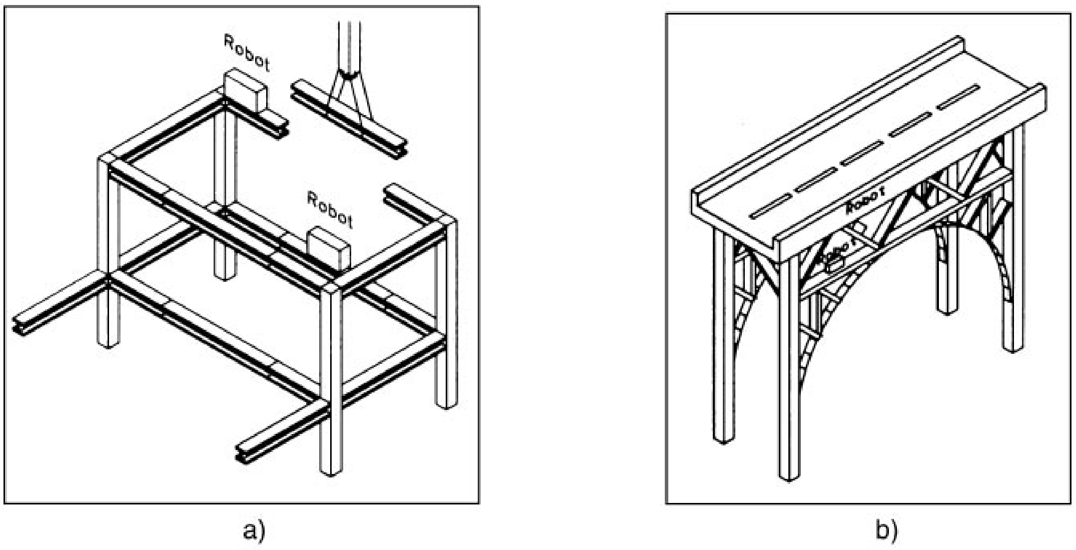
With the camera it is possible to inspect:

a) The colour to check for rust, paint and structural defects,
b) The geometric features of screws and bolts to check if they are at their required torque.

The laser is mainly used for:

a) Localization of the robot with respect to the metallic structure, and
b) It helps to extract information from the camera images.

## 2. ROBOT STRUCTURE
The mechanical, electromechanical and control design and development of the ROMA robot was performed by the University Carlos III of Madrid. There were several stages during the mechanical design. First of all, the analysis of the different movements of the robot in different scenarios was



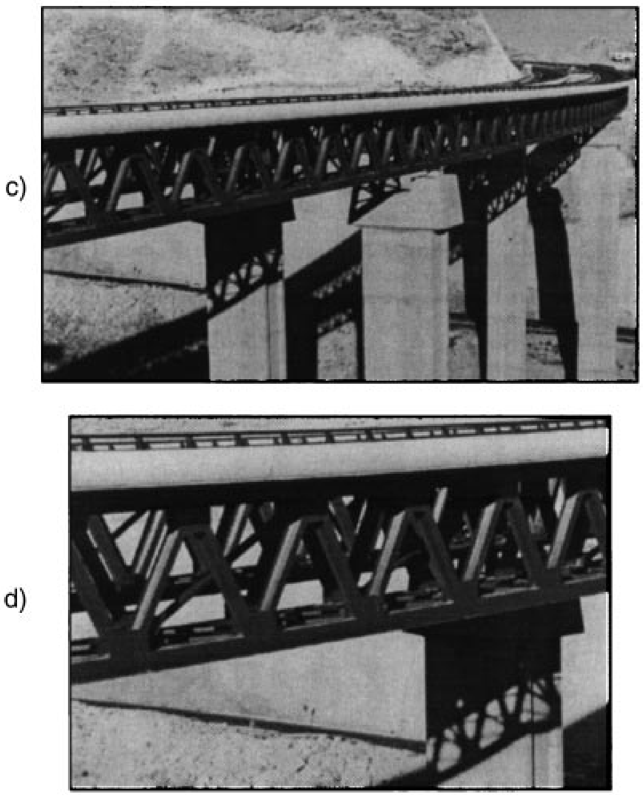


Fig. 1. a) Metal-based building structure, b) metal-based bridge for ROMA robot inspection, c) bridge real view, and d) its close-up.

performed in order to select its kinematics structure. The output of this process was the selection of the robot's number of DOFs and their ranges. Then the electro-mechanical design was performed using the dynamics analysis and simulation package ADAMS.[11] This helps to define the length and weight of the robot's parts, and select the electrical motors, batteries, etc. for the previously calculated torques.

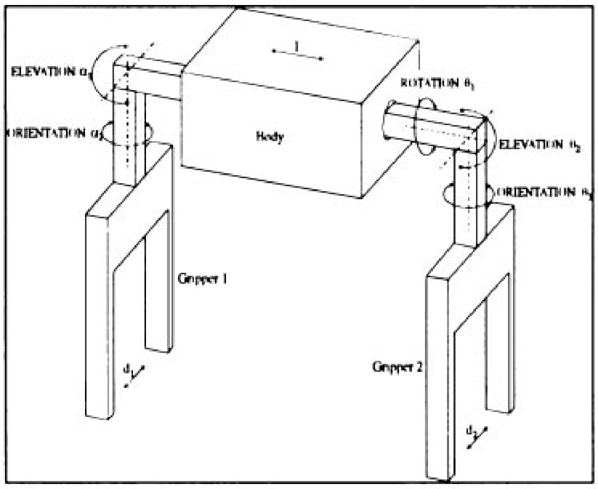
The ROMA robot is formed by three main parts (Figure 2a):

a) The body of the robot, which includes the CPU, the servo controller multiaxis board, one servo motor amplifier (driver), the batteries, the radio-based Ethernet communication with the "ground" operation centre, and auxiliary electronics, such as the multiplexing system;
b) The locomotion system formed by two grippers attached to the robot's body, which are driven by AC motors with Harmonic Drive reductors. This permits 3D movements along complex structures;
c) The sensorial platform based on a camera and a laser telemeter, for inspection operations and for the robot navigation. Figure 2b shows the ROMA robot on the lab test structure.

The robot has eight DOF kinematics:

a) Four DOFs for the elevation ($\alpha_1$ and $\theta_2$) and orientation and ($\alpha_2$ and $\theta_3$) of each gripper;

a)



b)

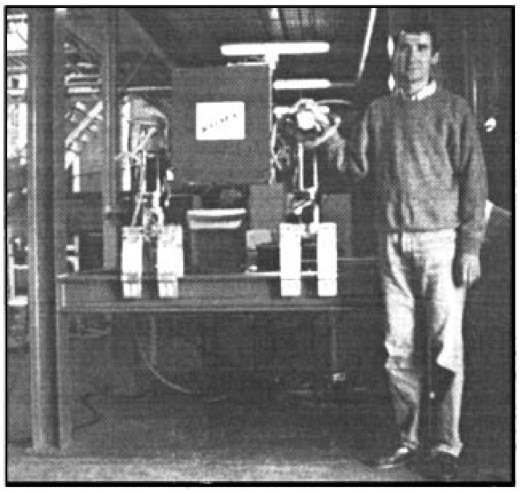


Fig. 2. ROMA robot structure: a) number of DOF, and b) real robot view.

b) One DOF for the rotation ($\theta_1$) of the gripper 2,
c) One DOF for the "extension" (1) of the body; and
d) Two DOFs for the grippers ($d_1$ and $d_2$).

