The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

An Experimental Evaluation of a Novel Minimum-Jerk Cartesian Controller for Humanoid Robots

Ugo Pattacini, Francesco Nori, Lorenzo Natale, Giorgio Metta, and Giulio Sandini

Abstract—**In this paper we describe the design of a Cartesian Controller for a generic robot manipulator. We address some of the challenges that are typically encountered in the field of humanoid robotics. The solution we propose deals with a large number of degrees of freedom, produce smooth, human-like motion and is able to compute the trajectory on-line. In this paper we support the idea that to produce significant advancements in the field of robotics it is important to compare different approaches not only at the theoretical level but also at the implementation level. For this reason we test our software on the iCub platform and compare its performance against other available solutions.**

I. Introduction

As researchers in the field of humanoid robotics we commonly find ourselves promising our funders that soon we will see the world populated by robots substituting humans in everyday tasks. At the same time – and quite sadly – in our laboratories students spend an enormous amount of time facing tasks that everyone considers quite easy or at least "already solved" in the literature. The problem is that, if on the one hand the scientific literature is full of papers describing techniques that solve virtually all possible tasks, on the other hand it is difficult to find the implementation of those techniques and use them out of the box. This is not to say that scientific papers are incorrect, do not contain good work or are not useful. It is true, however, that researchers put a lot of effort in writing papers and developing new techniques, but rarely concentrate on writing good implementations of these techniques and making them publicly available for comparison purposes or just as building blocks to work out more sophisticated tasks.

In particular when we decided to implement a Cartesian Controller for the iCub platform we found that it was not as easy as we initially expected. In humanoid robotics we often deal with kinematics structures that have a large (and variable) number of degrees of freedom. The problem is further complicated because trajectories have to be computed quickly in real-time. Finally, humanoid robots are programmed to produce smooth movements. All these aspects rule out the possibility to resort to expensive off-line solutions that are typically employed in industrial settings [1].

To deal with these issues we designed a novel Cartesian Controller which extends the Multi-Referential dynamical system approach [2] in two aspects. Firstly we have modified the trajectory generator to produce trajectories that have minimum-jerk profile. Secondly, we have applied an interior point optimization technique [3] to solve the inverse kinematics problem. We have shown that our solution has some advantages with respect to [2] and standard approaches in the literature [6], [7]. We also conducted an experimental comparison between our implementation and publicly available software, demonstrating the performance gain of our technique in terms of smoothness, speed, repeatability and robustness.

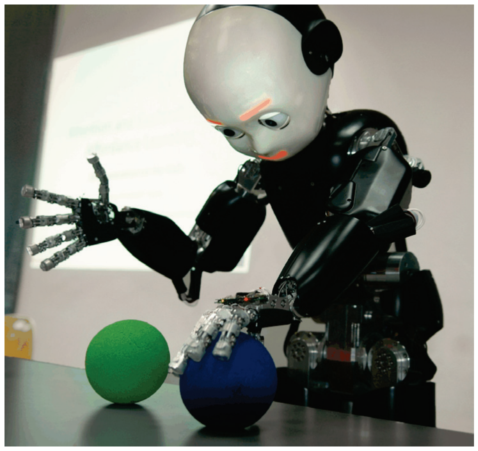
Fig. 1. The iCub robot.

II. Platform

The experiments reported in this paper were performed on the iCub [8]. The iCub is a 53 degrees of freedom humanoid robot (Fig. 1) shaped as a human child. The iCub was designed to study manipulation so most of the mechanical complexity is in the arms and hands, which are actuated by a total of 16 motors each (9 and 7 in the hand and arm respectively). The iCub is equipped with cameras, force sensors, and gyroscopes. All the software running on the iCub – including the software we describe in this paper – is released open-source (GPL) and is freely available for download.

Manuscript received March 10, 2010. This work was fully supported by European FP7 ICT project No. 215805 (CHRIS).

U. Pattacini, F. Nori, L. Natale, G. Metta, and G. Sandini are with the Italian Institute of Technology (IIT), Department of Robotics, Brain and Cognitive Sciences (RBCS), Via Morego 30, Genova, Italy (e-mail: ugo.pattacini@iit.it, francesco.nori@iit.it, lorenzo.natale@iit.it, giorgio.metta@iit.it, giulio.sandini@iit.it).

