EVA Crew Assistant Supporting Astronauts in Space Missions: Autonomy and Cooperation

E. Zereik*, A. Sorbara**, A. Merlo**, G. Casalino* and F. Didot***

*Department of Communication Computer and System Sciences, University of Genoa, Italy
e-mail: enrica.zereik@dist.unige.it, casalino@dist.unige.it

**Thales Alenia Space Italy, Turin, Italy
e-mail: andrea.sorbara@external.thalesaleniaspace.com, andrea.merlo@thalesaleniaspace.com

***ESA/ESTEC, Noordwijk, The Netherlands
e-mail: frederic.didot@esa.int

Abstract

Robotic crew assistants employed in space missions can greatly improve astronaut working quality and assure mission success. This work presents some results obtained in the development of an effective dual-arm mobile manipulator for surface operations. This research is focused on two different aspects: the execution of operation in a completely autonomous manner (without human supervision), as well as cooperation with humans. Vision and force based control strategies were exploited together with a suitable control algorithm based on Dynamic Programming, proposed for the robot coordination.

1 Introduction

In recent years, robotic research has gradually tried to increase its presence in everyday life, in such a way to relieve people of boring tasks, thus improving their life quality. In space environment, robot contribution has even more importance, as activities required in space can be very difficult or dangerous to mission crew mates and, moreover, astronauts are hindered by their suits and sometimes have not the necessary dexterity to accomplish delicate tasks. Robotics can thus help correctly execute repetitive routines in hostile environments and improve space mission safety level. In this scenario a high level of autonomy is needed by the robotic system as it must be able to work essentially by itself, without needing human supervision; moreover, also reliability and safety are fundamental features the robot must have in order to cooperate with humans.

For these reasons, cooperation and autonomy are fundamental issues for space robots: both cooperation in tasks that turn out to be difficult for astronauts by themselves, such as the lifting and transportation of heavy and large equipment, and autonomous activities, such as the grasping and manipulation of objects needing to be replaced, can highly limit man intervention and improve the chances of success in missions.

In this work a robotic EVA crew assistant demonstrator is presented; this system, named *Eurobot Ground Prototype* (EGP), has the basic objective to demonstrate the usefulness of a robotic crew member providing effective assistance to its astronaut mates in different ways. EGP, in fact, is able to autonomously grasp objects (such as ORU, spare parts, MLI panels, antennas and several instruments) from their stowage and relocate them where required. Moreover, the robot can help humans in the transportation of objects that are too big or heavy to be carried by astronauts themselves. This project, conducted in conjunction with the European Space Agency and Thales Alenia Space, meant to demonstrate one of the potential applications in space of such a robotic crew assistant, while underlying its effectiveness within a scenario chosen to be as representative as possible of real space mission environments. The state of the art in the robotic crew assistant field is established by robots that can perform planetary surface operations but that must be supervised and controlled by a human operator through a telepresence system. One of the most advanced of these robots is NASA's Robonaut, a humanoid torso mounted on a wheeled rover, which can work as an EVA astronaut equivalent[1]. EGP originality, in this sense, lies in its capability to autonomously perform tasks such as manipulate ORU, MLI panels, launch fittings without requiring any human intervention. Challenging issues, during EGP control software development, have been addressed: the robot is able to assist the astronaut during operations like equipment moving, following his movements thanks to its force/torque sensors and understanding his voice commands. Moreover, vision and force control allowed EGP to accomplish autonomous manipulation tasks, thus correctly interacting with its working environment, and to execute operations involving equipment handling (ORU, MLI, launch fittings…) and failed item replacement. Finally, for such a robotic crew assistant, coordination among each subsystem is a key issue, in order to assure a good behavior and to increase the robot workspace size.

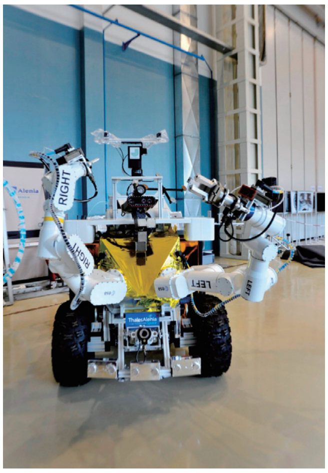
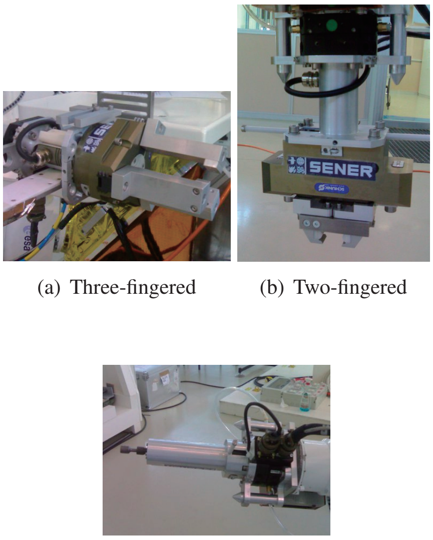


Figure 1. EGP is a robotic crew assistant made up of a rover and two arms, endowed with cameras and force/torque sensors



(a) Three-fingered

(b) Two-fingered

(c) Electrical screwdriver

Figure 2. EGP exchangeable end-effectors

2 EGP Description and Objectives

EGP project originates from a previous ESA project called Eurobot Wet Model (EWM) ([2] and [3]) aimed to develop a three-arm robot servicing the International Space Station. Two of the three EWM arms were then mounted on the top-front part of a four-wheeled rover, in such a way to obtain a mobile crew assistant, shown in Figure 1, able to navigate on a planetary surface. Each EGP arm has 7 degrees of freedom and both a firewire color camera (needed by autonomous vision-guided operations) and a JR3 force/torque sensor (to be able to react to external stimuli) are placed onto its wrist.

