MULTIROBOT LIGHTWEIGHT OBJECT MANIPULATION AND ASSEMBLY WITHIN SPACE APPLICATIONS

G.Casalino, D.Angeletti, A.Caffaz, K.Redini

DIST – University of Genova, Via Opera Pia 13, 16145 Genova, Italy
{pino,henoch,caffaz,redkat}@dist.unige.it

ABSTRACT

The paper describes the results and the experience gained within the development of the functional and algorithmic control architecture for a dual arm workcell composed by two 7-d.of. electrically actuated robotic manipulators, presently located at the GRAAL – DIST labs of the University of Genova.
The workcell now also serves as one of the reference test-bed made available to ASI for experimenting and validating (possibly in a multirobot coordinated environment) the use of manipulative task control methodologies and techniques.

1. INTRODUCTION

The paper describes the results and the experience gained within the development of the functional and algorithmic control architecture for the dual arm workcell of fig. 1,

Fig.1 .The dual arm workcell

composed by two 7-d.of. electrically actuated robotic manipulators, presently located at the GRAAL – DIST labs of the University of Genova.
Formerly realized during the last three years, within the EC funded project AMADEUS (Advanced Manipulators for Deep Underwater Sampling) the workcell now also serves as one of the reference test-bed made available to ASI for experimenting and validating (possibly in a multirobot coordinated environment) the use of manipulative task control methodologies and techniques. This in the perspective of a possible implementation of such techniques within the framework of the NASA-ASI project PAT (Payload Tutor) specially devoted to the development of robotic systems capable of executing human-like, lightweight objects manipulative tasks, on board of unmanned orbiting vehicles.
The definition, development and experimental evaluation, of the functional and algorithmic control architectures most suitable for the accomplishment of the required manipulative tasks, actually constitutes the goal that DIST has proposed to attain as the final output of a currently on-going basic research project, supervised and supported by ASI.
In this latter framework, scope of the of the present work will be consequently that of reporting about the project via a presentation, in some details, of the major milestones which have been established till the current stage of development.
To this aim, the overall functional and algorithmic control architecture will be gradually described by following the same bottom-up modular design approach, which lead to modular architecture organized into four distinct levels of a control hierarchy.
A such hierarchy results first constituted by a Very Low Level Control layer (VLLC), performing the independent velocity control of each robot in the joint space, then followed by an upper Low Level Control layer (LLC), where the end-effectors positions are independently controlled within each one of the corresponding cartesian spaces. Then a third Medium Level Control layer (MLC) is added to the hierarchy, playing the role of closed loop coordinating the end-effectors motions, during the execution of cooperative tasks. Finally, a fourth High Level Control layer (HLC) is also added, acting as a discrete event task scheduler, monitor and closed loop exception handler.
A brief description of the adopted underlying Hw/Sw architecture implementing the overall functional control structure will be given in the final part of the paper. This will also include a very brief description of the SW design tools purposely developed and used (together with others of commercial type) for “fast prototyping” the overall real-time control software.

2. THE INDEPENDENT MOTION CONTROL OF THE ARMS

Consider fig. 2, where the kinematic structure of one of the seven d.o.f arms of the workcell is schematically represented. In this figure, T_e is the transformation matrix of the end-

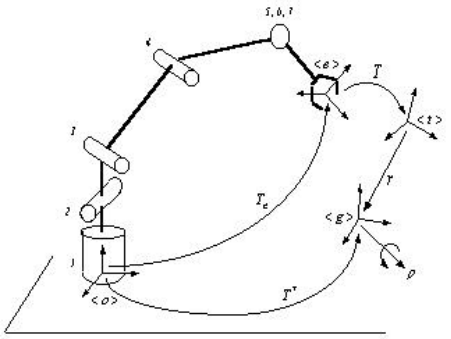

Fig.2. Kinematic structure for a single arm.

effector frame <*e*> w.r.t. base frame <*o*>, *T* is the (constant) transformation matrix of the tool center frame <*t*> w.r.t. <*e*>, and T^* the (generally time varying) transformation matrix of the reference frame <*g*> (w.r.t. <*o*>) to be tracked by tool frame <*t*>. Moreover, vectors *r* and *r* (both projected on the base frame <*o*>) represent the distance and the misalignment (equivalent rotation vector) of reference frame <*g*> w.r.t. <*t*>. By collecting together the two error vectors, the six dimensional global error vector *e* is defined as $e = [\, r \;, r \,]^T$.
The control scheme for a single arm is schematically depicted in fig. 3, where the block "*Robot + VLLC*" (Very Low Level