III. The Cartesian Controller

Given the Cartesian position of a target object, reaching is performed in two separate modules (Fig. 2). The first stage employs a *nonlinear optimization technique* to determine the arm joints configuration that achieves the desired pose (i.e. end-effector position and orientation). The second stage consists of a *biologically inspired kinematic controller* that computes the velocity of the motors to produce a human-like quasi-straight trajectory of the end-effector. In the following these two modules composing the structure of proposed Cartesian controller are analyzed in depth.

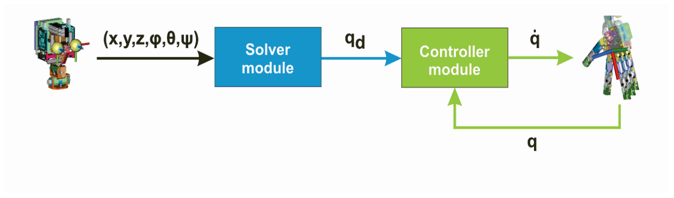

Fig. 2. Diagram of proposed Cartesian controller.

A. The Solver

We consider the general problem of computing the value of joint encoders $q^* \in \mathbb{R}^n$ in order to reach a given position $x_d \in \mathbb{R}^3$ and orientation $\alpha_d \in \mathbb{R}^4$ of the end-effector (where α_d is represented in axis/angle notation[1]). At the same time, the computed solution has to satisfy a set of given constraints expressed as inequalities. Formally this problem can be stated as follows:

$$q^* = \arg\min_{q \in \mathbb{R}^n} \left(\left\| \alpha_d - K_\alpha(q) \right\|^2 + \beta \cdot \left(q_{rest} - q\right)^T W \left(q_{rest} - q\right) \right)$$
$$\text{s.t.} \begin{cases} \left\| x_d - K_x(q) \right\|^2 < \varepsilon \\ q_L < q < q_U \end{cases}, \tag{1}$$

where K_x and K_α are the forward kinematic functions that respectively compute the position and the orientation of the end-effector from the joint angles q; q_{rest} is a preferred joint configuration, W is a diagonal matrix of weighting factors, β is a positive scalar weighting the influence of q_{rest} and ε is a parameter for tuning the precision of the movement: typically $\beta < 1$ and $\varepsilon \in [10^{-5}, 10^{-4}]$. Moreover, the solution to problem (1) has to comply with a set of additional constraints: for example, we required that the solution lies between lower and upper bounds ($q_L, q_U \in \mathbb{R}^n$) of physically admissible values.

In our case the joints vector has 10 components (7 joints for the arm, 3 joints for the torso) and we have chosen the value of q_{rest} so that the torso of the robot when controlled is as close as possible to the vertical position. We proposed to use an interior point optimization technique to solve the problem (1), in particular we used *IpOpt* [3], a public domain software package designed for large-scale nonlinear optimization. This approach has the following advantages:

1) *Quick convergence*. IpOpt is reliable and fast enough to be employed in real-time as demonstrated in the remainder, especially compared to more traditional iterative methods such as the *Cyclic Coordinate Descent* (CCD) [12] adopted in [2].

2) *Scalability*. The intrinsic capability of the optimizer to treat nonlinear problems in any arbitrary number of variables is here exploited to make the controller's structure easily scalable with the dimension n of the joint space. For example, it is possible to switch at run-time from the control of the 7-DOF iCub arm to the complete 10-DOF structure inclusive of the torso or to any combination of the joints depending on the task.

3) *Automatic handling of singularities and joint limits*. This technique automatically deals with singularities in the arm Jacobian and joint limits, and can find solutions in virtually any working conditions.

4) *Tasks hierarchy*. The task is split in two subtasks: the control of the orientation and the control of the position of the end-effector. Different priorities can be assigned to the subtasks. In our case the control of the position has higher priority with respect to the orientation subtask (the former is handled as a nonlinear constraint and thus is evaluated before the cost) because we deemed that to accomplish a successful grasp which is the ultimate goal of reaching for a humanoid it is central to cope with circumstances when the final object is attainable in position and thus can be touched, but the orientation cannot be reached perfectly at the same time.

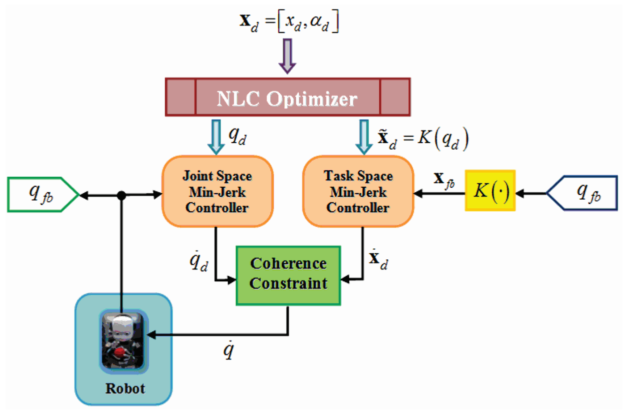

Fig. 3. Multi-Referential scheme. $K(\cdot)$ is the forward kinematic map.