EGP arm position and orientation on board the rover (onto its top-front part) were accurately selected according to different requirements: first of all, in order to allow the robot to reach an object on the ground and grasp it and then to make the handover of items to and from a human astronaut feasible; moreover, the arm location onto the rover permits allow EGP to lift any tool and store it onto its payload tray (on its top-side part), both autonomously and while cooperating with astronauts. A stereo camera assembly for rover navigation and a three-eyed robotic head for manipulation tasks constitute EGP vision system, together with the two cameras mounted on the two arm wrists. The robotic head is made up of three firewire cameras mounted on a pan/tilt unit accommodated on top of the rover, between the two arms. Each camera has a maximum resolution of 1024×768 pixels and can acquire images at a frame rate of 30 fps all by itself or, alternatively, at 7.5 fps if three cameras are performing at the same time. This head duty is that of providing the human operator (both on a ground station and on the planet) with the possibility to supervise each manipulation operation executed by Eurobot.

EGP two main aspects are linked to cooperation with humans and autonomous behaviors: the problem of physically cooperate with astronauts (performing tasks such as lifting equipment together) has been addressed and a coordination algorithm for this kind of task has been proposed and will be explained in the following. Furthermore, Eurobot can autonomously manipulate objects, thus performing difficult tasks such as failed equipment replacement all by itself: thanks to its self-exchangeable grippers, EGP can grasp both non-robotics and robotics interfaces. For example, a two-fingered gripper was selected to be compatible with EVA interfaces that are mounted on the Lander and on the objects to be handled by the robot. Besides the two-fingered one, a three-fingered end-effector MLI handling and sample collection and an electrical screwdriver for bolt screw operations were included into EGP tools. All these grippers are shown in Figure 2

3 Cooperation

Focusing on the cooperative task of equipment lifting together with astronauts, a general strategy for control and coordination of complex robotic structures is detailed. This method, based on Dynamic Programming techniques, is derived from work illustrated in [5], [6] and [7] and developed at GRAAL Lab, in Genoa; this work has been generalized and can be applied to all complex robots, regardless of their specific structure. Further control approaches to coordination can be found in [8] and [9]. Our method can thus be employed in different situations: for example, a simple fixed-base arm or one or even more arms mounted on a moving platform. Moreover, different control objectives can be required to the robot: besides the classical goal reaching task, we can introduce safety constraints; for example we can request the robot to keep a safe posture during its motion (e.g. far from joint limits and singularities) and to avoid external obstacles. Each requirement can then be assigned to a different priority level, according to the importance linked to each task: the goal reaching constraints has, commonly, the utmost importance, while the other secondary tasks (i.e. safety maintenance) usually are linked to decreasing priorities. However in this case, whenever the robot approaches a bad situation (e.g. the risk of hitting an obstacle or reaching a posture close to a singularity or a joint limit), a mechanism for priority inversion is needed, in order to discard the primary task (reaching the goal position) and recover from a potentially dangerous situation; this priority change strategy will be detailed in the following.

3.1 Defining Control Goals

Each control objective specifies a desired velocity to be accomplished by the robot and then the control actions for all the subsystems of the overall robotic structure are computed, according to the priorities of the different required velocity contributions. Note that these desired velocities can be obtained in different ways, according to the robot capability to interact with the environment: for example, both EGP vision system and its force/torque sensors can generate reference velocity command to be executed by the robot to accomplish its operations.

In the case of a moving base manipulator, Equation 1 shows the general structure of a task to be required to the robot; the condition $q \in A_{i-1}$ forces the final velocity reference to fulfill all previous tasks, having a higher priority.

$$A_i \triangleq \left\{ \dot{q} = \underset{\dot{q} \in A_{i-1}}{\operatorname{argmin}} \left\| (\dot{x}_i - S_i \dot{y}) - J_i \dot{q} \right\|^2 \right\} \tag{1}$$

In Equation 1 $\dot{x}_i$ represents the required velocity reference, while the moving base velocity is taken into account by the term $\dot{y}$ that induces, in turn, a motion to the end-effector; this so induced velocity can be computed through the rigid body transformation S_i. After all desired requirements have been written down in the form stated by Equation 1, starting from the first (and with the highest priority) one, we can express the needed control action, satisfying this primary task, by solving Equation 2

$$\left\| \dot{\varepsilon}_1^o \right\|^2 = \min_{\dot{q}} \left\| (\dot{x}_1 - S_1 \dot{y}) - J_1 \dot{q} \right\|^2 \tag{2}$$

and finding its least square solution as in Equation 3.

$$\dot{q} = J_1^{\#} (\dot{x}_1 - S_1 \dot{y}) + \left(I - J_1^{\#} J_1 \right) \dot{z}_1 \tag{3}$$

In order to make the control problem in Equation 2 assume a more compact form, we can define the following quantities

$$\begin{aligned} \dot{\bar{X}}_1 &= \dot{x}_1 - J_1 \dot{h}_0 = \dot{x}_1 \\ \bar{J}_1 &= J_1 Q_0 = J_1 \\ \bar{S}_1 &= S_1 - J_1 P_0 \end{aligned} \tag{4}$$

thus leading to rewrite expression in Equation 2 as

$$\left\| \dot{\varepsilon}_1^o \right\|^2 = \min_{\dot{q}} \left\| \left(\dot{\bar{X}}_1 - \bar{S}_1 \dot{y} \right) - \bar{J}_1 \dot{q} \right\|^2 \tag{5}$$

where the following expressions hold.