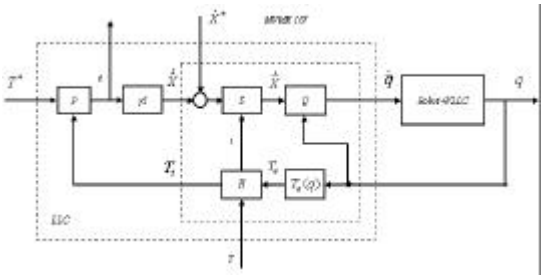

Fig.3. Single arm control scheme

Control) represents the physical arm equipped with its seven joint drivers, each one implementing a closed loop velocity control at the corresponding joint. Thus such block can be seen as a compact one receiving reference joint velocities as input, while giving the corresponding joint positions as outputs[1].
The remaining part of the scheme, termed as the "Low Level Control" (LLC) loop, is instead composed by the interconnection of the following blocks.
The processing block *P*, used for real time evaluating the global error *e* via the solution of the well known "versor lemma" equations (Refs. 1, 8) for its rotational part *r*, and via the difference between the first three elements of the last columns of T^* and T_t, respectively, for its linear part *r*.
The gain matrix γI ($\gamma > 0$), used for translating the global error *e* into the generalized cartesian velocity vector $\hat{X} = [\hat{w}' \quad \hat{r}']^T$ (it also projected on <*o*>) to be assigned to the tool frame T_t in order to have the convergence of *e* toward zero.
The additional cartesian velocity input X^*, only used for coordination purposes (as described in section 4).
Then block *S*, translating $\hat{X}$ into the related cartesian velocity $\bar{X}$ for the end effector frame <*e*>, via the use of the well known rigid body velocity relationships

$$\begin{cases} \bar{w} = \hat{w} \\ \bar{v} = \hat{v} + [s \wedge]\hat{w} \end{cases} \tag{1}$$

where *s* is the vector distance (projected on <*o*>) of frame <*t*> with respect to <*e*>; i.e. vector

$$s = R_e\, t \tag{2}$$

With R_e the rotation matrix part of the end-effector frame transformation matrix T_e, and *t* the same (constant) distance vector projected on the end-effector frame <*e*> (i.e. the first three elements of the last column of the constant transformation matrix *T*).
Finally, within the same scheme block *H* is used for implementing the simple transformation relationship

$$T_t = T_e\, T \tag{3}$$

within the end-effector Cartesian velocity reference signal $\bar{X}$, evaluated as above indicated, is in turn transformed via the functional block *Q*, into a corresponding set of joint velocity reference inputs, globally represented by the already mentioned seven dimensional vector $\dot{\bar{q}}$.
Since the interface block *Q* must actually face with some aspects of critical nature, a description of its implementation is now presented with some details.
To this aims consider again the redundant structure as depicted fig. 4, showing the most important singular configuration

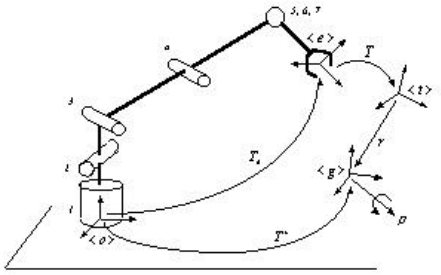

Fig.4. Singularity position at joint 4 level

exhibited by the arm. As it is well know, in the vicinity of such configuration joint velocities produced via the simple pseudoinversion of the jacobian matrix would tend to infinite values; thus inducing (at least temporarily) instabilities and not acceptable vibrations within the whole structure.
Thus the need for regularization naturally arises, together with that of however maintaining the arm sufficiently "far" from

[1] Substantially this block behaves as the time integral of joint velocities provided that sufficiently high bandwidth loops are guaranteed by the VLLC system itself.

such configuration. Then, in order to satisfy to the above needs, the following algorithm has been finally adopted for generating $\dot{\bar{q}}$ from the inside of the interface block *Q*

$$\dot{\bar{q}} = W^{-1} J^T (JW^{-1}J^T + \mathbf{II})^{-1} (\dot{\bar{X}} - h_4 \dot{\tilde{q}}_4) + \dot{\tilde{q}} \qquad (4)$$

$$\begin{aligned} \dot{\tilde{q}} &= [0, 0, 0, \dot{\tilde{q}}_4, 0, 0, 0]^T \\ \dot{\tilde{q}}_4 &= -k\,(q_4 - \bar{q}_4) \end{aligned} \qquad (5)$$

The rationale underlying the above expression can be simply explained as follows.

