A Universal Task-Level Ground Control and Programming System for Space Robot Applications

The MARCO Concept and its Application to the ETS-VII Project

B. Brunner, K. Landzettel, G. Schreiber, B.M. Steinmetz, G. Hirzinger

DLR Oberpfaffenhofen
German Aerospace Center
Institute of Robotics and System Dynamics
D-82234 Wessling
Bernhard.Brunner@dlr.de

Abstract

The paper outlines the main features of DLR's ground control station for space robotics applications. It combines sensor-based task-level *teleprogramming* (as the basis for autonomy) with the features of *teleoperation* and shared autonomy. The hierarchical system structure is shown as well as the flexibility in programming and controlling each kind of space robotics application. The teaching by showing approach is the key to a easy-to-use programming interface at different levels of space robot controlling. This approach has led to a modular task-directed programming scheme, called *M*odular *A&R C*ontroller (MARCO), which provides a very flexible architecture to adapt the application-specific requirements to a given controlling scheme. To demonstrate the power of MARCO, we describe the results of the *GETEX* experiment, which has been performed in April '99 at the first free-floating space robot on NASDA's ETS-VII satellite[1].

Introduction and Overview

After the success of ROTEX, the first remotely controlled robot in space, DLR has focused its work in telerobotics on the design of a high-level task-oriented robot programming system, which is characterized as *learning by showing in a virtual environment*. The goal was to develop a unified concept for a flexible, highly interactive, on-line programmable teleoperation ground station as well as an off-line programming system, which includes all the sensor-based control features already tested in ROTEX[2], but in addition provides the feasibility to program a robot system at an implicit, task-directed level, including a high degree of on-board autonomy.
This means that a non-specialist user like a payload expert will be able to control a remote robot system e.g. for internal servicing within a space station, i.e. in a well-known environment. This requires a sophisticated man-machine-interface, which hides the robot control details and provides an *intuitive programming interface*. For that reason, we have developed a network-transparent graphical user interface, based on the quasi-standards VRML and Java. Using a task-level protocol is the preferable method to remotely operate robots as it demands only extreme narrowband connections and does not bother about large time delays. The user interacts via the virtual view with the real environment, as (s)he has only to define, what (s)he wants to do, not how it has to be done. Supported operations are e.g. open/close a door/drawer, pick&place an orbital replaceable unit etc.
However, for external servicing with free-flying robots, e.g. the repair of a defect satellite, high interactivity between man and machine is required, because the remote environment will be mainly unknown. All the well-known problems w.r.t. teleoperation under long time delays can only be solved by the *predictive graphics* approach. One of the main requirements is the feasibility to update the simulated world according to the real world as well as to provide *local autonomy based on intelligent sensor data processing* without large a priori knowledge.
To fulfill the requirements of both application fields, we have developed a 2in2-layer model[3], which represents the programming and control structure from the executive to the planning level in a hierarchical way. According to the application requirements the user can use the necessary and sufficient level of commanding and programming or switch between the *different layers* especially in case of failure detection and recovery.
This control and programming system may be used for several applications: the task-oriented non-expert programming layer is demonstrated by the implementation of a net-browser VRML plugin[4] to control a prototypic intravehicular environment, an extension of the ROTEX workcell, at the task level without any knowledge of robotics.
It seems straight-forward kind of work to make the MARCO programming and control environment applicable for the Technology Exposure Facility (EuTEF) of the International Space Station.
As a realistic test, the ground control facilities of our system were used in April '99 to remotely control the Japanese ETS-7 robot, the first robot in free space. The main goals of DLR's contribution within the GETEX project were the utilisation of the world model update concept using the real video images, to verify our task-level programming approach including on-board autonomy via selected image features and force-torque information as well as the verification of the dynamic simulation due to the interactions between robot and carrier.
Our cooperation with NASDA w.r.t. to the dynamics verification was one important step towards *a free-flying service satellite*. For more details see [5]. In our lab the semi-autonomous telemanipulation feature of the ground control and programming system is used for the ESS (experimental servicing satellite) scenario, where a free-flying telerobot is supposed to approach, inspect und repair a malfunctioning satellite, e.g. the TV-Sat-1, where after launch one solar panel

Proc. Fifth International Symposium on Artificial Intelligence, Robotics and Automation in Space, 1-3 June 1999 (ESA SP-440)

had not opened. A special, in-house-developed capture tool, containing 6 laser range finders, a wrist-mounted force-torque sensor and stereo camera, allows, in combination with the dynamics behavior prediction, the fully autonomous servoing, insertion and capturing of apogee motors, which are typical for any geostationary satellite.
Furtheron, a *robonaut* system is proposed which can take on or share intravehicular payload activities, so far carried out by astronauts. Due to the fact that the payloads are optimized for human operation, the robot endeffector must be able to interact with this human-adapted environment. We have equipped our 7-axes light-weight-robot with a human-like 4-finger-hand to handle devices, which are standard in a human environment, and with a 3-axes gantry to reach all positions within an experimental spacelab setup. The control and programming system as used for the above applications is flexible enough for usage in this multi-degrees-of-freedom system. In extension to the former application a data glove is used for teleoperating and programming human-like grasp and manipulation actions.
It should be mentioned that our programming system is immediately applicable to *planetary rovers* as well as to terrestrial service robotics: instead of the gantry a mobile platform is used to implement a „butler" robot, which will be able to perform helpful tasks in an ordinary environment.