$$\begin{aligned} \dot{h}_0 &= 0 \\ Q_0 &= I \\ P_0 &= 0 \end{aligned} \tag{6}$$

According to these definitions, the desired joint velocity that fulfills the current task can be expressed as in Equation 7

$$\begin{aligned} \dot{q} &= \bar{J}_1^{\#} \dot{\bar{X}}_1 - \bar{J}_1^{\#} \bar{S}_i \dot{y} + \left(I - \bar{J}_1^{\#} \bar{J}_1 \right) \dot{z}_1 = \\ &= \dot{\zeta}_1 - P_1 \dot{y} + H_1 \dot{z}_1 = \\ &= \left(Q_0 \dot{\zeta}_1 + \dot{h}_0 \right) - P_1 \dot{y} + Q_0 H_1 \dot{z}_1 = \\ &= \dot{h}_1 - P_1 \dot{y} + Q_1 \dot{z}_1 = \bar{J}_1^{\#} \left(\dot{\bar{X}}_1 - \bar{S}_1 \dot{y} \right) + Q_1 \dot{z}_1 \end{aligned} \tag{7}$$

after establishing the following assumptions:

$$\begin{aligned} \dot{\zeta}_1 &= \bar{J}_1^{\#} \dot{\bar{X}} \\ \dot{h}_1 &= \dot{h}_0 + Q_0 \dot{\zeta}_1 \\ P_1 &= P_0 + Q_0 \bar{J}_1^{\#} \bar{S}_1 \\ H_1 &= \left(I - \bar{J}_1^{\#} \bar{J}_1 \right) \\ Q_1 &= Q_0 H_1 \end{aligned} \tag{8}$$

Note that the cost-to-go associated to the control problem can be expressed as:

$$\begin{aligned} \left\| \dot{\varepsilon}_1^o \right\|^2 &= \left\| \dot{\bar{X}}_1 - \bar{S}_1 \dot{y} - \bar{J}_1 \bar{J}_1^{\#} \left(\dot{\bar{X}}_1 - \bar{S}_1 \dot{y} \right) \right\|^2 = \\ &= \left\| \left(I - \bar{J}_1 \bar{J}_1^{\#} \right) \left(\dot{\bar{X}}_1 - \bar{S}_1 \dot{y} \right) \right\|^2 = \\ &= \left\| \bar{V}_1 \left(\dot{\bar{X}}_1 - \bar{S}_1 \dot{y} \right) \right\|^2 ; \ \bar{V}_1 \triangleq \left(I - \bar{J}_1^{\#} \bar{J}_1 \right) \end{aligned} \tag{9}$$

The same passages hold for all other tasks: for the second one, in fact, we can state the control problem in the same form as Equation 2

$$\left\| \dot{\varepsilon}_2^o \right\|^2 = \min_{\dot{q} \in A_1} \left\| (\dot{x}_2 - S_2 \dot{y}) - J_2 \dot{q} \right\|^2 \tag{10}$$

with the additional condition $\dot{q} \in A_1$; this last implies the following statement

$$\dot{q} \in A_1 \quad \Leftrightarrow \quad \dot{q} = \dot{h}_1 - P_1 \dot{y} + Q_1 \dot{z}_1 \tag{11}$$

thanks to which we can rewrite the control problem as:

$$\begin{aligned} \left\| \dot{\varepsilon}_2^o \right\|^2 &= \min_{\dot{z}_1} \left\| \dot{x}_2 - S_2 \dot{y} - J_2 \dot{h}_1 + J_2 P_1 \dot{y} - J_2 Q_1 \dot{z}_1 \right\|^2 = \\ &= \min_{\dot{z}_1} \left\| \left(\dot{\bar{X}}_2 - \bar{S}_2 \dot{y} \right) - \bar{J}_2 \dot{z}_1 \right\|^2 \end{aligned} \tag{12}$$

The term $\dot{z}_1$ is an arbitrary vector added in order to better exploit robot degrees of freedom that are still available (in such a way to satisfy, as much as possible, secondary requirements); note that this vector does not affect the primary task: it is in fact multiplied by the term $\left(I - \bar{J}_1^{\#} \bar{J}_1 \right)$, which is an orthogonal projector onto the null space (or kernel) of $\bar{J}_1$. This velocity contribution $\dot{z}_1$ will thus be in charge of maintaining a good robot manipulability, while the goal frame reaching is still pursued.