First of all note that term $\dot{\tilde{q}}$ corresponds to an elastic feedback tending to maintain the fourth joint far from its zero value. The relevant elastic gain k has been chosen in such a way to substantially act only in the vicinity of the zero position; thus resulting in a finite support, bell shaped, positive radial basis function of q_4 centered on zero.

Due to the presence of $\dot{\tilde{q}}$, the first term in (4) is consequently used for both compensating its Cartesian effects $h_4 \dot{\tilde{q}}_4$ (h_4 is the fourth column of the end-effector Jacobian matrix *J*) while also guaranteeing the tracking of the required Cartesian velocity $\dot{\bar{X}}$. This obviously corresponds to perform a projection of $\dot{\tilde{q}}$ on the null space of *J*, which can be however done by minimally depressing the elastic action given by $\dot{\tilde{q}}_4$; via the use of the following Jacobian weighting matrix *W*

$$W \doteq diag\,[1,1,1,w,1,1,1] \;\; ; \;\; w \doteq 1 + k \qquad (6)$$

Finally note in (4) how the term is also added for regularization purposes, taking it also the form of a finite support, bell shaped, positive radial basis function (centered on zero) of the determinant of the matrix $JW^{-1}J^T$ (or, even better, of the ratio between its minimum and maximum singular value).

With the adoption of the above technique, it can be shown that singularity can always be avoided provided that $\dot{\bar{X}}$ is guaranteed to always lie within specific norm bounds, which can be however enlarged for increasing values of *k* (Ref. 8).

In fig. 3, the sub-scheme enclosed in the inner dotted box constitutes what is generally termed as the "Interface part between the Cartesian and the joint spaces" of the overall scheme.

The structure of the overall scheme of fig.2 has been proposed by some of the authors since 95 (Ref. 3); its nice stability and robustness properties, with respect to possible uncertainties of both dynamic and geometric nature, have been they also proved by the authors within different reports and papers. Theoretical details can be for instance found in the works (Refs. 8, 3, 10, 1-2).

3. THE TELEOPERATED, INDEPENDENT MOTION CONTROL OF THE ARMS

A teleoperated control mode has been actually chosen for driving the end effector of each one of the arms; which means that the corresponding reference frame is made to move around the space by integrating velocity data (angular and linear) acquired from the HCI[2] via the use of a suitable "space mouse" device interacting with it.

As it will be however better explained shortly, a simple integration of the velocity data coming from the space mice may not suffice for effectively driving each one of the arms. As a matter of fact, the more complex functional diagram depicted in fig. 5 has revealed to be the most suitable one for

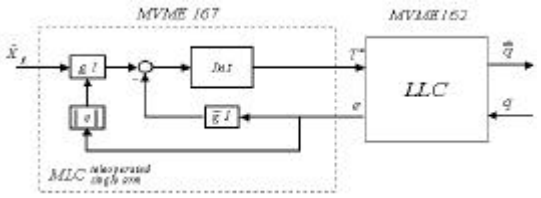

Fig.5. MLC for a single arm

teleoperation purposes (Ref. 8).

In order to explain the rationale underlying the use of the proposed scheme, let us first assume for a while that blocks named *gI* and *gI* simply correspond to the identity and the null one, respectively (i.e. *g=1* and *g=0*); then in this case the diagram directly reduces to the sole integration, performed by the block named *Int*, of the acquired absolute velocity data w_g and v_g assigned to the goal reference frame <*g*>.

Obviously enough, provided that the absolute velocities data w_g and v_g are interpreted as projected on the absolute frame <*o*>, block *Int* results in performing the integration of the following differential equations (with the first of the two efficiently integrated via the use of the well known "Rodriguez formula" (Ref. 12))

$$\begin{cases} \dot{R}^* = [w_g \wedge] \cdot R^* \\ \dot{p}^* = v_g \end{cases} \qquad (7)$$

In the above equations the starting conditions are always set to be equal to the initial posture assumed by the tool frame <*t*> of the arm. The resulting absolute reference outputs R^* and p^* (rotation matrix and position, projected on <*o*>, of <*g*> w.r.t. <*o*>) are then assembled together in order to give rise to the corresponding transformation matrix T^*.