The MARCO system

The goal for the development of our high-level programming system was to design a *unified concept* for a flexible, highly interactive, on-line programmable teleoperation station as well as an off-line programming tool, which includes all the sensor-based control features as tested already in ROTEX, but in addition provides the possibility to program a robot system on an implicit, task-directed level.
A non-specialist user – e.g. a payload expert – should be able to remotely control the robot system in case of internal servicing in a space station (i.e. in a well-known environment). However, for external servicing (e.g. the repair of a defect satellite) high interactivity between man an machine is demanded. For that reason the design of our programming system is based on a 2in2-layer-concept, which represents the *hierarchical control* structure from the planning to the executive layer:

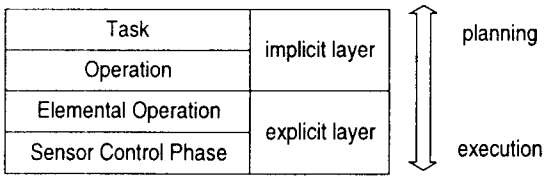

Figure 1 2in2-layer-model

On the implicit level the instruction set is reduced to **what** has to be done. No specific robot actions will be considered at this task-oriented level. On the other hand the robot system has to know **how** the task can be successfully executed, which is described in the explicit layers.

Reflex (Sensor Control Phase)

At the lowest level of the MARCO system the sensor control mechanism is active. These so-called reflexes guarantee the *local autonomy* at the remote robot's site via using sensory data processing algorithms in an extensive way. The teaching by showing paradigm is used at this layer to show the reference situation, which the robot should reach, from the sensor's view: in the virtual environment we store the nominal sensory patterns and generate appropriate reactions (of robot movements) on deviations in the sensor space.
A reflex is described by

- A controller function, which maps the deviation in the sensor space into cartesian robot move commands
- A state recognition component, which detects the controller's end conditions (success, failure)
- The constraint frame information, which supports the controller function with the task frame data to interpret the sensory data correctly (e.g. for shared control)
- A sensor fusion algorithm, if sensor values of different types have to be transformed into a common reference system (e.g. vision and distance sensors).

Elemental Operations

The explicit programming layer is completed by the Elemental Operation (*ElemOp*) level. It integrates the sensor control facilities with position and endeffector control. According to the constraint frame concept, the non-sensor-controlled degrees of freedom (dof) of the cartesian space will be position controlled

- in case of *teleoperation* directly with a telecommand device like the SpaceMouse.
- in case of *off-line programming* by deriving the position commands from the selected task. Each object, which can be handled, includes a relative approach position, determined off-line by moving the virtual end-effector in the simulation into the desired pose w.r.t. the respective object and storing the geometrical relationship between the object's reference frame and the tool center point.

It should be mentioned that the ElemOp layer aims at a manipulator-independent programming style: if the position and sensor control function are restricted to the cartesian level, kinematical restrictions of the used manipulator system can be neglegted. This implies the general reusability of so-defined ElemOps in case of changing the robot type or modifying the workcell.
A model-based on-line collision detection supervises all the robot activities. For global transfer motions a computational very fast *path planning* algorithm[6] avoids collisions and singularities in the robot's joint space.

Operations

Wheras the Reflex and ElemOp levels require the robotics expert, the implicit, task-directed level provides a powerful man-machine-interface for the non-specialist user. We devide the implicit layer into the Operation and the Task level.
An Operation is characterized by a sequence of ElemOps, which hides the robot-dependent actions. Only for the specification of an Operation the robot expert is necessary, because (s)he is able to build the ElemOp sequence. For the user of an Operation the manipulator is fully *transparent*, i.e. not visible.
We categorize the Operation level into two classes:

- An Object-Operation is a sequence of ElemOps, which is related to a class of objects available within the workcell, e.g. GET <object>, OPEN <door>.
- A Place-Operation is related to an object, which has the function of a fixture for a handled object, e.g. INSERT <object> INTO <place>. Before an Place-Operation can be activated, the corresponding Object-Operation has to be executed. <object> is the object, known from the predecessor Object-Operation, <place> the current fixture, to which the object is related.

Each object in the workcell environment can be connected with an Object-Operation and/or an Place-Operation. Because an Operation is defined for a class of objects, the instantiation of formal parameters (e.g. the approach frame for the APPROACH-ElemOp) has been done during the connection of the Operation with the concrete object instance.
To apply the Operation level, the user only has to select the object/place, which (s)he wants to handle, and to start the Object-/Place-Operation. For that reason the programming interface is based on a virtual reality (VR) environment, which shows the workcell without the robot system. Via a *3D-interface* (DataGlove or a 3D-cursor, driven by the SpaceMouse) an object can be grasped and moved to an appropriate place. If the user has moved all the objects to the places he want, the execution of the generated task can be started by doing a specific VR-hand gesture. For supervision the system shows the state of the Operation execution, i.e. the ElemOp, which is currently active. Also the position and orientation of the currently moved object is fed back.

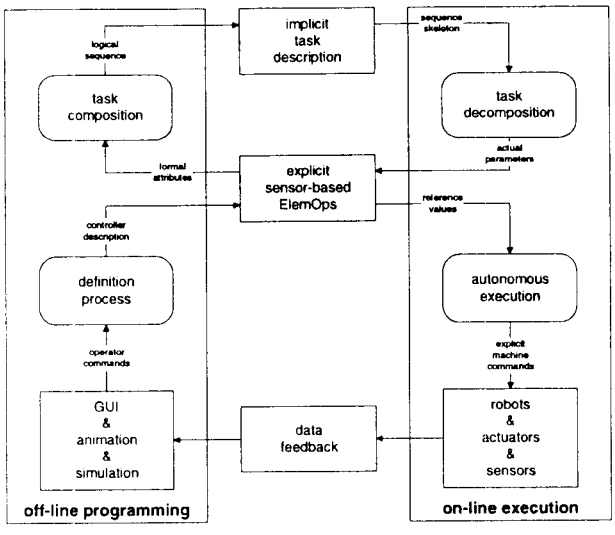

Figure 2 task-directed sensor-based programming

Tasks

Whereas the Operation level represents the subtask layer, the possibility to specify complete robot tasks must be available in a task-directed programming system. A Task is described by a consistent sequence of Operations, which are instantiated with concrete object instances (see Figure 2). To generate a Task, we use the VR-environment as described above. All the Operations, activated by selecting the desired objects or places, are recorded with the respective object or place description. An expressive example will be given in the GETEX-section.