Solving Equation 12 for $\dot{z}_1$ we can retrieve the expression

$$\begin{aligned} \dot{z}_1 &= \bar{J}_2^{\#} \left(\dot{\bar{X}}_2 - \bar{S}_2 \dot{y} \right) + \left(I - \bar{J}_2^{\#} \bar{J}_2 \right) \dot{z}_2 = \\ &= \bar{J}_2^{\#} \dot{\bar{X}}_2 - \bar{J}_2^{\#} \bar{S}_2 \dot{y} + H_2 \dot{z}_2 \end{aligned} \tag{13}$$

and the corresponding control action can be written as

$$\begin{aligned}\dot{q} &= \dot{h}_1 - P_1\dot{y} + Q_1\dot{\zeta}_2 - Q_1\bar{J}_2^{\#}\bar{S}_2\dot{y} + Q_1H_2\dot{z}_2 = \\ &= \left(\dot{h}_1 + Q_1\dot{\zeta}_2\right) - \left(P_1 + Q_1\bar{J}_2^{\#}\bar{S}_2\right)\dot{y} + Q_2\dot{z}_2 = \\ &= \dot{h}_2 - P_2\dot{y} + Q_2\dot{z}_2\end{aligned} \tag{14}$$

while the expression for the cost-to-go is similar to Equation 9. As the same assumptions and passages hold for the $i-th$ task, the algorithm can be generalized as in Equation 15

$$\begin{aligned}\dot{q} &= \dot{h}_i - P_i\dot{y} + Q_i\dot{z}_i \\ \left\|\dot{\varepsilon}_i^o\right\|^2 &= \left\|\left(I - \bar{J}_i\bar{J}_i^{\#}\right)\left(\dot{\bar{X}}_i - \bar{S}_i\dot{y}\right)\right\|^2 = \\ &= \left\|\bar{V}_i\left(\dot{\bar{X}}_i - \bar{S}_i\dot{y}\right)\right\|^2; \quad \bar{V}_i = \left(I - \bar{J}_i\bar{J}_i^{\#}\right)\end{aligned} \tag{15}$$

provided that the usual following definitions have been established:

$$\begin{aligned}\dot{\bar{X}}_i &= \dot{\bar{x}}_i - J_i\dot{h}_{i-1} \\ \bar{J}_i &= J_iQ_{i-1} \\ \dot{\zeta}_i &= \bar{J}_i^{\#}\dot{\bar{X}}_i \\ H_i &= \left(I - \bar{J}_i^{\#}\bar{J}_i\right) \\ \bar{S}_i &= S_i - J_iP_{i-1} \\ P_i &= P_{i-1} + Q_{i-1}\bar{J}_i^{\#}\bar{S}_i \\ \dot{h}_i &= \dot{h}_{i-1} + Q_{i-1}\dot{\zeta}_i \\ Q_i &= Q_{i-1}H_i\end{aligned} \tag{16}$$

Note that the vector $\dot{y}$ was assumed to be as general as possible: it can be considered as the velocity of a generic robotic moving base (with other robots - like arms - on top) and so this algorithm can be employed regardless of the particular robot structure.

Before applying this algorithm to the specific EGP case, we have to make some assumptions; first of all, we consider a particular cooperative task such as the lifting of equipment by Eurobot and an astronaut together. This task is guided by the human himself, as he is able to command EGP motion through his own movements: the object to be transported is assumed to be already grasped by both astronaut and robot (this grasping phase is guided through EGP vision and force systems, as will be clear in a while) and EGP is guided by the force applied to the object by the astronaut himself. Note that, for such a task, only one of Eurobot arms is employed, as equipment other side is lifted by the human crew member. Figure 3 summarizes all relations among velocities and control variables of arms and rover.

Different control objectives can be defined; within the specific context the following requirements are considered: fulfill the desired reference velocity, proportional to the force applied to the object by the human and sensed by EGP force/torque sensors; keep equipment maximally aligned to the terrain slope or, similarly, maintain vectors k_e and k_p aligned (Figure 3); assure a good arm manipulability during task execution. From these requirements it is possible to determine the following control tasks, with a priority level decreasing from the first to the last one

$$1)\ \bar{v} \approx v = \left(J_{L1}\dot{q}_1 + J_{L2}\dot{q}_2\right) \ \propto \ {}^sF \tag{17a}$$

$$2)\ \bar{\omega} \approx G\omega \triangleq \left(\bar{J}_{A1}\dot{q}_1 + \bar{J}_{A2}\dot{q}_2\right) \tag{17b}$$

$$3)\ \dot{\bar{\mu}} \approx p\dot{q}_1 \ \leftarrow \ \dot{\bar{\mu}} \ \triangleq \ \lambda\mu \ \leftarrow \ \mu \ \triangleq \ \det\left(J_{L1}^TJ_{L1}\right) \tag{17c}$$

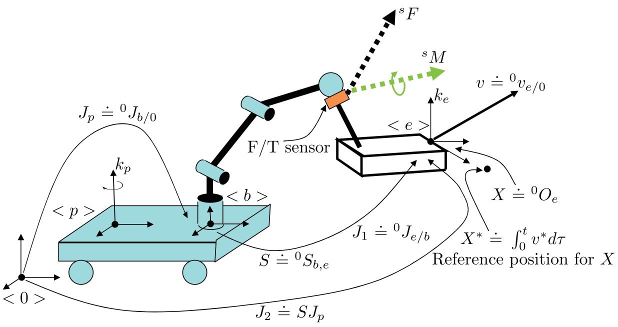


Figure 3. Frame positioning and main relations among rover and arm

being $\dot{q}_1$, $\dot{q}_2$ the joint velocities of arm and vehicle, J_{L1}, J_{L2} and J_{A1}, J_{A2}, respectively, the linear and angular parts of their Jacobian matrices, v and ω the (linear and angular) velocities induced to the arm end-effector, $\bar{v}$ and $\bar{\omega}$ the desired final (linear and angular) velocities for the arm end-effector (computed taking into account both arm and rover motion). Equations 17a and 17b deal with the requirement of accomplish a (linear) velocity proportional to the force applied by astronaut and to keep the object aligned to the traversed terrain. Note that matrix G has a "reductive" effect on the Jacobian matrices: it allows to control only vectors k_e and k_p to be aligned and not all unit vectors of frames $< e >$ and $< p >$; this reduction is important because allows the system to have more degrees of freedom still available for secondary tasks. Finally, Equation 17c tries to keep the arm *Manipulability Measure* above an assigned threshold μ_0 or, similarly, tries to force the arm to assume only good postures during its motion. For further details on this Manipulability Measure refer to [4]. After defining all required control tasks, both Backward and Forward Phases of our algorithm based on Dynamic Programming can be executed.