It should be however noted that in this case, provided that the sole block *Int.* were actually used, there might be the risk of having (at least temporarily) the tracking error *e* attaining very high values before eventually con verging to zero. This might for instance happen in case of a "fast" virtual motion of the space mouse or, more commonly, when the target frame <*g*> is pulled too far outside the robot workspace. In particular, since in this latter case the robot is progressively pushed toward its singularity while trying to reduce errors that instead (due to mechanical constraints) could even attain unbounded values, very high values for $\dot{\bar{X}}$ may consequently be induced within the scheme of fig. 3. Then, just due to this,

[2] Developed by IAN-CNR of Genova-Italy, partner of the AMADEUS project

it results that the assumed norm bounds of $\dot{X}$ (i.e. these allowing the effective avoidance of singularity, as mentioned in sect. 2) generally risk to be seriously overpassed.

As a matter of fact, since this drawback could be avoided by simply forcing the tracking error *e* to remain confined within a specified superior norm bound, this is consequently done via the use of the block *gI*, having the gain *g* taking on the form of a finite support, bell shaped, positive radial basis function centered on zero, of the tracking error norm. As it can be easily seen, since the velocities of the target frame are now progressively reduced toward zero in case the error *e* approaches the specified norm limit, it consequently occurs that error *e* itself cannot ever overpass such norm bound (obviously provided that, as it is, it always starts from lower norm values).

With the sole use of the above described "norm error limiter" block it is however clear that, in case of attainment of the specified norm error bound in correspondence of points located outside of the robot workspace (where the error itself cannot be anymore reduced via robot motion, and where the velocity gain *g* however attains the zero value) unfortunately enough there will not be anymore the possibility of moving again the goal frame <*g*> (this due to the fact that, in the impossibility of reducing the error, its velocity will be consequently maintained at a null value by the *gI* block itself).

Then, in order to also avoid such an additional drawback, a moderate, tracking error based, constant velocity feedback, has been consequently added via the constant gain block termed as *gI.*

As a matter of fact, since such block always exerts a moderate recalling action on <*g*> itself, directed toward the tool frame <*t*> of the arm, it always prevents the error *e* to exactly reach its upper norm bound; thus always avoiding the total zeroing of the gain in the *gI* block.

Obviously enough, when the arm is instead inside its work space, the motion capabilities of the arm itself (i.e. the capability guaranteed by the LLC gains of maintaining *e* inside its lower limits closed to zero, (provided *e* itself starts from the inside of such limits) simply result in a “masking” of the (moderate) feedback action produced by the *gI* block.

The final assessment of the functional scheme reported in fig. 5 corresponds to what has been termed as the MLC (Medium level Control for the teleoperation of a single arm) where motion tasks are handled by totally operating within the relevant Cartesian space.

4. THE TELEOPERATED, COORDINATED MOTION CONTROL OF THE ARMS

The situation depicted in fig. 6 shows the need of being able to move both the end-effectors without actually varying their mutual position and orientation, during lightweight object teleoperated transformations).

In order to develop such capability, let us start from the (assumed steady) condition when the object has been firmly grasped by both the grippers after a preliminary phase of object approach, and consequent grasping, performed via teleoperated independent control mode (see fig. 6).

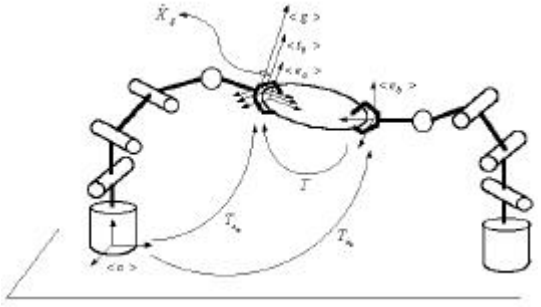

Fig.6. Teleoperated dual arm workcell handling an object

Then the existing transformation matrix T between the two end-effectors must be preliminary evaluated (on the basis of known geometrical informations) via the expression

$$T = T_{e_b} T_{e_a} \tag{8}$$

A tool center frame <t_b>, having the constraint matrix T as transformation matrix w.r.t. <e_b>, is then assigned to arm *b*; while the tool center frame <t_a> of arm *a* is let to coincide with its end effector frame <e_a>.

Then the coordinated motion of the arms can be performed on the basis of a "parallel task composition", to be implemented in the following way.