Different graphical user interfaces

Our task-directed programming system with its VR-environment provides a man-machine-interface at a very high level, i.e. without any detailed system knowledge, especially w.r.t. the implicit layers.
To edit all four levels as well as to apply the Reflex and ElemOp level for teleoperation, a sophisticated graphical user interface based on the OSF/Motif standard has been developed (see Figure 3, screen down on the left). This GUI makes it possible to switch between the different execution levels in an easy way.

Based on the ROTEX experience we have implemented a prototypic teleoperation station, to remotely control space robotics applications by predictive graphics. Figure 3 shows different views of the simulated environment (far, near, camera view), the Motif-GUI, and the real video feedback image, superimposed with a wireframe model of the predicted state (up on the right). All the screens can be viewed in stereo mode for full immersion into the workcell environment.

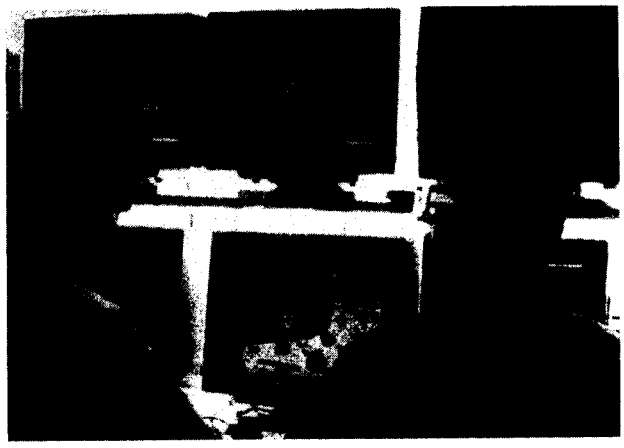

Figure 3 GUI of the universal programming and control station (MARCO)

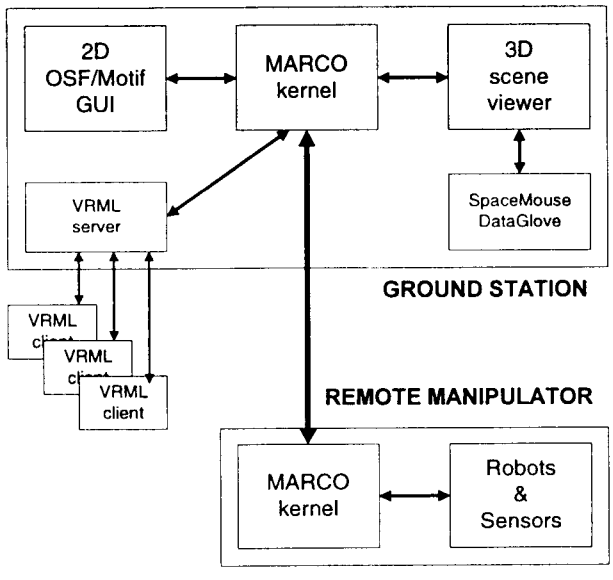

Figure 4 User Interface Structure

Java/VRML client interface

New chances towards standardization in teleprogramming arise with Java and VRML. This combination makes it possible to build easy-to-use and very cheap telerobotic stations, especially for payload users, which are not robotics experts. Via the simple Pick&Place semantics as described above, tasks can be composed and forced to a server, which will execute the desired actions. The user only clicks onto the objects, which (s)he want to handle, and starts the execution. This server also allows the *cooperative work* at the same environment: only one client is able to generate and start a desired task, all the other clients can view the current workcell state. After finishing the execution control is switched to the first

client, which sends the appropriate „I will do it"-command to the server.
In fact, the MARCO system acts as the server, so that the implicit layers of our telerobotic station system are fully programmable via a simple Java/VRML client, e.g. the ROTEX environment at our lab (see Figure 5).

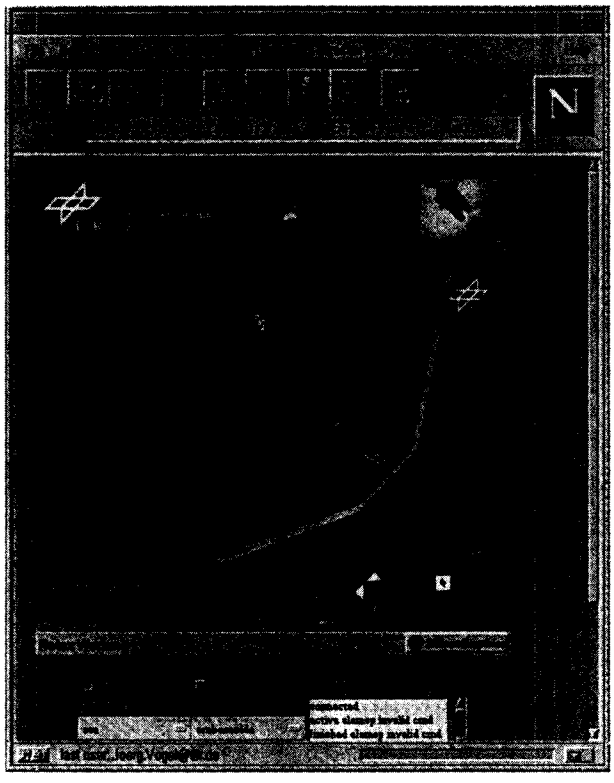

Figure 5 Java/VRML client

General scene viewer

Besides the Motif-based GUI, which provides the 2D-interface to change parameters and compose ElemOps etc. a powerful 3D scene viewer is connected to the MARCO system, which exploits the sophisticated graphics hardware to enable real time rendering and simulation of different camera aspects. This is achieved with *IRIS Performer*, but also *Open Inventor* as 3D graphics library is utilized, especially for porting the application to different hardware like PC's.
Texture mapping and highly detailed geometries are as well supported, as all different kinds of graphic devices, like SpaceMouse, dataglove, cave-like stereo projection and head-mounted displays. It is clear, that the Viewer is open for further extensions, like new devices or different scenarios.