In this way the robot structure has a non-redundant kinematics and a minimum possible number of DOFs for 3D complex movements. This is one of the main specifications because each DOF (including its actuator) increases the total robot weight and consequently increases the required torque to move the overall weight of the robot. Table I summarizes the main characteristic of the ROMA robot.

Figure 3 illustrates a schematic diagram of the hardware and control architecture of the robot, as well as the ground operation centre responsible for the tele-operation, monitoring and programming. Each axis is driven by a brushless AC motor through a PID adaptive controller implemented by a multiaxis control board (PMAC). This card is equiped with its own microprocessor which is dedicated to all eight control loops responsible for the robot motion. This leaves the computer CPU free to process other tasks, such as handling the camera image, the laser telemeter, and communicate the necessary information to the ground computer through the *Aironet* card. More details about the structure and design are given in reference 12.

The "ground computer" sends the commands to the robot at the task level,[13] which are divided into simple movements for each joint, in order to get a completed gait. It is possible to send low level orders, like move a joint, release a break, etc. As illustrated in Figure 3, there is only one amplifier, due to weight reasons during the movement of the motors. For this reason, four special multiplexing cards have been designed to allow the selection of the different signals involved in every axis movement.

Each axis controller, in addition to the common PID, includes two feedfordward loops (related to velocity and acceleration), with the possibility of changing any parameter on-line, even during the motion of the axis (Figure 4). Due to the fact that gravity factors (angle of elevation of the robot, moving up or down, etc.) have a high influence on the quality of motion, an adaptive control scheme has been implemented. All adaptive gain scheduling is used for deciding on-line all the parameters of the controller, which are found experimentally. In order to facilitate this search, the motor-link system has been identified using (l), proposed by Abderrahim,[14] where $M$ is the link mass and $l$ is the distance between the axis of rotation and the link centre of gravity in the perpendicular plane to the axis of rotation, $T$ is the torque delivered by the motor, $I$ is the inertia seen at the motor axis ($I_{rotor}+I_{load}$), $B$ is the viscous friction coefficient, $f_1$ and $f_2$ are the coulomb friction coefficients, and $\theta(t)$ is the angular position of the motor at time $t$. With this equation (1) the gravity loading is not ignored, and it is taken into account during the motion control.

$$T(t)=I\frac{d^2\theta(t)}{dt^2}+B\frac{d\theta(t)}{dt}+Mglsin(\theta(t))+$$
$$+f_1 sign(\dot{\theta}(t)+|\dot{\theta}(\mathrm{t})|)+f_2 sign(\dot{\theta}(t)-|\dot{\theta}(\mathrm{t})|) \tag{1}$$

Starting from equation (1), the values of the gain table were calculated in three stages: First, using the measured values, shown in Figure 5, and the MATLAB tools, the parameters of the joint were identified. Second, the identified model was used during a simulation exercise to allow the tuning of the P1D parameters. The last part consists of the implementation of the PID controller in the robot and achievement of fine tuning to the parameters.

An example of a gain table can be seen in Table II where the controller associated to axis 3 fixes these values. This is the case where the gripper of the robot is moving up or down relative to the body of the robot. In the case where the gripper is attached to the metallic structure and the rest of

Table I: The main characteristics of the ROMA robot.

| Characteristics | Values |
|---|---|
| Elevation range | $-10° \rightarrow +190°$ |
| Orientation range | $-190° \rightarrow +190°$ |
| Rotation range | $-190° \rightarrow +190°$ |
| Extension range | 500 mm |
| Grippers extension range | 300 mm |
| Robot weight | 75 kg |
| Maximum linear velocity | $\approx 1$ m/min |
| Autonomy | $\approx 3$ h |

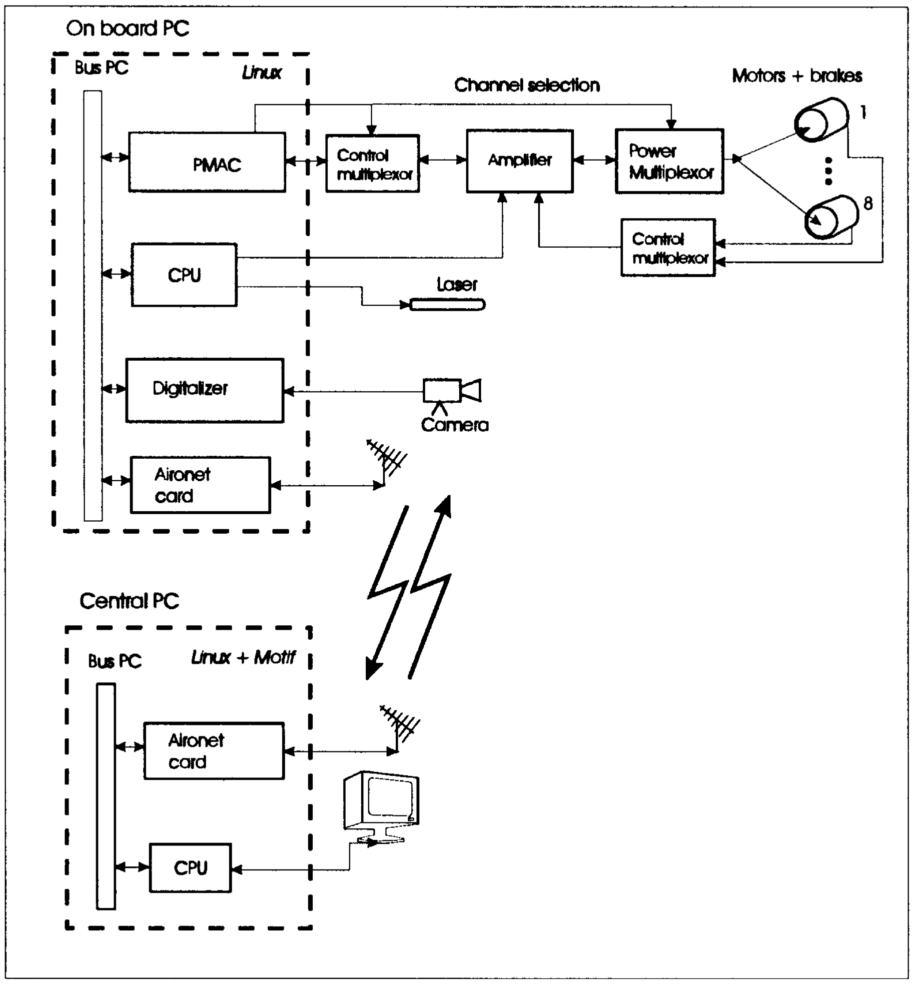


Fig. 3. Control architecture of the ROMA robot and its remote operation base.