5) *Description of complex constraints*. It is easy to add new constraints as linear and/or nonlinear inequalities either in task or joint space. In the case of the iCub, for instance, we added a set of constraints that avoid breaking the tendons that actuate the three joints of the shoulder: these tendons indeed are shared among the joints, whose movements are thus limited by the tendons lengths within a compact subset of the convex hull described by the physical joints bounds

[1] In axis/angle representation any rotation is described by a unit vector ω, indicating the direction of rotation, and an angle ϑ that accounts for the magnitude of rotation around the axis according to the right-hand rule; hence it holds: $\alpha_d = [\omega \ \ \theta]^T$, $\omega \in \mathbb{R}^3$, $\|\omega\| = 1$, $\theta \in [0, \pi]$.

[13]. Thereby, three linear inequalities hold among the shoulder joints that are conveniently included into (1) taking the form $l \le C \cdot q_{sh} \le L$, being q_{sh} the vector of the three shoulder joints, l, L the lower and upper limits imposed by the tendons lengths and C a suitable 3-by-3 matrix.

B. The Controller: design of minimum-jerk solution

The goal of the controller's module is to determine the smooth velocity profiles in the joint space which steer the arm from the current posture q to the final configuration q^*, while at the same time ensuring that the joints lie within well defined limits. This can be obtained by applying the Multi-Referential Dynamical Systems approach [2], in which two dynamical controllers, one in joint space and one in task space, evolve concurrently (Fig. 3). The coherence constraint between the two tasks – providing that $\dot{x} = J\dot{q}$ is guaranteed at each instant with J the Jacobian of the forward kinematic map – is enforced with the Lagrangian multipliers method and can be used to modulate the relative influence of each controller (i.e. to avoid joint angles limits). The advantage of such a redundant representation of the movement is that a quasi-straight trajectory profile can be generated for the end-effector in the task space reproducing a human-like behavior [5], [9], while retaining converge property and robustness against singularities [2].
In [2] the two controllers are implemented with Vector-Integration-To-Endpoint (VITE) models [4], which approximate the neural signal commanding a pair of agonist-antagonist muscles and whose behavior is regulated by a second order differential equation. According to the implementation in [2], the angular velocities, output of the coherence constraint block, are integrated to generate position references which are then sent to a second cascade controller that is in charge of yielding the velocity profiles in closed loop with a proportional law (Fig. 4).

Aside from the connection to biological evidences, a second significant merit of this approach is the description of the model given as a compact and autonomous dynamic equation which makes the controller implementation straightforward and robust against external perturbation of the movement creating an attractor landscape towards the goal, i.e. the target configuration [14]. On the other hand, the specific choice of a coupled second order dynamic systems in cascade with a P controller entails a major disadvantage when applied to the control of a robotic limb: notably, the generated velocity profiles become less human-like as the required execution time becomes shorter. When a fast response is requested, trajectories approach an exponential response (typical of a first order dynamical system), irrespectively of how the controller's parameters are tuned; therefore, the corresponding velocities are no longer bell-shaped, having a steep acceleration at the beginning followed by a slow decay. The reason is twofold: primarily a second order system cannot reproduce the smoothness typical of biological motion [9] (for example it does not impose zero acceleration at starting point) and secondly the presence of the proportional controller reduces the performances since it cannot guarantee the velocities computed by the coherence constraint block. As result fast movements tend to be jerky producing unwanted vibrations.

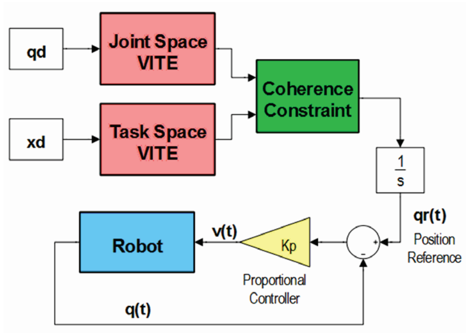

Fig. 4 Schematic of implemented multi-referential VITE controllers in the work of Hersch and Billard [2].