The GETEX experiment on ETS-VII

From April 19-21, 1999 the DLR's MARCO tele-robotic and -programming system was used to control the robot arm on the Japanese ETS-VII satellite. The main goals of this GErman Technology EXperiment on ETS-VII (GETEX) were

- to verify a MARCO-based telerobotic *ground control* station for remote control of a free-floating robot, in particular
- to perform a peg-in-hole experiment, using VR methods and the „*vision&force*" control sceme, by closing sensor control loops directly on-board (force) and via the ground track (vision), thus proving MARCO's sensor-based autonomy features,
- to conduct experiments with relevance to the *dynamic behavior* of ETS-VII in free motion mode and thus to verify the existing dynamic models.

All experiments could be performed very successfully. To implement the User Interface Structure as depicted in Figure 4, we had to add some modules for communication with the Japanese ground control system, but not to change the overall MARCO ground control structure.
To check and test our interfaces as well as our MARCO control station within the ETS-VII scenario, an on-line simulator has been developed, which emulates the remotely operated robot, its command interfaces and its environment. The simulator is able to emulate all different modes, timing, the environmental interactions, and the prediction of satellite attitude while moving the robot arm. This kind of simulation has turned up to be very useful for prooving software correctness while interacting with the telerobot.

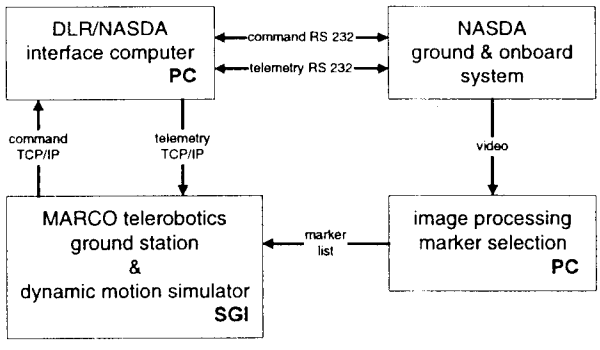

Figure 6 GETEX ground control configuration

The original MARCO kernel couldn't be implemented on-board the ETS-VII, because only the ElemOp-Layer was available on-board. All the other layers were implemented as add-ons on-ground, but this was no limitation to the verification of our task-level programming methods, because the downlink feedback data were rich enough to parametrize the next ElemOp according to the current execution state.
It should be mentioned, that the know-how, gained during the phase of adapting the MARCO system to the ETS-VII constraints, will be very useful for further space robot missions.

Figure 7 Pick TBTL by VR-hand

The MARCO system worked that well, that we decided, together with the Japanese partners, to execute the whole peg-in-hole experiment with the TBTL (TaskBoard TooL) in the automatic mode: after teach-in of the desired task sequence (pick TBTL, see Figure 7, and place it to HOLE A, see Figure 8) in the VR environment, the execution was started and performed fully automatically. No voice confirmation between each ElemOp was further needed, as it had to be done during the test runs.

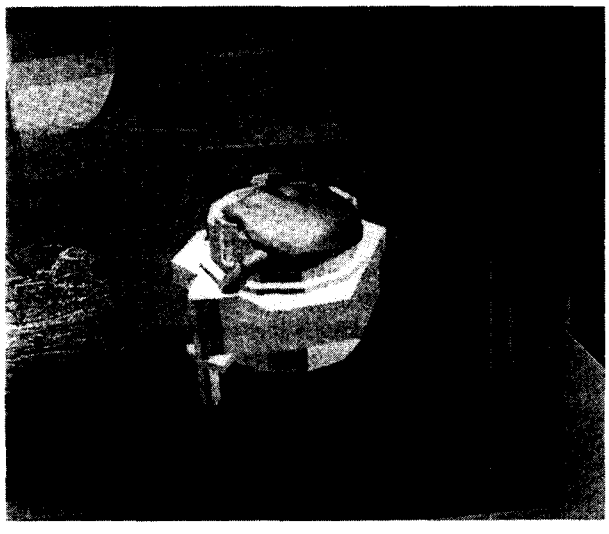

Figure 8 Place TBTL into Hole A
The real robot and object status (here the TBTL), fed back in the telemetry channel, is shown wireframed.

To get the TBTL, we first carried out a visual servoing task (at the reflex layer), which uses some marker features in the video image to control the tool center point (TCP) of the robot autonomously into the desired sensor-related pose. For that reason we have developed an approach, which doesn't need any calibration. The control law may be written as

$$v_c = \alpha\, C\, (s\text{-}s^*)$$

where $(s\text{-}s^*)$ is the vector-valued deviation between the current and the nominal sensory pattern indicating the displacement of the current robot pose x from the nominal pose x^*. v_c is the velocity command, α represents a scalar dynamic expression, at least a real constant, determining the closed control loop behavior, and C represents a projection operator used for mapping the sensor space onto the (cartesian) control space. C is determined by neural network learning or using analytical methods.