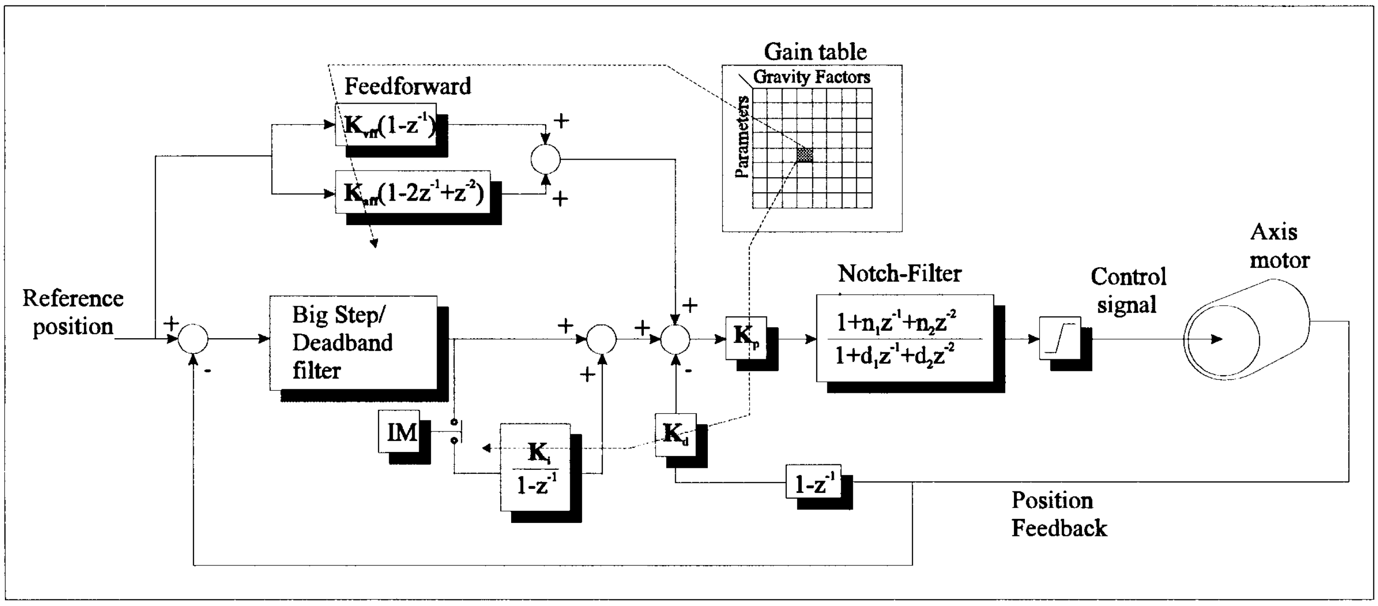


Fig. 4. Controller of each axis.

the body rotates up or down about axis 3, the table would be different. Therefore, most of the motor axes, except 1 and 8, have various gain tables depending on the nature of the movement. The parameters in the table depend on the direction and the range of the movements.

## 3. CLIMBING STRATEGY

The robot has been designed taking into account not only its mobility among the complex 3D environment but the performance of these movements with a minimum energy consumption which permits one to maximize the robots

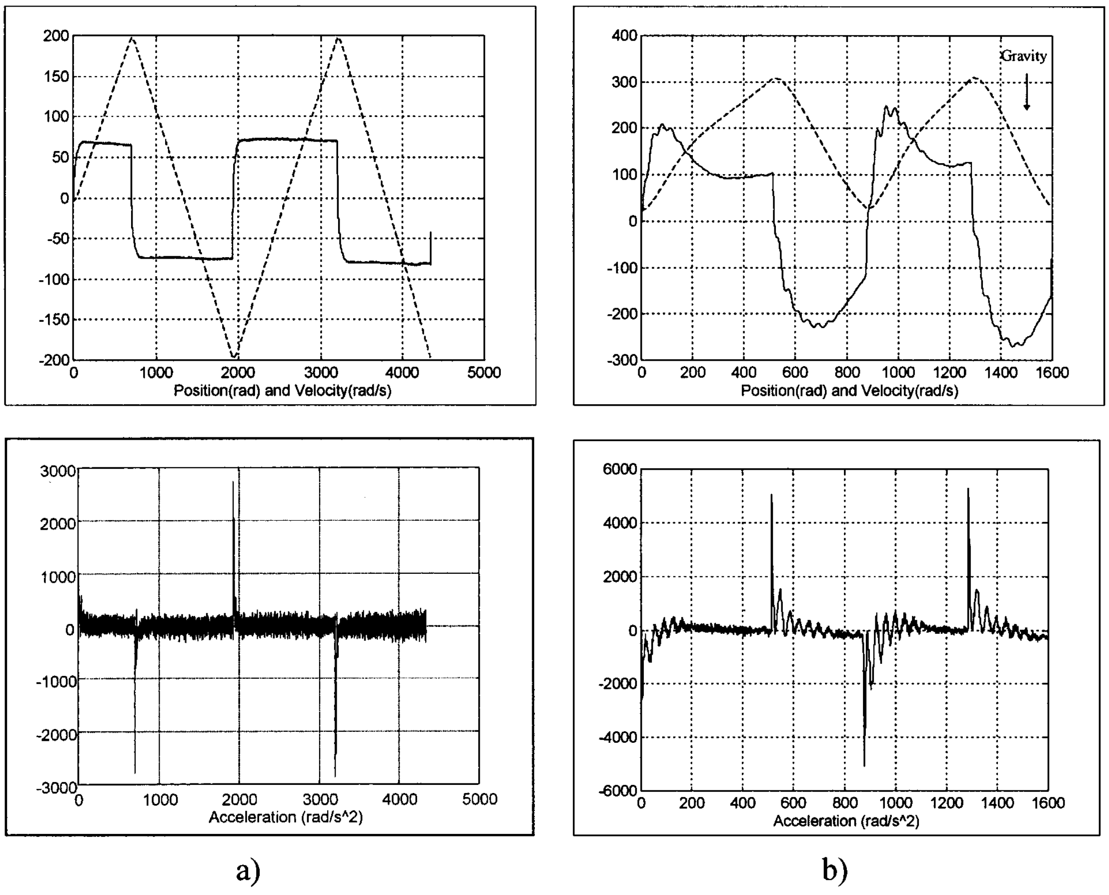


Fig. 5. Position, velocity and acceleration profiles: a) Axis 2 without gravity, and b) Axis 3 with gravity factors.

autonomy. The robot autonomy is one of the most important specifications. For this purpose the robot movements were analysed taking into consideration the minimum power consumption. These are three different types of movements:

a) 1D movement along the beam or column (Figure 6);
b) 2D movements in the horizontal or vertical planes of the structure (Figures 7 and 8); and
c) 3D movements which allow the robot to change position from one plane of the structure to another (Figure 9).