3.2 Dynamic Programming

Our Dynamic Programming based technique starts from the already established requirements for the robot and goes on with a Backward and a Forward Phase. In the *Backward Phase* control actions are retrieved starting from the top robotic structure (in this case the arm) until arriving to the bottom one (the rover). From Equation 17, recalling that $v_2 = J_{L2}\dot{q}_2$ and $\omega_2 = \bar{J}_{A2}\dot{q}_2$, we can solve for $\dot{q}_1$ in the first task, just getting an expression for $\dot{q}_1$ as a function of the desired linear velocity $\bar{v}$ and parametrized by the (still indeterminate) rover velocity v_2 and the arbitrary vector $\dot{z}_1$. Exploiting all other tasks to compute vectors $\dot{z}_i$, it is possible to retrieve the joint velocity expression as $\dot{q}_1 = L_1\left(\bar{v}, \bar{\omega}, \dot{\bar{\mu}}; v_2, \omega_2\right)$; hence the following Equation 18 holds:

$$\begin{aligned}&1)\ \bar{v} \approx \left[J_{L1}L_1\left(\bar{v}, \bar{\omega}, \dot{\bar{\mu}}; v_2, \omega_2\right) + v_2\right] \\ &2)\ \bar{\omega} \approx \left[\bar{J}_{A1}L_1\left(\bar{v}, \bar{\omega}, \dot{\bar{\mu}}; v_2, \omega_2\right) + \omega_2\right] \\ &3)\ \dot{\bar{\mu}} \approx pL_1\left(\bar{v}, \bar{\omega}, \dot{\bar{\mu}}; v_2, \omega_2\right)\end{aligned} \tag{18}$$

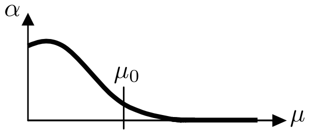


Figure 4. Trend of Alpha gain

From Equation 18 and exploiting again relations $v_2 = J_{L2}\dot{q}_2$ and $\omega_2 = \bar{J}_{A2}\dot{q}_2$, we can obtain that $\dot{q}_2 = L_2(\bar{v}, \bar{\omega}, \dot{\bar{\mu}})$; the Backward Phase thus results in the computation of $\dot{q}_1$ and $\dot{q}_2$ as functions of the reference commands.

Afterwards, the *Forward Phase* has to evaluate these retrieved expressions in order to achieved desired control actions to guide the robot to assume required position and orientation, while avoiding bad postures. This phase starts from the bottom (rover) and ends with the top structure (arm), in order to compute $\dot{q}_2$ first and $\dot{q}_1$ too. As already suggested, the control actions achieved by this procedure maximally satisfy all initial requirements; however, the risk of manipulability losses still exists. In the case of a mobile manipulator, this happens, as an example, when the required end-effector motion includes directions the platform cannot compensate for, e.g. translation along the rover vertical axis. For these reasons, during EGP motion, its manipulability must be constantly checked and a bad situations must be prevented.

This can be done through the introduction of a "priority change" mechanism, in such a way to avoid the accomplishment of a desired velocity that would cause the robot to assume a bad posture. To this aim, the linear velocity reference $\bar{v}$ is modified with the addition of a further term $\hat{\bar{v}}$, as in Equation 19:

$$\tilde{v} = (1-\alpha)\,\bar{v} + \alpha\hat{\bar{v}} \tag{19}$$

This α gain, whose trend is depicted in Figure 4, implements the priority change mechanism: when the robot has a good manipulability $\alpha = 0$ and so $\tilde{v} \equiv \bar{v}$ (that is the velocity contribution is aimed to only accomplish the goal reaching task), while if the robot is close to a bad posture $\alpha = 1$ and $\tilde{v} \equiv \hat{\bar{v}}$ (the control system cares only about the correction velocity term to increase the Manipulability Measure - constantly monitored - and bring it back above the threshold μ_0). This "last-resort" solution of changing the reference command is obviously employed only after all subsystems have computed their own contribution to velocity.

The presented algorithmic approach is effective in keeping the robot far from bad situations (singularities, joint limits...) and, moreover, can be generally applied to any kind of robot: the Dynamic Programming strategy and the priority change mechanism are independent of the specific system to control and both can be adapted to suit it; for further details refer to [4].