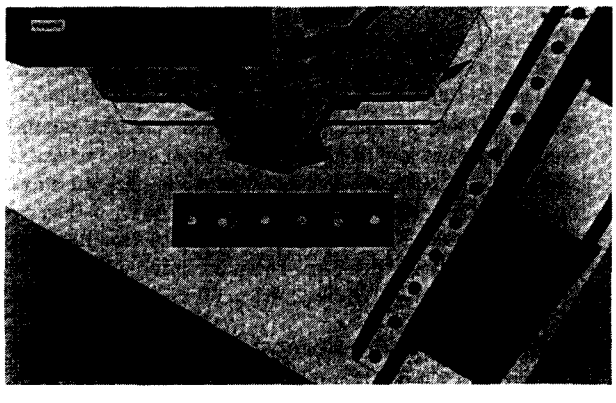

Figure 9 View out of the hand camera, showing the tracking markers for visual servoing

Here we have applied the analytical method for determination of C, which is represented by the Pseudoinverse of the Jacobian matrix of the m deviations in the sensor space w.r.t. the n deviations in the control space. For that we moved the robot's TCP a little bit around in all $n=6$ degree of freedoms, recorded the corresponding sensor values and generated the Jacobian from the resulting difference quotients.

$$J_{i,j} = \left.\frac{\partial y_i}{\partial x_j}\right|_{x^*} \quad i = 1..m, j = 1..n$$

We performed the experimental determination of C in our simulation environment as well as in the real one. The result was nearly the same, due to the accuracy of our camera simulation. The camera parameters have been estimated using an in-house developed camera calibration tool.

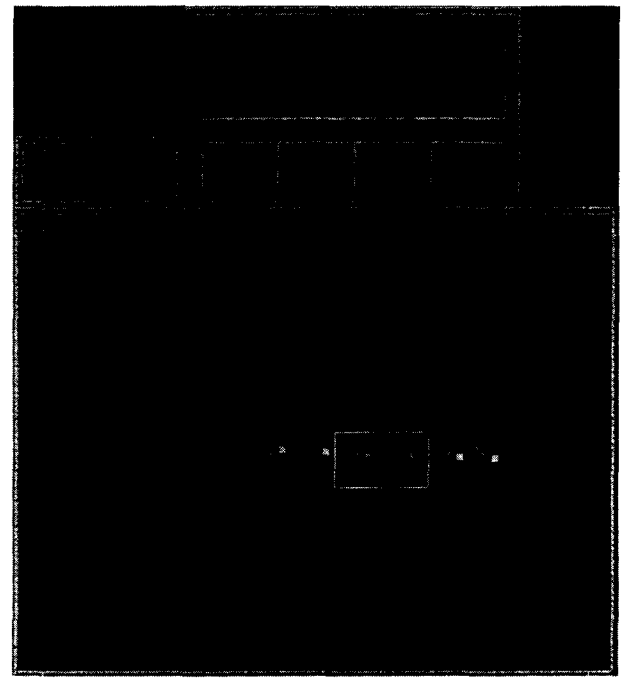

Figure 10 marker selection from real video image

For control we used the TaskBoard marker features, which were originally available to teleoperate the TCP into the right position. The goal was to find the markers in the life video image and to generate the appropriate straight path command to move the robot into the desired (sensor-defined) target pose. To verify the vision-based sensor control loop, we moved the TCP intentionally into a position different from the target pose (a few centimeters in all translational directions and about 20 degrees in z-rotation).

After 3 cycles (with $\alpha = 1$), the target pose was reached. To extract the markers from the video image we used a blob-finding algorithm supported by the MIL (Matrox Image Library) functionality. Because this algorithm delivered more „markers" as desired, e.g. due to bad lighting conditions, we selected the markers interactively and checked the resulting control command before sending it to the real robot. Figure 10 shows the simulated(■) and the real(**X**) markers, with the interactive selection frame.

The differences between the ■ and **X** markers in Figure 10 result from a different TCP pose, to show the two representations. If real and simulated TCP are the in the same pose, the real and simulated markers have nearly the same 2D-coordinates.

A major part of the GETEX experiment time was allocated to the so-called *Dynamic Motion* experiments, which consisted of a series of manoeuvres carried out by the manipulator while the attitude control system of ETS-VII was switched off. In such a mode of operation, a space robot consisting of a manipulator and a satellite is generally considered to be free of external forces[7] [8]. The robot therefore is assumed to have constant angular momentum, due to the law of the conservation of angular momentum, which means that if the arm moves and thus introduces angular momentum into the system, the satellite reacts with a compensating motion. The amount of satellite rotation produced depends on the mass and inertia of the bodies which constitute the system. The description of a TCP trajectory in orbit-fixed coordinates, as it is necessary e.g. for the capturing of a defect satellite, has to account for the satellite reaction. For more details see [5].
The experiments conducted during the GETEX mission aimed at a verification of the existing models of free-floating space robots and at the identification of the dynamic model parameters such as the satellite inertia tensor. A further goal was to obtain some insight into the nature and importance of disturbances acting on a robotic satellite in low Earth orbit and to gather data for the future design of controllers which will combine the manipulator motion control with the satellite attitude control. Therefore, a variety of different manoeuvres were executed (an example of which is shown in Figure 11), which include simple point-to-point operations and closed-loop re-orientation manoeuvres, sequences during which only one joint was active at a time as well as sequences during which all joints were moving simultaneously.

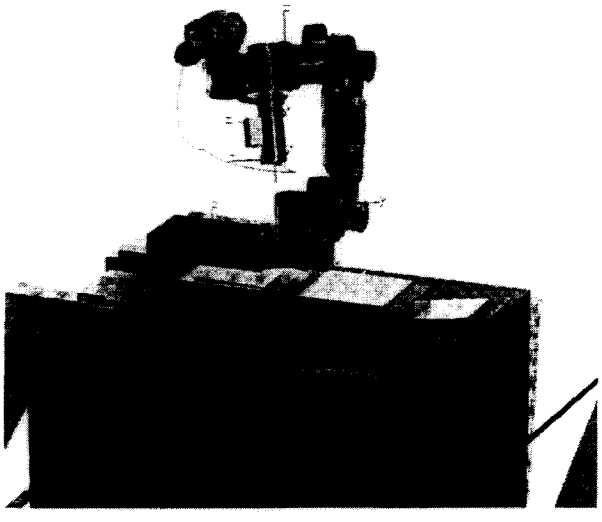

Figure 11 Example of a *Dynamic Motion* manoeuvre carried out during the GETEX mission.
The shaded robot indicates the reference position. The satellite reaction to the arm motion is scaled by a factor of 10 in this picture.