To ensure robot movement in the structure, different sequences of elemental movements are possible. These are 1-DOF simple movements designated as “movements primitives” (MP). If the minimum energy consumption criterion is taken into account, it is necessary to select among all the possible MPs the one with minimum energy consumption for the case under consideration. For example,

Table II: Gain table for axis 3.

| Range (°) | Dir. | $K_p$ | $K_d$ | $K_i$ | $K_{vff}$ | $K_{aff}$ |
|---|---|---|---|---|---|---|
| 0–45 | ↑ | 2500 | 725 | 40 | 90 | 35 |
| 45–90 | ↑ | 3800 | 725 | 40 | 75 | 35 |
| 90–135 | ↑ | 3400 | 400 | 40 | 50 | 35 |
| 135–180 | ↑ | 2500 | 400 | 40 | 40 | 35 |
| 0–45 | ↓ | 1500 | 200 | 40 | 90 | 35 |
| 45–90 | ↓ | 1000 | 200 | 40 | 75 | 35 |
| 90–135 | ↓ | 1300 | 140 | 40 | 50 | 35 |
| 133–180 | ↓ | 1500 | 140 | 40 | 40 | 35 |

there are three different possibilities to perform the 1D forward movement on the top of a metallic beam (Figure 10):

(1) **Dragging**, like a “caterpillar” with the following sequence:

a) release one of the grippers,
b) extend the robot body,
c) lock the gripper to the beam,
d) release the other gripper,
e) shrink the robot body, and
f) lock the second gripper to the beam.

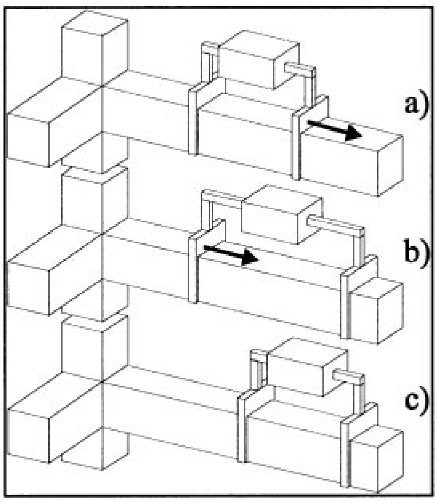
(2) **Horizontal rotation**, like a “centrifuge” with the following sequence:

a) release one of the grippers,
b) small upwards rotation of the robot (elevation) about the horizontal axis,
c) 180° rotation of the robot around the vertical axis,
d) small downwards rotation of the robot about the horizontal axis, and
e) lock the gripper to the beam.

(3) **Vertical rotation**, like an “acrobat” with the following sequence:

a) release one of the grippers,
b) 180° upwards rotation of the robot about the horizontal axis,
c) 180° down rotation of the robot gripper around the horizontal axis, and
d) lock the gripper to the beam.

a)



b)

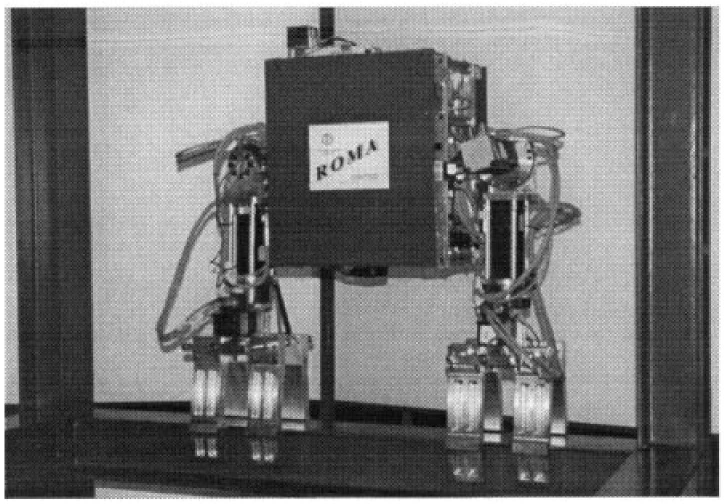
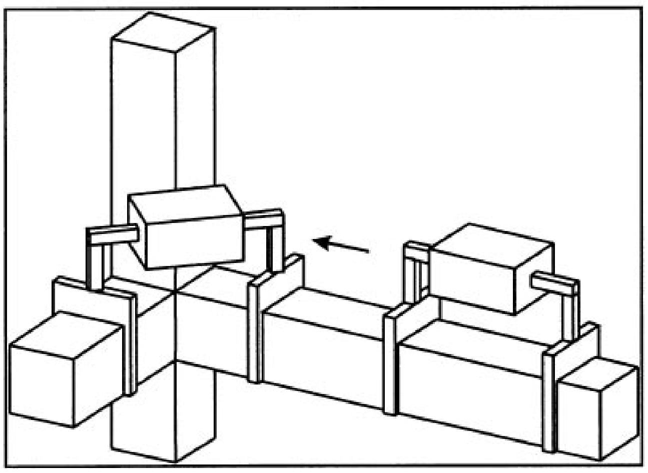


Fig. 6. 1D movement along the beam: a) general scheme, b) robot view.

a)



b)

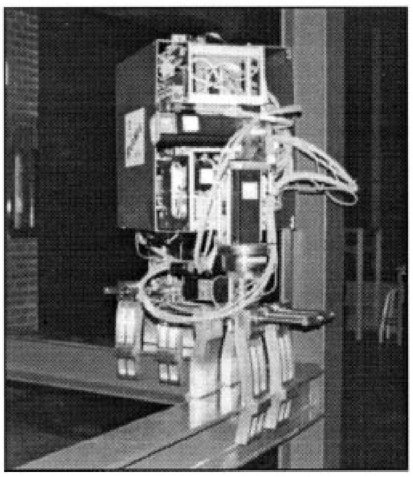
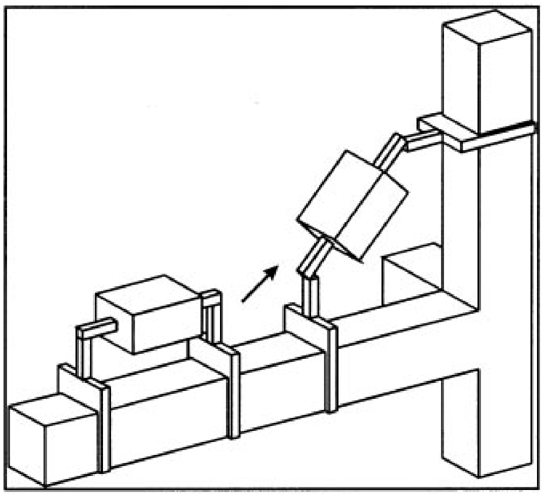


Fig. 7. 2D horizontal robot movement: a) scheme, and b) robot view.

a)



b)

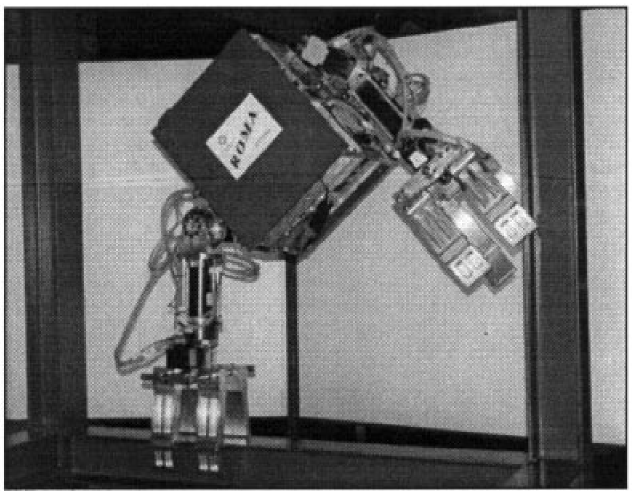


Fig. 8. 2D robot vertical movement: a) scheme, and b) robot view.