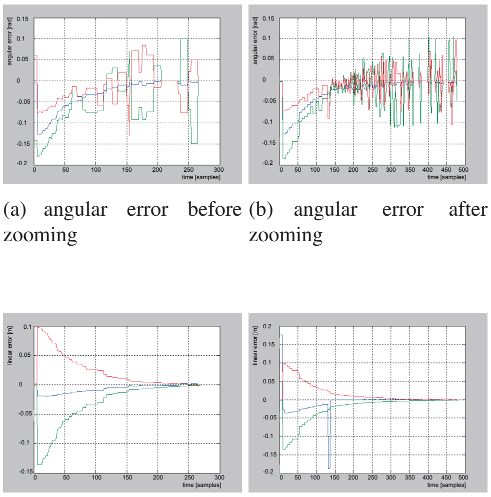
4 Autonomy

The other EGP fundamental aspect is its autonomous behavior: it is in fact able to grasp items on its own (exploiting its cameras and force/torque sensors) and handle it in order to accomplish complex operations in a completely autonomous manner. In fact, for EGP demonstration, a Lander simulator was developed and on it all objects of interest were stowed; Eurobot has thus an a-priori knowledge of each item nominal position on the Lander. Nevertheless, this information is not enough for the robot to correctly grasp item: many errors can occur due to the low DEM resolution, the non-perfect rover actuation and because of EGP localization procedure, executed through an IMU and both mechanical and visual odometry. For these reasons, a strategy based on both vision and force control has been employed, thus allowing EGP to identify and manipulate specific objects in the 3D space. Furthermore, this approach to perception can be exploited to generate velocities to be required to the explained Dynamic Programming strategy for robot control.

4.1 Vision-based Object Recognition

The recognition of general 3D objects is not so easy and robust: many problems arise, such as the need of a precise geometrical model for each object, the possible difficulty to choose the right item if several similar objects are present in the scene, the deformations, occurring because of different points of view, which can make the object seem different (thus leading to a failure in the recognition task) and so on [10], [11]. In order to avoid such problems, allowing EGP to be able to interact with the environment through vision, a method based on fiducial markers was used: each object of interest is associated to a different marker, univocally recognizable by the system, and its position in 3D space (6 degrees of freedom - 3 for linear position and 3 for orientation) is computed in such a way to control the robot to grasp the marked object (obviously enough, the association between markers and objects to be manipulated is known). Each marker is a square planar pattern with known size and structure and is assigned to a unique identification number: this allows the robot to recognize and locate more than one object in the scene. For this purpose an open-source software library, named *ARToolKitPlus*, was selected [12], [13]; it provides some functionalities for marker pose estimation and is robust enough, in terms of both precision and velocity with which the position is computed, in order to be used in a real-time control loop. Moreover, another open-source library, named *OpenCV*, was exploited for all image processing functions [14].

Basically, the marker tracking procedure is simply the following: the camera is acquired by the system that searches for one or more markers in each frame; if at least one marker is recognized, the transformation matrix giving the marker position with respect to the camera is computed and can be used by robot control loop as a feedback, in such a way to make EGP assume a predetermined position and orientation with respect to the marker itself. This general procedure was evaluated and turned out to be not enough robust, considering the high level of reliability required to EGP autonomy skill; more image processing procedure were thus tried and included in the final vision



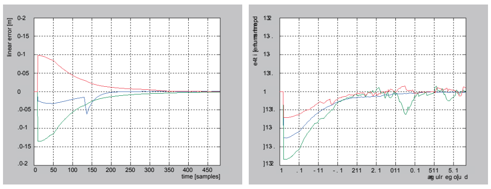
(a) angular error before zooming (b) angular error after zooming

(c) linear error before zooming (d) linear error after zooming

Figure 5. Estimation error dynamics

strategy. Lighting conditions, complexity of the captured scene and distance from which the marker is seen are the main adverse factors affecting the tracking procedure results. Figure 5 shows the difference between the zoomed and non-zoomed case.

Some steps were added to the vision chain in order to overcome these limitations: first of all a preliminary image thresholding is executed; before all other computations, the estimator receives the input image, converts it to black and white and applies to it a threshold value for luminance, in order to increase contrast. If the marker is found, the same threshold is applied to all other frames and the tracking algorithm goes on, otherwise other threshold values, among some previously defined ones, are tried until the marker is recognized. Then the image is cleaned around the areas where a marker was probably identified: an output image is built up, equal to the original one in the "potential marked" zones and white elsewhere. Note that, at this point, the marker position is not yet computed, as this is the computationally heaviest part of the overall algorithm. Finally, when the robot has almost reached the final desired position, the cleaned image is zoomed in its center: with a "bigger" marker in the processed image, the estimator computation time decreases. This faster dynamics achievement is important because errors in the estimated position occur at higher frequency and can then be removed through a simple low-pass filter, as highlighted by Figure 6. Peak in Figure 5(d) is due to a misalignment between the already modified reference and the not yet updated estimation. Obviously, while zooming the image, the marker appears to be closer than it actually is and the estimated linear components are smaller; so also the reference linear position must be accordingly rectified. Figure 7 depicts the overall adopted image processing algorithm.