1) The tool frame <t_b> is asked to remain coinciding with the tool frame <t_a>. 2) The tool frame <t_a> is asked to remain coinciding with the tool frame <t_b>. 3) Meanwhile, the tool frame <t_a> is also asked to closely follow the moving goal frame <*g*> (supposed to be initially coinciding with <t_a>). Such situation, actually indicated in fig. 6, conceptually corresponds to what is also reported in fig. 7, where, the

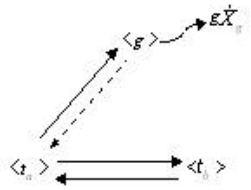

Fig.7. Frame tracking arrangement for teleoperated coordinated motion

dotted arrow is also used for indicating the “attractive” action contemporarily performed on $< g >$, toward <t_a>, by the (moderate) feedback produced by the still assumed presence of block *gI*, accomplishing to the same role as described in section 3.

As it is apparent from fig. 7 (and in accordance to what has been above established) while frame <t_b> is required to track a single frame (actually frame <t_a>) frame <t_a>is instead asked to contemporarily follow two different target frames (namely, frame <t_b> itself and the moving one <*g*>).

The same obviously holds also for frame <*g*> itself, which is required to move around space while also tracking, due the presence of the previously mentioned moderate feedback action, frame <t_a>.

To this respect, it must be however noted how such seemingly

contrasting aspects actually are of only apparent nature: this simple due to the fact that the above established global task actually admits a unique solution, contemporarily fulfilling to all the composing subtasks (i.e. the one corresponding to have, eventually or just from the beginning, all the frames coinciding among them (Ref. 11)).

Seen from the implementative point of view, the control scheme allowing the execution of the above specified overall task consequently results to be that of fig. 8, corresponding to

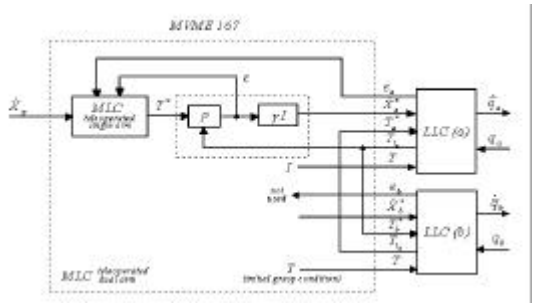

Fig.8. Control scheme for teleoperated coordinated motion, corresponding to the conceptual scheme of fig.7.

a suitable composition of the same control modules already defined within the independent control of the arms.

Within this figure, the two blocks *LLC(a)* and *(b),* located at the right, are the same control schemes already used for the independent control of the two arms; with the only difference that, now, each one of them receives, as input reference frame to be tracked, the tool frame of the other. Moreover, block *(a)* also receives, as an additional input, the Cartesian velocity control signal X_a^* generated by the external control loop (the one enclosed in the inner dotted box) delegated to control the tracking of $<g>$ by part of $<t_a>$. Furtherly, the transformation matrix T^* of the moving frame $<g>$ is generated via a mechanism (the leftmost block) which is the same of that described in section 3.

As it concerns this last block, a slight difference with the one used in section 3 however exists. Such difference simply consists on the fact that now the block admits, other than the tracking error *e* between $<g>$ and $<t_a>$ inputs as feedback (as before), also the "internal" one e_a, between $<t_a>$ and $<t_b>$. In this situation, while *e* is still singularly used as input to the constant feedback block *gI,* joined with e_a is instead employed for defining the quantity

$$x \doteq sqrt(\|e\|^2 + h\|e_a\|^2) \;\; ; \;\; h > 0 \qquad (9)$$

which is now used as argument to the gain *g* of block *gI* (obviously still having the form of a finite support, bell shaped, positive radial basis function centered on zero) where its finite support extends within the validity of a (suitably defined) threshold condition £ .

As a matter of fact, with the use of the above argument, and related suitable choices for the weighting parameter *h* and threshold value , it becomes possible to coordinately move both the arms while avoiding to both errors the overpassing of the corresponding, prespecified, upper norm bounds; namely: *e/ h* for e_a and *e* itself for *e.*

The entire scheme globally enclosed within the dotted box in the left side of figure 8, represents what is termed to be the MLC part of the teleoperated coordinated control mode; since it also completely operating at a Cartesian level only, by following modalities which, result they also, structurally independent from the underlying specific robotic system.