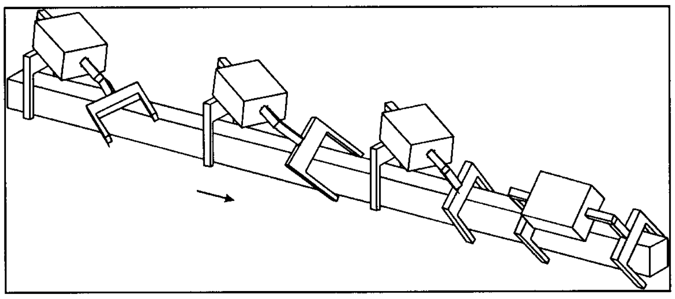


Fig. 9. 3D robot movement.

As shown in Figure 10, all three movements are performed by a combination of 7 MP (A, B, C, D, E, F and G). This is why it is necessary to compute the energy consumption for each of them and then select the minimum energy consumption strategy for the forward movement. Table III shows the energy consumption and the robot speed along the beam for the above mentioned alternatives. This calculated energy is the sum of the power consumption of each motor involved in each specific movement. As result of this study, including the gravity force, the best compromise between energy consumption and robot speed is dragging, being the accepted MP A and B.

For the 2D movements between different beams the sequence is similar to the above adding to it the elevation and and/or orientation of the grippers. Finally, most of the complex 3D movements are ensured by combining all individual movements, including the gripper rotation. As in the above mentioned example, for 2D and 3D movements the minimum energy consumption MP is selected.

## 4. ENVIRONMENT MODELLING

The environments of the ROMA robot are mainly metal-closed structures of buildings or bridges. To plan the robots path in this type of environment it is necessary to design the environment model, which is based on "environment primitives". There are two main primitives: a) Beam primitive, which includes beams and columns of the structure; and b) Beams-cross, which join the beams and columns to form 2D or 3D structures. An example of the 2D open beam-cross structure is presented in Figure 11, where the three level coding of the primitives ("a.b.c") has the following form: "a" is the number of the beam, "b" is the number of the face of the beam, and "c" is the reference point on the beam (starting, centre or end point). The environment is modelled as a graph, called "environment graph", using the described primitives shown in Figure 11.

## 5. PATH PLANNING STRATEGY

The common inspection task is performed through the whole metallic structure, which makes it necessary to plan the robot path with the following requirements:

a) Compute the round trip path taking into account the power consumption;
b) Perform the inspection of all faces of all beams, i.e. visit all the nodes and transitions of the environment graph.

These two requirements entail solving a TSP-like problem:

(i) Starting and finishing the inspection task in the same point;
(ii) Transiting, if possible, only once along each beam face;
(iii) Optimizing the robot consuming energy.

To solve the TSP problem it is necessary that the graph should be Hamiltonian, i.e. the graph in which it would be possible to visit all the nodes without repeating any. Normally, to determine if the graph is a Hamiltonian, a difficult non-polynomial (NP) problem should be solved. This is why usually the graph is transformed into a complete one, in which every vertex is adjacent to every other one, which is always a Hamiltonian.[15] However, in our case, the environment graph is easier to transform directly into a Hamiltonian which is less redundant than the complete one. For this purpose the following developed procedure is applied:

a) Considering that each beam primitive will be visited an even number of times;
b) Introducing a central point beam primitive virtual node (node "a.b.2" in Figure 11) for each beam face;
c) Introducing the remaining transitions in the beam-cross primitives in order to connect each node to all other nodes of the beam-cross.

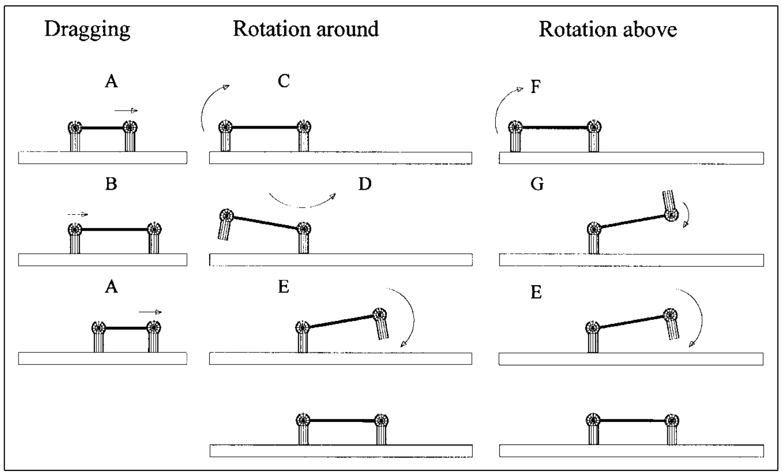


Fig. 10. Forward robot movement alternatives.

Table III: Comparative study of different 1D forward movement.

| Type of movement | MP | Energy (J) | Speed (m/min) |
|---|---|---|---|
| Dragging | A, B | 392 | 1 |
| Rotating around | C, D, E | 784 | 0.4 |
| Rotating above | F, G, E | 940 | 0.25 |

To compare the efficiency of the developed ROMA path planning algorithm the optimum and "best heuristics" ones were computed. The optimum algorithm is based on the "Branch & Bound 1-Tree" technique developed by Hurwitz.[16] Due to the fact that the TSP problem is an NP, the computing time of this optimum algorithm increases in an exponential way. This makes it impractical to implement it for the ROMA robot environments which have a large number of nodes (more than 100).

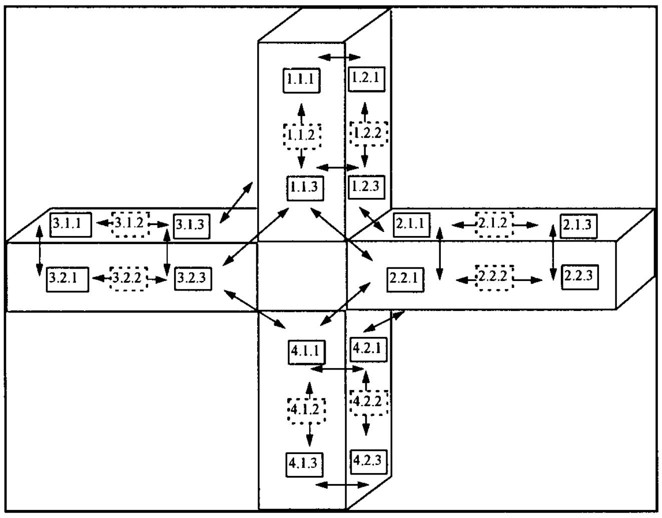


Fig. 11. 2D environment formed by one cross-beam and four beams.