(a) linear error (b) angular error

Figure 6. Filtered estimation error

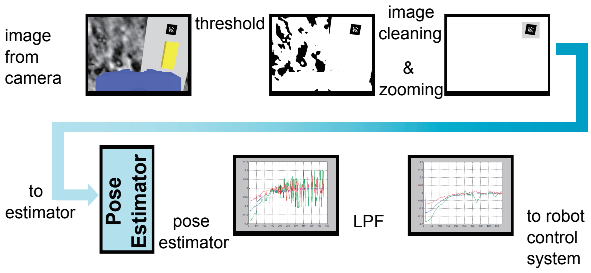


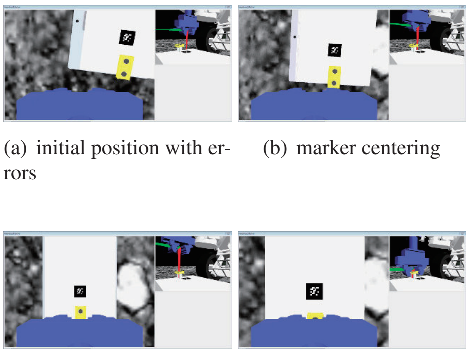
Figure 7. Tracking procedure steps

4.2 Force-based Object Approach

After the object has been recognized by the vision system and the robot has been controlled to stop in front of it, the actual approach and grasp begins. This phase is guided by EGP force/torque sensors: a direct force control [15], [16] is employed to zero potential residual errors (that are experimentally very small) caused by vision. Once the marker has been centered, during the arm approach toward the object, if a contact is detected the force control starts its action, generating a desired velocity. We made the hypothesis of only a pure force applied to the contact point; as this contact point does not coincide with the sensor location, the JR3 sensor, mounted on the arm wrist, will sense both a force f_s and a torque τ_s . These components are compared to desired force and torque references and from the so computed error the velocity to be assigned to the end-effector is obtained. In order to get this velocity to correctly drive the arm end-effector, the estimation of the contact point comes to be an essential step; this point, represented by r, can be computed exploiting the relation among torque, force and lever arm vector and with the aid of the known distance d between the sensor and the contact point itself, as in Equation 20:

$$\begin{aligned} \tau_s &= \left[-f_s \wedge\right] r \\ r &= \left[r_x r_y d\right] \end{aligned} \tag{20}$$

where r_x and r_y are the two unknown components of the contact point and the z axis of the reference frame is along the arm, positive outgoing from the end-effector. So, the contact point can be estimated solving Equation 20 and, once computed, is used as the new tool center point in order to correctly control the robot with a velocity $\dot{x}_c = K\left(F_s - F_s\right)$, simply proportional to the force error (F_s represents the generalized force and torque vector).



(a) initial position with errors

(b) marker centering

(c) position corrected

(d) grasped handle

Figure 8. Two view simulator

All these algorithms (both for what concerns force and vision based approaches and for the coordination and control technique) combine to bring about the final correct EGP behavior in both autonomous and cooperative tasks. The following sections describe all the results achieved from both simulative and experimental tests conducted on EGP.

5 Simulative and Experimental Results

EGP tasks were widely simulated and experimental tests in an apposite scenario were conducted; simulations about the entire vision-based tracking procedure were performed after the realization of a simulator with two different views (one showing the 3D visualization of the overall robot, the other displaying what is seen by the end-effector mounted camera of the working arm). The overall tracking algorithm was thus evaluated and the grasping of an ORU placed on the planet surface was correctly simulated by EGP, as showed by Figure 8.

Moreover, many experiments were carried out and were aimed to evaluate different aspects of EGP achieved operations. First of all the only vision system performances were assessed, in order to establish if the image processing was robust enough to face real conditions and to determine the success of the demonstration. The effectiveness of this vision algorithm was confirmed by these preliminary experimental results, in terms of both convergence velocity and precision: Eurobot was able to center the marker within few seconds and the vision guided phase ended up with a maximum linear error smaller than 0.15 cm and an angular error below 0.03 radians per second (i.e. more or less 1.72 degrees). Furthermore, thanks to the precision obtained from the previous vision based control strategy, the force phase could begin with small residual errors to be compensated. This allowed the robot to correctly grasp the ORU, even if some limitations were highlighted. Finally, thanks to the completion of the simple ORU grasping task, more complex operations were performed: for example, failed equipment replacement was achieved as a combination of different grasping tasks, as showed by frames in Figure 9.