Before concluding the present section, it might be interesting to make some considerations concerning a possible variant to the previous scheme; i.e. the one corresponding to the "task

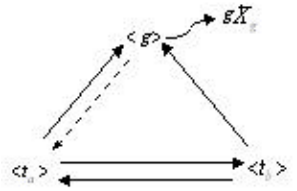

Fig.9. Another possibility of frame tracking arrangement for teleoperated coordinated motion.

composition" diagram of fig. 9 and the relevant control scheme of fig. 10. Within such variant, tool frame $<t_b>$is it

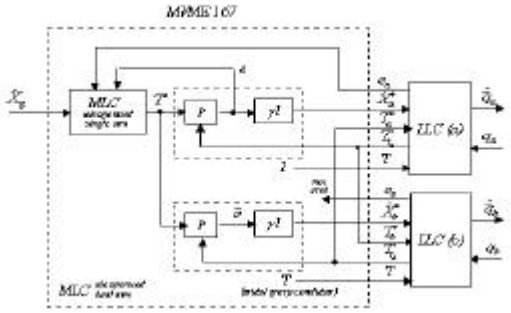

Fig.10. Control scheme for teleoperated coordinated motion, corresponding to the conceptual scheme of fig.9.

also asked to follow the moving frame $<g>$, as it is for $<t_a>$ within the diagram of fig.7.

As a result of such modification, the inherent priority previously existing between the tasks indicated in fig. 7 (resulting in a role of preminent "master frame" substantially played by frame $<t_a>$) is now totally lost in favor of a more equilibrated distribution of the tasks within the various involved all frames.

Moreover, the entire set of frames now results to be “more binded” (by using a colloquial expression) than that of fig. 6, due to the redundant existence of additional feedback control action.

5. THE AUTOMATIC, INDEPENDENT MOTION CONTROL OF THE ARMS

The so called "Automatic, Independent Motion Control Mode" is here intended occur when the end effector of each arm is asked to reach a whatever assigned position/orientation in space, starting from a whatever initial posture. Within such kind of control (reentering within the more general category of the so called "point-to-point" control modes) the

position/orientation trajectory followed while reaching the final posture is automatically established by the implemented corresponding control loop. Then, it is just in this sense that such kind of control mode differs from the already considered teleoperated one, where the trajectory is instead on line planned and assigned by the operator interacting with the system (via a continuos time reference velocity profile, generated by a space mouse device).

In order to implement such kind of control mode it should be however clear that, at least in principle, the sole LLC fundamental scheme of fig.3 would actually suffice provided that it could be somehow assured that the induced internal cartesian velocities $\bar{X}$ remain norm bounded in order to guarantee the possibility of singularity handling (see section 2.).

As a matter no fact, since a way for guaranteeing such possibility is (similarly to what have been discussed in section 2.) that of always maintaining the global error e within a suitably assigned maximum norm bound, the conceptual scheme of fig. 11 can actually be suggested.

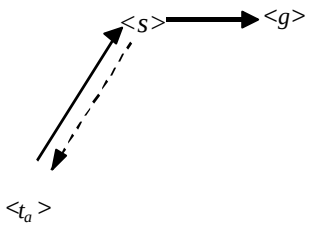

Fig. 11. Frame tracking arrangement for the automatic independent motion

In such scheme, frame <*s*> is an auxiliary one that, starting from a *posture* coinciding with the tool frame <*t*>, is commanded to reach the goal frame <*g*> via the use of a cartesian control loop whose feedback gain still takes on the same form "*gI* " as in section 2 (i.e. with *g* of the form of a finite support, bell shaped, positive radial basis function centered on zero, of the norm of the error *e* between frame <*s*> itself and <*t*>).

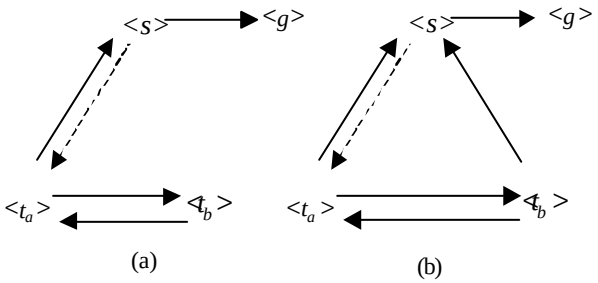

Fig. 12. Frame tracking arrangement for the automatic mode

Meanwhile frame <*t*> is commanded to follow <*s*> (and a moderate constant feedback action still of the form " $\bar{g}I$ " also exists, weakly recalling <*s*> toward <*t*>).

As it can be easily realized, this guarantees the boundedness of the tracking error *e* (of <*s*> w.r.t. <*t*>), and also its convergence to zero together with the other tracking error *e'* (of <*g*> w.r.t. <*s*>) provided <*g*> is inside the arm work space. Otherwise (<*g*> outside the arm work space) both the errors *e, e'* only will remain bounded (in particular with the norm of *e* inside its upper limit) eventually reaching the equilibrium condition

$$g(\| e \|)\, e' - \bar{g}\, e = 0 \qquad (10)$$

only satisfied by a corresponding vector e whose norm is necessarily strictly lower then its assigned upper bound.