To solve this problem several traditional heuristics have been checked in order to obtain a sub-optimum solution. The following "best heuristics" are checked:

a) "Nearest addition" which consists of adding to an initial tour (close robot path) a nearest node which is not in the tour;
b) "Farthest insertion" which consists of inserting to an initial tour the farthest node;
c) "Easytour" which follows the order of definition of the nodes, etc.

To improve the results of these heuristics in the final stage of each algorithm an "improver" is proposed.[17] It consists of a local iterative search which exchanges the nodes positions in each heuristic tour in order to optimize the energy criterion. In this case, the exchange strategy is based on a variation of a well known Lin and Kernighan algorithm.[18] Nevertheless, despite that the "best heuristics" has a good energy optimization, for a large number of nodes (about 900) this algorithm is also impractical.

The compromise between computing time of the path planning algorithm and energy consumption for a large number of nodes was solved by the ROMA heuristic algorithm. It is based on a "Nearest Neighbour" algorithm,[19] modified and adapted to our case. It visits all beam faces (nodes "a.b.2") without repeating any, and uses the nearest neighbour node criterion to transit between them. Figures 12 and 13 show the effectiveness of the proposed path planning algorithm which can be computed in quasi realtime for a large number of nodes. Figure 12 points out the different compositions of columns and beams that have been considered in the resolution of different algorithms, and Figure 13 shows the different results using the estimated energy among the movements presented in Table III.

During the movement, the robot's battery level can be checked continuously. Using this data the robot predicts the minimum battery level necessary to perform the returning

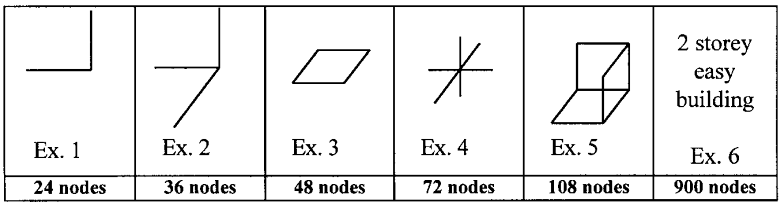


Fig. 12. Environment examples.

| *Algorithm* | *Optimum* | | *Best heuristics* | | | *ROMA* | | |
|---|---|---|---|---|---|---|---|---|
| Example | Energy (J) | Time (sec) | Energy (J) | Time (sec) | Over energy (%) | Energy (J) | Time (sec) | Over energy (%) |
| 1 | 2885 | 0.31 | 2896 | 0.38 | 0.38 | 2885 | 0.02 | 0 |
| 2 | 3633 | 1.63 | 3633 | 0.08 | 0 | 4002 | 0.05 | 10.15 |
| 3 | 4272 | 3.26 | 4480 | 0.25 | 4.86 | 4272 | 0.05 | 0 |
| 4 | 6850 | 486.52 | 7052 | 0.78 | 2.94 | 7052 | 0.3 | 2.94 |
| 5 | 11948 | 1196069 | ∝ | 1.76 | ∝ | 13055 | 0.21 | 9.26 |
| 6 | 84620* | - | ∝ | 2417.57 | ∝ | 94916 | 2.09 | 12.16 |

Fig. 13. Results of the path planning algorithms (* Held-Karp estimation).

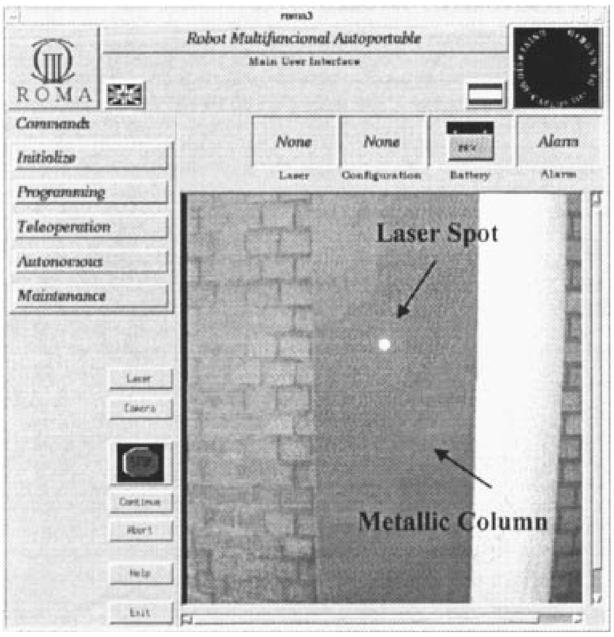


Fig. 14. The Main Window of the GUI.

movement to the starting point. This prediction is based on the modified algorithm A* using the energy criteria.[20]

## 6. GUI AND ROBOT PROGRAMMING

Robot teleoperation, supervision and monitoring are performed through a specially developed Graphical User Interface (GUI); its main window is shown in Figure 14. The GUI is homed in the "ground" operation centre, which communicates with the robot on-board computer via radio Ethernet. The GUI has been designed with an open and modular philosophy allowing future reusability and scalability. New modules can be added at any time to enhance the functionality and user-friendliness of the system, and also to add new features and programming possibilities.

The GUI includes the main following features:

a) Teleoperated or autonomous robot movement mode;
b) Status of the robot, including its position in the structure and its kinematic configuration, the batteries level, alarms, etc.;
c) Sensorial information, which consist of the actual camera image and the laser telemeter latest reading.

In this way the User Interface is part of the architecture of the control system of the ROMA robot as a whole. The interface supplies an online visualisation of the environment in front of the robot, where the camera and the laser are directed. The camera image placed on the lower right side of the main User Interface allows a realistic telepresence. The operator can save the image of the camera at any time and store in a file for later examination and analysis.

The communication between the GUI and the control programmes on the on-board computer is performed through three channels (sockets) on a client server basis. The server, which is the main control programme in the on-board computer, is first executed to wait for the connection of the client programme that could be running on any computer within the same radio network. Thereafter, the server enters in a loop and watches for any commands sent by the client and, in turn, dispatches them to their corresponding destination.