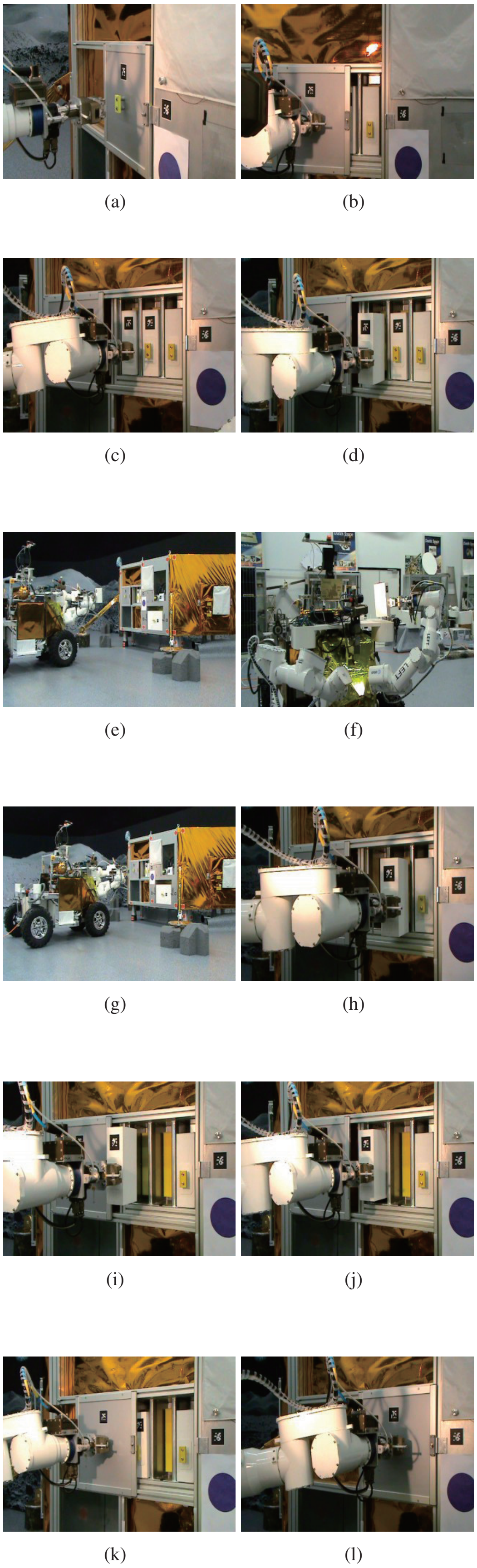


Figure 9. EGP equipment replacement

6 Discussion and Final Conclusions

As already suggested, the main goal of space robotics is provide a valuable support to mission crew members, relieving them of dangerous and hard work, thus saving their valuable time: the achievement of intelligent autonomous systems that could actively and reliably perceive surrounding environment and act accordingly to each potential situation is a fundamental goal for future planetary exploration activities.

EGP project gave positive results, even if some problems arose: one of the major limitations is due to EGP end-effector length: on each arm, the JR3 force/torque sensor is mounted on the wrist and the contact with external objects occurs at the palm level. As the distance between wrist and palm is remarkable, the force feedback turns out to be not enough accurate and, in some cases, it can be even confused with the sensor noise. For what concerns vision system, within Eurobot programme, ESA confirmed the choice to employ markers in order to simplify the accomplishment of autonomous tasks, considering the high level of reliability thus achieved. Nevertheless, removing markers from the scene and tracking object natural features could represent a great improvement for EGP vision system, because it would allow engineers not to count on a rigidly structured environment and to accomplish the same operations, even if some kind of error occur during the mission.

In conclusion, good interaction skills and autonomy capabilities were achieved and embedded into Eurobot behavior; these features, however, are not independent from the specific application and context and rely on a partial knowledge of the environment (through marker presence). While, on one hand, this increases EGP reliability, on the other side it restricts its area of applicability: such a robot cannot yet be employed, for instance, with the aim to recognize, all on its own, a planetary surface region with some scientific interests and to take samples. Furthermore, for what concerns robotic coordination, a suitable control algorithm has been developed; a positive aspect consists in the possibility to employ it in different situations, no matter the structure of the robot to be controlled. Moreover, as already explained, the definition of several control objectives is possible and easy to implement. For all these reasons, EGP represents a satisfactory answer to the need of increasing the quality level of robotic support in space missions and can be considered as a first step toward more and more accurate and complex systems.

References

[1] R.O. Ambrose, H. Aldridge, R.S. Askew, R.R. Burridge, W. Bluethmann, M. Diftler, C. Lovchik, D. Magruder and F. Rehnmark, “Robonaut: NASA’s space humanoid”, IEEE Intelligent Systems and their Applications, August 2000, pp. 57-63.

[2] S. Michaud, M. Dominguez, U. Nguyen, L. Zago and S. Droz, “Eurobot End-Effectors”, Proceedings of the 8th ESA Workshop on Advanced Space Technologies for Robotics and Automation, 2004.

[3] S. Estable, T. Hülsing, I. Ahrns, H.G. Backhaus, B. Maediger, J. Pizarro De La Iglesia and F. Didot, “Eurobot Monitoring & Control Station”, Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space, 2008.

[4] E. Zereik, “Space Robotics Supporting Exploration Missions: Vision, Force Control and Coordination Strategies”, University of Genoa, April 2010.

[5] G. Casalino and A. Turetta, “Coordination and Control of Multiarm Nonholonomic Mobile Manipulators”, Springer Tracts in Advanced Robotics, Springer Berlin/Heidelberg, 2004, pp. 171-190.

[6] G. Casalino, A. Turetta and A. Sorbara, “Dynamic Programming based Computationally Distributed Kinematic Inversion Technique”, Advanced Space Technologies for Robotics and Automation, Noordwijk, The Netherlands, 2006.

[7] G. Casalino, A. Turetta and A. Sorbara, “Distributed Control Architecture for Self-reconfigurable Manipulators”, International Journal of Robotics Research, March 2008, pp. 481-504.

[8] M. Uchiyama and P. Dauchez, “Symmetric kinematic formulation and non-master/slave coordinated control of two-arm robots”, Advanced Robotics, 1993, pp. 361-383.

[9] P. Chiacchio, S. Chiaverini and B. Siciliano, “Direct and inverse kinematics for coordinated motion tasks of a two-manipulator system”, ASME Journal of Dynamic Systems, Measurement, and Control, 1996, pp. 691-697.

[10] R. Hartley and A. Zisserman, “Multiple View Geometry in Computer Vision”, Cambridge University Press, 2004.

[11] E. Trucco and A. Verri, “Introductory Techniques for 3-D Computer Vision”, Prentice Hall, 1998.

[12] B. Kainz and M. Streit, “How to Write an Application with Studierstube 4.0”, Technical report, Graz University of Technology, 2006.

[13] J. Cai, “Augmented Reality: the Studierstube Project”, Seminar report.

[14] G. Bradski and A. Kaehler, “Learning OpenCV: Computer Vision with the OpenCV Library”, OReilly Media, 2008.

[15] T. Kröger T, D. Kubus and F.M. Wahl, “6D Force and Acceleration Sensor Fusion for Compliant Manipulation Control”, IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006.

[16] B.J. Waibel and H. Kazerooni, “Theory and Experiments on the Stability of Robot Compliance Control”, IEEE Transactions on Robotics and Automation, 1991, pp. 95-104