The major constraints, due to mission security aspects, were the maximum satellite attitude error allowed by NASDA which was limited to $\pm 1.0^{\circ}$ around each axis and the fact that the maximum tool center point velocity was limited, too. Furthermore, the reaction wheels were turning at a very low but non-zero constant velocity during the experiments, which introduced undesired torques into the system. Their effects will have to be considered during the evaluation of the mission results.
In total, over 110 minutes of dynamic motion experiments have been carried out, of which 52 minutes have been spent in free motion mode. The remaining time was used to repeat the experiments in reaction wheel attitude control mode for verification purposes. First evaluations of the measurement data confirm the need to account for external disturbance forces acting on the satellite, such as the gravity gradient torque and magnetic toque.

The ESS scenario

For DLR the participation in the Japanese ETS-VII experiment was the first step to a very big challenge in space robotics: the capturing and repair of a failed satellite, completely controlled remotely from earth.
The technology study on the experimental servicing satellite (ESS)[9] applies robotics to solve the problem of servicing a non-cooperative target in or near to a geostationary orbit, a region of space still out of reach to manned spaceflight. A three-month demonstration flight of ESS has been planned and all phases of its mission have been defined. These include the acquisition, inspection and servicing of an orbiting satellite through to parking it in a *graveyard* – orbit.
For that external servicing task high interactivity between man and machine is required, because the remote environment will be mainly unknown. The MARCO system[10] will be used to give the system the local autonomy by intelligent sensor data processing. Because all the satellites, built so far, are not equipped for servicing, the final stages of approach and the subsequent capture of the target are the most critical phases of the mission.

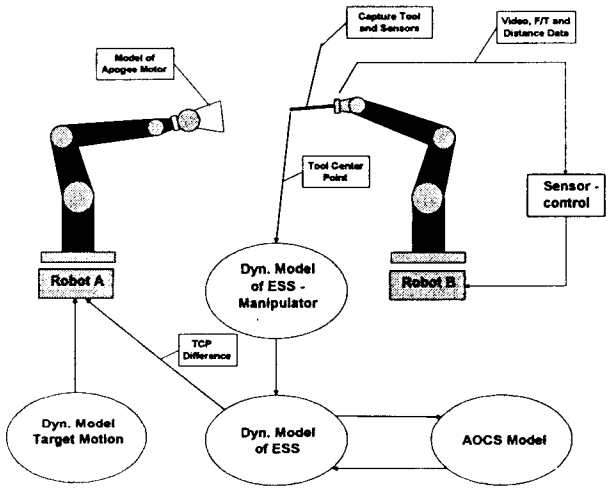

Figure 12 ESS simulation and testbed

The manipulator of ESS, equipped with a capturing tool, must follow the residual movements of a selected object on the target (e.g. the main thruster) by means of an image processing system whose data are passed through an extended Kalman filtering process. With the robot controller monitoring laser distance sensor values, force, torque and travel, the capture tool is inserted into the cone of the thruster.
To simulate the dynamic behavior of the chaser during robot motions, we have arranged two KUKA robots as shown in Figure 12. Robot B is used to carry out the capturing task, Robot A emulates the entire dynamic relation between the chaser and the target satellite, where the dynamic coupling with the AOCS is included.
After capturing the target satellite, the ensemble is stabilised and reoriented. To free the manipulator for servicing activities and to provide a stiff mechanical coupling, the target satellite is grasped by means of a docking mechanism (grasping brackets in Figure 14).

To perform its servicing tasks, the robot replaces the capture tool with an appropriate servicing tool such as a scissor or a gripper. This requires that a tool adaptor, fitted with an integrated force and torque sensor and a stereo camera, is attached to the manipulator's endmost section. The tool exchange process is executed automatically, but control of the repair task itself must be shared between the machine and a human operator at the ground station. To counter the transmission time delay, a predictive graphical simulation of the robot's behaviour in its environment is used at the ground station.

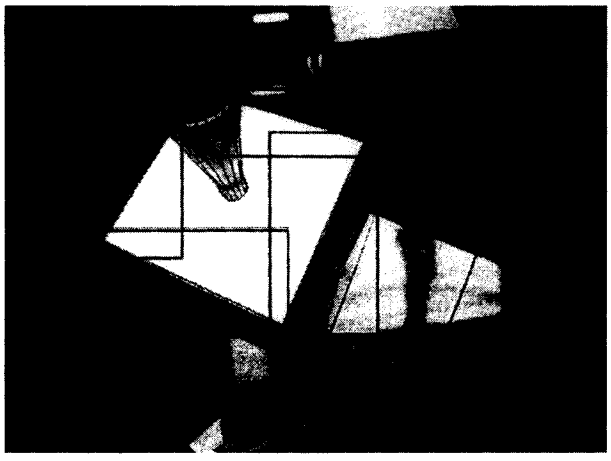

Figure 13 Tracking of target's apogee as seen from the wrist-mounted hand camera.
The wireframe model of the target is projected into the live video image at the currently estimated pose.

Although ESS is a highly complex automatic system, it is easy to maintain and its architecture is simple and extendable. This implies the use of modular hardware and software.