The GUI and its associated programmes run on two parallel processes in the ground computer and interchange information via signals and shared memory. Figure 15 illustrates the interconnection of the GUI modules. The main process containing the MOTIF graphical part is responsible for the interaction with the user and sends

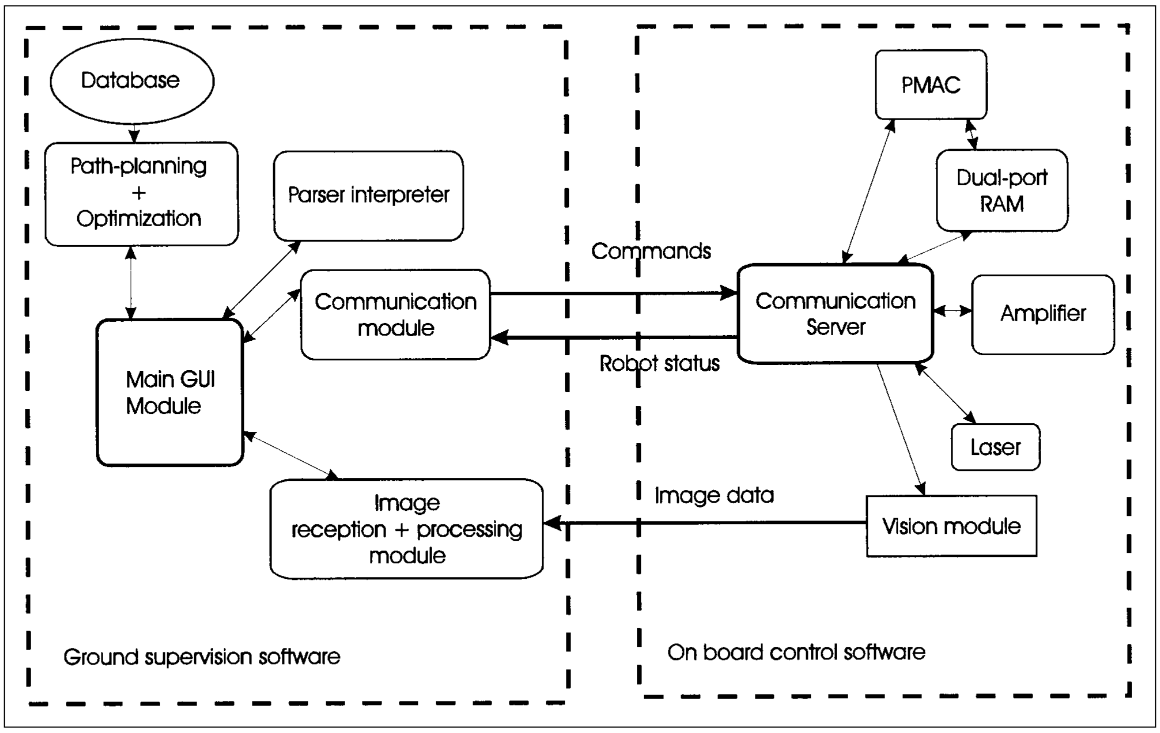
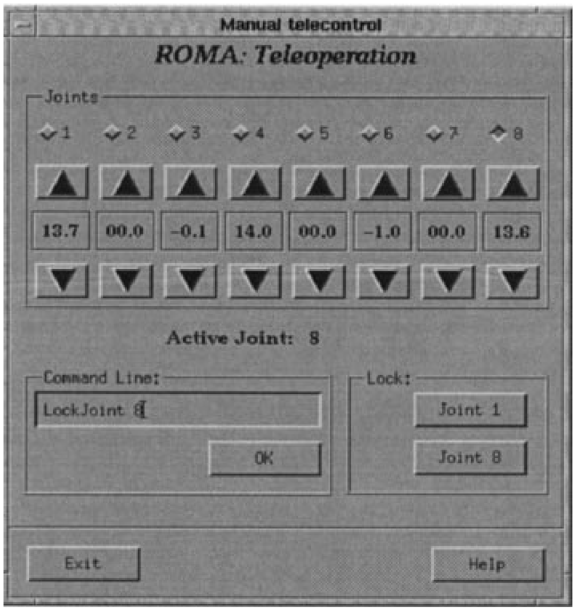


Fig. 15. Software architecture.



13.7 | 00.0 | -0.1 | 14.0 | 00.0 | -1.0 | 00.0 | 13.6

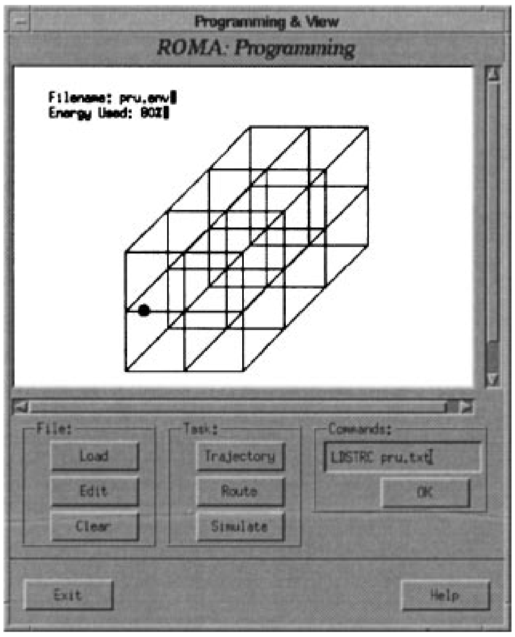
Active Joint: 8

LockJoint 8

Fig. 16. Teleoperation Interface.

orders to the onboard server via the first channel. The second channel configured to asynchronous mode is used for reading incoming information from the server. The reading is triggered only when data is available (received). The module responsible for receiving the camera image data from the onboard computer runs as a separate process, which continuously read from the frame grabber via the third channel. Every time a complete frame data is received the data is processed to create the image and put in a common area accessible by the main module, which receives a signal on the completion of the image creation. Thus, the main module triggers the function responsible for refreshing the camera image on the main GUI.

The teleoperation of the robot is performed through using a pop-up simulated pendant as shown in Figure 16. It offers a simple and intuitive device which permits to teleoperate



Filename: pru.env

Energy Used: 80%

LDSTRC pru.txt

Fig. 17. Programming Interface.

```
ROBOT "roma.rob"
# Task Name
NAME "edif900.tsk"
###############################
BEGSTC
# Initial Node
N, 1, 0,0 , 0
# First Line
rn, 1, 1, 1, 2,  3, DX, 1
# First Floor
rn, 1, 4, 1, 5,  2, Dy, 1
# Horizontal (X) Beams
B,  1, 1, 2
rb, 1, 1, 1, 2, 2, 1
rb, 1, 3, 1, 4, 2, 4
# Horizontal (Y) Beams
b, 10, 1, 5
rb, 10, 10, 1,11, 3, 1
rb, 12, 15, 1, 16, 1, 4
ENDSTC

# TASK DEFINITION
IPOS, 1, 1, 1
TFACES
```

Fig. 18. Program of ROMA Robot.

the robot axis by axis, or through sending more complex movement commands via the command line field. A list of these commands includes: *MoveForward x, Move-Backwards x, LockJoint x, GoEndBeam, RotateRight, ChangeFaceRight, Laser*, etc, where *x* is a numerical parameter. When the movement commands are received by the on-board control programme, they are first parsed and then the corresponding sequence of motion primitives is looked up in the data base. The parameters for each primitive are calculated using the robot internal inverse kinematics model as required.