The functionals scheme corresponding to the above conceptual one is reported in the following fig.13

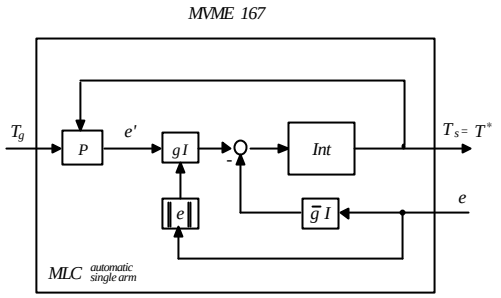

Fig. 13. Medium Level Control for automatic independent motion

As it can be seen from the figure, the part of control concerning the tracking of <*s*> by part of <*t*> is let to the LLC of the arm. Only the remaining part of the scheme is implemented a as MLC.

6. THE AUTOMATIC, COORDINATED MOTION CONTROL OF THE ARMS

The same idea underlying the Automatic Independent motion Control previously seen, can be easily extended to the case of Automatic Coordinated Motion Control via the use of one of the conceptual schemes of fig. 12 (where the right figure is an improvement of the left).

To the above conceptual schemes there obviously correspond

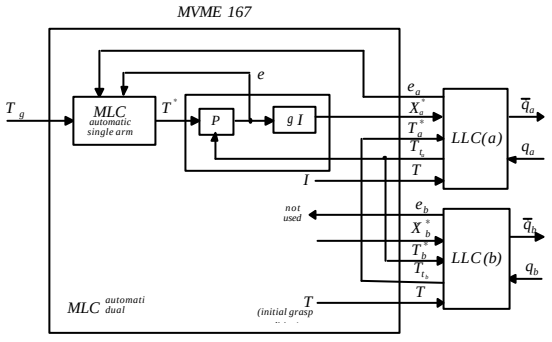

Fig. 14. Control scheme for automatic coordinated motion, corresponding to the conceptual scheme of fig.12(a).

the functional ones of figg. 14, 15; which are in turn strictly analogous with the teleoperated ones of figg. 8, 10, since only

differing for the internal presence of the functional block "*MLC automatic single arm*" in lieu of the teleoperated one.
As it can be seen from the above figures, the part concerning the mutual tracking between frames $< t_a >$ and $< t_b >$ is let to the two LLC part of the overall scheme, in the teleoperated, coordinated motion case.

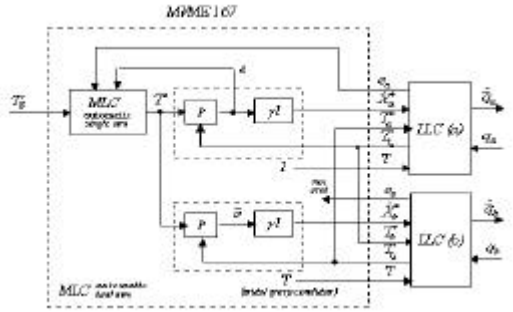

Fig.15. Control scheme for automatic coordinated motion, corresponding to the conceptual scheme of fig.12(b).

Also note how, by following the same identical philosophy which has been adopted in section 5, also the block "*MLC automatic single arm*" now receives as input both the "external" tracking error *e* and the "internal" one e_a .

7. THE HW / SW IMPLEMENTATION

The control schemes have been developed using the well known Simulink and RealTimeWorkshop tools, which easily allow a fast graphical programming of the overall scheme and extensive simulation implementation on a PC platform and VxWorks VME based targets for the real time control followed by automatic coding and down-loading on.
Customized blocks libraries performing the numerical regularized inversion (SVD algorithm) using the C-code of the Linpack library have been also developed.
The blocks used for evaluating the various transformation matrices and Jacobians of each one of the arms via optimized analytical expressions, have been automatically and preliminary deduced within the so called Robotics Developer Studio environment (Refs. 13, 5, 9): a purposely developed, Maple based, symbolic modeler for complex mechanical structures; then automatically translated into optimized C-code language, and eventually imported into the Simulink environment.
Finally, synchronized exchanges of data among processes, within real-time environments and multiprocessors architectures has been also realized via transmitting and receiving "channels" (Refs. 5, 7, 6) implemented in the C code and translated into simulink blocks.
As a consequence of the above developing process, each one of, the resulting real-time software systems is made running on the corresponding MVME board controlling the relevant arm; that is: a MVME162 for each one of the LLC of the arms, and one MVME 167 for the MLC of one or both the arms.
All the data are transmitted, received and processed, within a fixed sampling interval of 5 msecs. An exception is instead made for the matrix inversion via SVD, which is completed within a fixed value of six elementary sampling intervals; that is every 30 msecs.