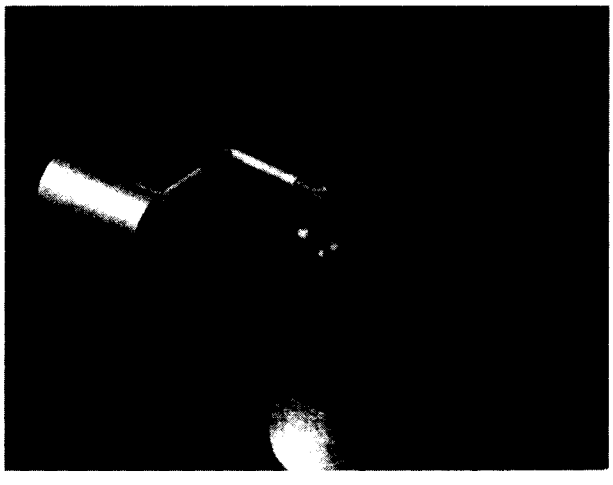

Figure 14 An artist's view of ESS, catching the apogee of TV-Sat-1

To curb costs, standardised elements are used wherever possible to realise the basic satellite functions. The satellite is now sufficiently defined to allow component procurement (in the next stage of the project) to proceed.

Technology Exposure Facility (EuTEF) at ISS

Recently we have performed extensive studies[11] of the European Technology Exposure Facility (EuTEF) to be robot operated at the truss structure of the International Space Station (ISS). Each payload module (PM) consists of the standard body structure (SBS) mounted on the standard grasping interface (SGI). It is placed on the express pallet (ExPA) base by means of the standard receptacle (SR). A payload module may be manipulated by the use of the basic end-effector (BEE) mounted on the robot's flange. Due to the fact that the TEF scenario will be very well-known and predictible, the implicit commanding levels can be used without any problems.

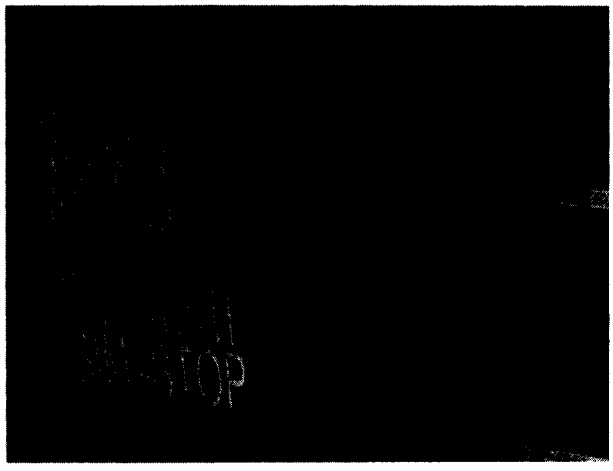

Figure 15 3D-cursor VR-interface with "function objects"and the BEE approaching a SBS

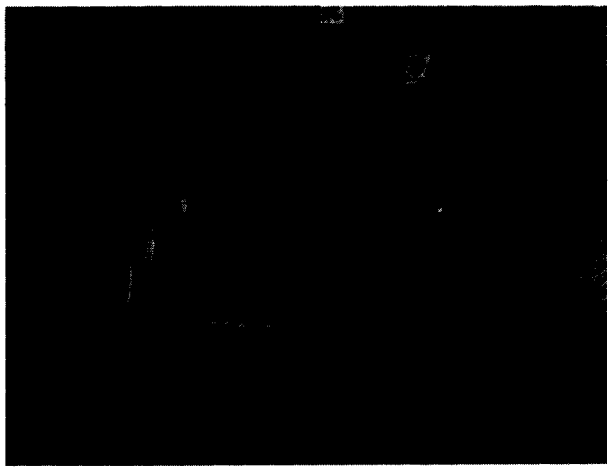

Figure 16 The BEE, approaching the SGI via visual servoing (4 markers around the middle hole)

We have the idea that the payload experts are sitting home in front of their PC's and command and supervise the TEF environment via a simple Java/VRML interface. Only in case of a failure the robot expert will take over the control and teleoperate the TEF robot into a save status or finish the desired task using the explicit MARCO levels.
We propose to apply the GETEX experience in *vision&force* control to the EuTEF scenario, e.g. to support the approach phase to a SGI by visual servoing: to align the BEE with the SGI's grasping position, the robot arm could track 4 colored markers in front of the SGI, which should be easily extracted by an image processing system, and then continue the grasping action with a well-known force/torque-control algorithm.

Multi-fingered Service Robotics

For dangerous and expensive extravehicular tasks as well as for intravehicular payload activities, which are optimized for human operation, we propose a robonaut system, which is able to interact with a human-adapted environment. We have equipped our 7-axes light-weight-robot with a dextrous human-like 4-finger-hand[12] to handle devices, which are standard in a human environment, and with a 3-axes gantry to reach all positions within an experimental spacelab setup (see Figure 17). The control and programming system as used for the above applications is flexible enough for usage in this multi-degrees-of-freedom system. In extension to the former application a data glove is used for teleoperating and programming human-like grasp and manipulation actions.