For robot security it is impossible to release one gripper until the other one is safely locked to the beam. The motors' electromechanical brakes maintain the gripper locking. Teleoperation commands are not executed until previous commands have been terminated or aborted. The new joint values and other data are sent regularly to the ground computer and the information is updated on the Teleoperation Interface.

The programming of a given task of the robot is initiated through the Programming Interface (PI) shown in Figure 17. The first step of this process starts with the definition of the inspection environment using a graphical editor. This can also be achieved using a special language. Second, a decision is made about the selection of the points to visit on the defined environment, or visit all points of the structure and define the task at these locations. Third, automatic energy calculation and generation of the optimal path take place. The latter is followed by an automatic coding of the robot program for the defined task. The coding of the program can also be done by hand, but in complex and large structures it is more appropriate to perform this tricky task automatically and let the ROMA tools take care of this process. Figure 18 illustrates a listing of the program to define the base of the structure shown in Figure 17.

Once the path is determined and the program is generated the PI offers a simulation facility, which visualises how the robot would move through the defined structure and also show the simulated energy consumption on the viewing window, as illustrated in Figure 17.

The GUI is equipped with the facility of a low level programming to issue direct commands to the multiaxis servo controller of the robot in its low level programming language. It serves for fault diagnosis, resetting, calibration, changing control parameters and assists in performing other important maintenance functions, such as enquiring about some states of the motors and the current control gains for each of them.

## 6. CONCLUSIONS

It has been shown that the developed ROMA robot has several important advantages in the field of climbing robots: It has a very dextrous kinematic structure, which allows it a variety of movements to access complex structures. It uses a powerful onboard sensory system, which permits sufficient positioning accuracy and error recovery. Moreover, the main advantage is the fact that it is designed as an autonomous system able to move freely in complex environments. These movements are performed at a very high security level.

A novel metholodogy for the design and path planning of a robot able to navigate in complex 3D environments, such as automatic inspection of bridges, buildings, metallic structures, among many others has been used to develop the ROMA robot.

The advanced Graphical User Interface, developed for the ROMA, allows an effective and user friendly operation of the robot with minimum operator training. It has been designed with an open and modular structure allowing future reusability and scalability.

The initial experiments using the ROMA robot confirm its advantages and usefulness in executing inspection operations in complex environments. There are clear indications that this type of robot could replace the human operators in dangerous tasks in the near future.

**Acknowledgements**
This work has been partially funded by the Spanish government agency CICYT under project TAP95-0088. The authors would like to acknowledge the technical support of A. Jardón, E. Jiménez, C. Palazuelos, J.A. Campo and F. Manera and also the company of APTECA for its help in the mechanical development.

## References

1. E. Gamboa, C. Balaguer, A. Barrientos, R. Saltaren and E.A. Puente, “Robot assembly system for the construction process automation” *IEEE International Conference on Robotics and Automation (ICRA’97)*, Albuquerque, USA (1997) pp. 46–51.
2. L.F. Peñin, M. Ferre, A. Barrientos and R. Aracil, “Teleoperated system for live power lines maintenance: ROBTET” *IEEE International Conference on Robotics and Automation (ICRA’98)*, Leuven, Belgium (1998) pp. 2110–2115.
3. P. Gonzalez, M. Armada and M.A. Jiménez, “Development of walking machine at IAICSIC” *IEE Workshop in Climbing and Walking Robot (CLAWAR’96)*, Portsmouth, UK (1996) pp. 12/1–12/3.
4. P.G. Backes, Y. Bar-Cohgen and B. Joffe, “The multifunctional automated crawling system (MACS)” *IEEE International Conference on Robotics and Automation (ICRA’97)*, Albuquerque, USA (1997) pp. 335–340.
5. J.J. Kerley, E. May and W. Ecklund “The climbing crawling robot” *Industrial Robot* **19**, No. 4, 32–33 (1992).
6. B.L. Luk, D.S. Cooke, A.A. Collie and T.S. White, “An arachnid robot for walking and climbing in disordered hazardous environments” *Proceedings of the International Conference on Intelligent Autonomous Systems. IAS-4*, Karlsruhe, Germany (1995) pp. 708–712.
7. F.W. Bach, M. Rachkov, J. Seevers and M. Hahn, “High tractive power wall-climbing robot” *Automation in Construction* **4**, No. 3, 213–224 (1995).
8. D.S. Cooke, N.D. Hewer, T.S. White, S. Galt and B.L. Luk, “Implementation of modularity in Robug IV” *First International Symposium on Mobile, Climbing and Walking Robots (CLAWAR’98)*, Brussels, Belgium (1998) pp. 185–191.
9. V. Gradetsky, “Wall climbing robots: evolution to intelligent autonomous vehicle” *First International Symposium on Mobile, Climbing and Walking Robots* (*CLAWAR’98*), Brussels, Belgium (1998) pp. 53–60.
10. R. Kamei, K. Ogasawara and R. Katamura, “Development of multi-purpose mobile robot capable of travelling along columns and beams” *11th International Symposium on Robotics and Automation in Construction (ISARC’94)*, Brighton, UK (1994) pp. 487–493.
11. C. Balaguer, E. Gambao, A. Barrientos, R. Saltarén and E.A. Puente, “Computer-aided methodology for design of robots in construction industry applications” *14th International Symposium on Robotics and Automation in Construction (ISARC’97)*, Pittsburgh, USA (1997) pp. 294–298.
12. C. Balaguer, J.M. Pastor, A. Giménez, V.M. Padrón and M. Abderrahim, “ROMA: A multifunctional autonomous self-supported climbing robot for inspection application” *3rd IFAC Symposium on Intelligent Autonomous Vehicles*, IAV’98, Madrid, Spain (1998) pp. 357–362.
13. A. Giménez, C. Balaguer, J.M. Pastor, V.M. Padrón and M. Abderrahim, “Design and path-planning for a climbing robot able to travel along 3D metallic structures” *First International Symposium (CLAWAR’98)*, Brussels, Belgium (1998) pp. 161–166.
14. M. Abderrahim, “Modelling of robotic manipulators” *Ph.D. Thesis* (Glasgow University, 1996).
15. F. Glover and A.P. Punnen, “The travelling salesman problem: new solvable cases and linkage with the development of approximation algorithms” *J. Operational Research* **48**, 502–510 (1997).
16. C. Hurwitz, “TSD: performance and description of heuristic” *Ph.D. Thesis* (1992).
17. E.L. Lawler et al., *The Travelling Salesman Problem* (John Wiley & Sons, New York, 1985).
18. S. Lin and B.W. Kernighan, “An effective heuristic algorithm for TSP” *Operational Research* **21**, 498–516 (1973).
19. D.J. Rosenkratz et al., “An analysis of several heuristics for the TSP, SIAM” *J. Computing* **6**, 563–581 (1977).
20. J.A. McHugh, *Algorithmic Graph Theory* (Prentice-Hall, USA, 1990).