8. THE FINITE STATE AUTOMATA SCHEDULING NETWORK

The "State-Flow" facility, very recently provided by the Matlab/Simulink environment for designing finite state Automata networks, has been used for scheduling all the operation modalities commanded by suitable logic signals provided by the HCI system.
The so designed network can be integrated with the continuos time schemes and automatically translated into optimised C-code (via the Real Time Workshop facility), then compiled and downloaded within the multiprocessor Hw architecture and Wx-Work distributed real-time operating system, by following the same, already discussed, modalities. Then, as a consequence of this, the network itself has been naturally installed at the MLC control of the overall system and integrated with all the various schemes of concern; i.e. made running within the MVME 167 master computing board, together with the related, scheduled, continuos time software.

9. CONCLUSIONS

The paper has described the complete functional and algorithmic architecture which have been adopted and implemented for the control of the dual arm workcell system.
The adopted philosophy of approach has clearly shown the possibility of eventually achieving the complete design by proceeding step by step via a suitable composition of simpler modules developed at previous stages of the design.

ACKNOWLEDGEMENTS

This paper has been supported by the UE funded "AMADEUS" project, under contract MAS3-CT95-0024 - DGXII D.3, national, the MURST funded, "RAMSETE" project on Robotics, and by ASI under contract ASI-ARS-98-172

REFERENCES

1. Aicardi M., Cannata G., Casalino G. (1996a). "Task Level Position Control of Robots, and Static Errors Compensation", *Proc. 2nd World Automation Congress WAC '96*, Montpellier, France.
2. Aicardi M., Cannata G. , Casalino G. (1996b). "Task Space Robot Control: Convergence Analysis and Gravity Compensation via Integral Feedback", *Proc. 35th IEEE CDC Conference*.
3. Aicardi M., Caiti A., Cannata G., Casalino G. (1995). "Stability and Robustness Analysis of a Two layered Hierarchical Architecture for the Control Of Robots in the Operational Space", *Proc. Of the IEEE Int. Conf. On Robotics*

& Automation, Nagoya, Japan.

4. Angeletti D., Cannata G., Casalino G. (1997a). “Task Function Based Control for a Dual Arm Underwater Robotic System”, *Proc. Symp. MMAR ‘97.*

5. Angeletti D., Cannata G., Reto S (1997b) “Design of Multiprocessor Robot Control Systems Using Simulink”, *Proc. of the1997 Italian Conference of MATLAB Users.*

6. Angeletti D. (1999a). "Communication Channels for VME bus Based Multiprocessor Systems Architectures", *Graal-Dist Technical Memo TM/99/1* (in Italian).

7. Cannata G., Casalino G. (1998). “Task Level Design of Robot Control Systems”, *Graal-Dist Technical Memo TM/98/1*.

8. Casalino G., D. Angeletti, G. Cannata, Marani G. (2000) "On the functional and alghoritmic control of the "Amadeus" dual arm robotic workcell" , World Automation Congress (WAC 2000) Maui, Hawaii

9. Casalino G., Angeletti D., Cannata G. Marani G. (1999). "Symbolic based Model Builder for Multibody Systems", *Fifth International Conference on Information Systems, Analysis and Synthesis (ISAS'99)*, Orlando, USA, August .

10. Casalino G. (1996/97a). "Hierarchical Dynamic Control of Robotic Manipulators", *In Notes from the course lessons in "Industrial Robotics"*, Faculty of Engineering, University of Genova, Academic year 1996/97 (in Italian).

11. Casalino G. (1996/97b). "Task Composition Within Planning and Control of Robotic manipulators", *In "Notes from the course lessons in Industrial Robotics"*, Faculty of Engineering, University of Genova, Academic year 1996/97 (in Italian).

12. J.J.Craig (1989). "Introduction to Robotics, Mechanics and Control", Addison-Wesley Publishing Company, Second Edition, New York.

13. Marani G., Casalino G. (1999). Robotics Developer Studio, Convegno Nazionale "Automazione 99", Roma.

14. Nakamura. Y .(1991). "Advanced Robotics: Redundancy and Optimization", Addison Wesley.