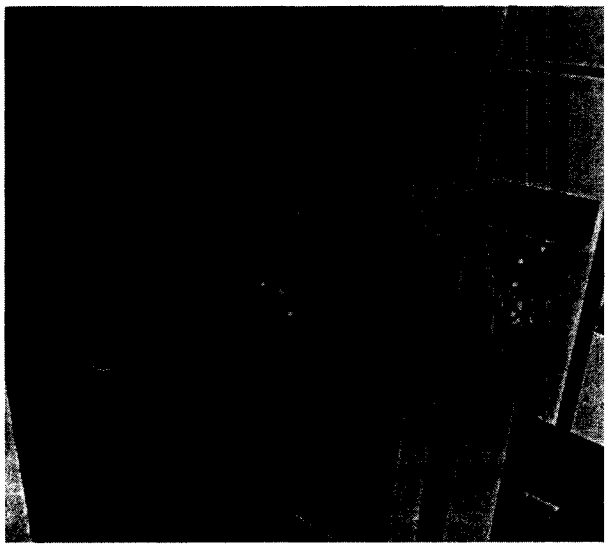

Figure 17 DLR's light-weight robot with 4-finger hand, mounted on a 3-axis-gantry

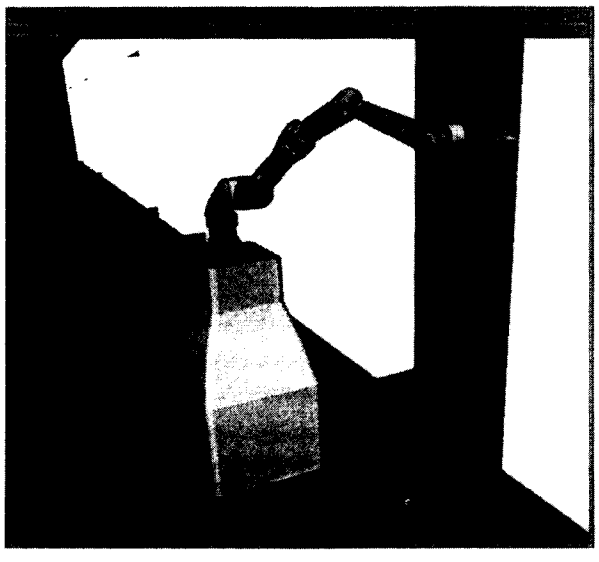

Figure 18 DLR's light-weight robot, mounted on a mobile platform for terrestrial service robot applications
(e.g. opening a door)

In the spirit of the current system, an operator should not have to be an robot expert, this holds extremely for the acceptance of redundant manipulator systems which provide additional freedom. Here the future development will lead to a *task specific exploitation of Redundancy*. In the first step, we showed, that the described system is able to handle this complex kind of kinematics. As future development, we will optimize the task specific exploitation of redundancy, so that the user will be able to easily control even such complex kinematics as 10 Axes of robotic manipulators plus additional 12 from the DLR Hand without being a robotics expert.
It should be mentioned that our programming system is immediately applicable to terrestrical service robotics: instead of the gantry a mobile platform is used to implement a „butler" robot, which will be able to perform helpful tasks in an ordinary environment, e.g. get and bring a bottle of water from the refrigerator to a disabled person (see Figure 18).

Conclusion

We have shown the universal capabilities of DLR's MARCO system for controlling any kind of robotics applications, especially for space. Recently (April '99) we have performed the GETEX mission at ETS-VII with very successful results. Now we believe that the extensive use of robotics at the ISS must be pushed by all industrial and political partners. Furtheron, in our opinion, there are only few space technologies which promise such high terrestrial spin-off and technology transfer potentials as the development of sensor-based task-level programming tools as well as intelligent (i.e. sensor-controlled) artificial robot arms. A recent example is the very encouraging feedback of the automotive industry during the Hannover fair '99, as we have applied the *„vision&force"* control paradigma to insert pistons into a rotating motorblock fully automatically.

References

[1] M. Oda, K. Kibe, F. Yamagata, „ETS-VII – Space Robot In-Orbit Experiment Satellite", IEEE Conf. on Robotics and Automation, Minneapolis, April 1996

[2] G. Hirzinger, B. Brunner, J. Dietrich, J. Heindl, „ROTEX – the First Remotely Controlled Robot in Space,", IEEE Conf. on Robotics and Automation, San Diego, May 1994

[3] B. Brunner, K. Landzettel, B.M. Steinmetz, G. Hirzinger, „TeleSensorProgramming – a task-directed programming approach for sensor-based space robots", Int. Conf. On Advanced Robotics (ICAR'95), Sant Feliu de Guixols, Spain, Sept. 1995

[4] http://www.robotic.dlr.de/Joerg.Vogel/Vrml/Rotex/

[5] K. Landzettel, B. Brunner, G. Hirzinger et. al., „DLR's robotics lab – recent developments in space robotics", i-SAIRAS 5th International Symposium on Artificial Intelligence, Robotics and Automation in Space, 1-3 June 1999, ESTEC, Noordwijk, The Netherlands

[6] B. Baginski, „Efficient Dynamic Collision Detection using Expanded Geometry Models", IEEE/RSJ/GI Int. Conf. on Intelligent Robots and Systems IROS'97, Grenoble, September 7-11, 1997

[7] Longman, R.W., Lindberg, R.E. and Zedd, M.F., "Satellite-mounted Robot Manipulators – New kinematic and Reaction Moment Compensation", International Journal of Robotics Research, 3, 1987.

[8] Dubowsky, S. and Papadopoulos, E., "The Kinematics, Dynamics and Control of Free-Flying and Free-Floating Space Robotic Systems", IEEE Transactions on Robotics and Automation, 5, 1993.

[9] http://esapub.esrin.esa.it/pff/pffv7n2/settv7n2.htm

[10] K. Landzettel, B. Brunner, G. Hirzinger, "The Telerobotic Concepts for ESS", IARP Workshop on Space Robotics, Montreal, July 1994

[11] K. Landzettel, B. Brunner, G. Hirzinger, I. Schäfer, M. Fischer, M. Grebenstein, N. Sporer, J. Schott, „ Space Robotics – Recent Advances in DLR's Robotics Lab", 5th Workshop on Advanced Space Technologies for Robotics and Automation (ASTRA'98), Noordwijk, The Netherlands, Dec 1998

[12] M. Fischer, P. van der Smagt, G. Hirzinger, „Learning techniques in a dataglove based telemanipulation system for the DLR hand", IEEE Conf. on Robotics and Automation, Leuven, Belgium, April 1998