To overcome these limitations, we maintained the multi-referential approach and replaced the VITE with more complex controllers which reproduce a trajectory that resemble a minimum-jerk profile both in joint and task space: movements are still represented and controlled in multiple frames of reference but preserve a smooth (bell shaped) velocity profile. To this end one might consider to rely on a trajectory generator which codes for example the minimum-jerk profile over the time interval T and specific starting and ending points x_0, x_d: in literature [10] it is well known that the desired shape depicted in Fig. 5 (red lines) is given by the following fifth order polynomial:

$$x(t) = x_0 + (x_d - x_0)\left(10\left(\frac{t}{T}\right)^3 - 15\left(\frac{t}{T}\right)^4 + 6\left(\frac{t}{T}\right)^5\right). \quad (2)$$

The seeming straightforwardness of this formulation hides a number of somewhat important issues that need to be taken into account by an effective design: it is required indeed to generate an internal temporal scale t that has to be reinitialized any time the feedback is acquired modifying the coefficients of (2) on-line; moreover, the feedback turns out to be mandatory since the coherence block disturbs the true velocity command causing eventually drifts if not compensated by feedback. Therefore, a regulator appears to be a more natural answer for the task and joint space minimum-jerk elements. Possibly the generator (2) can be applied as the feed-forward component of the regulator, operating merely on the target position and leaving a PID to take into account the feedback: even so the PID would work hard to stabilize the response against the drift, delivering velocities that do not comply with the requisite of human-likeness. This ultimately suggests devising a controller that intrinsically embeds the desirable property of smoothness.

We took inspiration from the feedback formulation of the minimum-jerk trajectory as the solution of an optimal control problem as reported in [11], where a third order linear time-varying (LTV) differential equation is derived:

$$\begin{bmatrix}\dot{x}\\ \ddot{x}\\ \dddot{x}\end{bmatrix}=\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ -\frac{60}{(T-t)^3} & -\frac{36}{(T-t)^2} & -\frac{9}{T-t}\end{bmatrix}\begin{bmatrix}x\\ \dot{x}\\ \ddot{x}\end{bmatrix}+\begin{bmatrix}0\\ 0\\ \frac{60}{(T-t)^3}\end{bmatrix}x_d. \quad (3)$$

The weak point of (3) is that it contains coefficients whose values become infinite when *t* approaches the execution time *T*. To sidestep this difficulty, we decided to employ a linear time-invariant (LTI) system of the same order whose parameters are tuned to better approximate a minimum jerk trajectory: in other words we sought the third order differential system that is the best time-invariant version of (3) minimizing the same jerk measure over the interval [0, *T*]. Formally, we started from the parametric equation of the trajectory expressed in the form:

$$x(t)=C_1\exp(\lambda_1\cdot t)+C_2\exp(\lambda_2\cdot t)+C_3\exp(\lambda_3\cdot t)+x_d, \quad (4)$$

that is a particular solution of a stable third order differential system with three independent real negative poles λ_i. The selection of real negative roots stems from the objective to identify a stable system and avoid damped resonant terms since we require a monotonic trend to the target, without overshoots, as it comes to be relevant for joints limits avoidance; in addition, oscillating components certainly introduce jerkiness in the response.

The coefficients C_i can be determined by imposing null initial conditions for the position, the velocity and the acceleration. Other monotonic solutions different from (4) exist for a generic third order equation depending on the multiplicity of the roots of the characteristic polynomial, thus we repeated our treatment exhaustively over all the possible cases, finding out that the most suitable structure of the solution simplified to having just one root λ of multiplicity 3. Therefore, defined $M(t)$ the measure of the squared jerk $\dddot{x}^2(t)$ accumulated up to time *t* as:

$$M(t)=\int_0^t \dddot{x}^2(\tau)\,d\tau, \quad (5)$$

we solved the following minimization problem for the case *T*=1 and x_d =1 without loss of generality:

$$\lambda^*=\arg\min_{\lambda\in\mathbb{R}}\left(M(\infty)\right) \quad \text{s.t.} \begin{cases}\lambda<0\\ x(1)\geq 1-\varepsilon\end{cases}. \quad (6)$$

The extension of $M(T)$ into $M(\infty)$ yields a simplification in the numeric evaluation of the cost and can be performed since $x(t)$ is a monotonic function. Furthermore, the second constraint in (6) forces the solution to reach the steady-state with a "rate" specified by the parameter ε. Without this lower bound on $x(1)$, any possible monotonically increasing function would be permitted, including functions with very slow time constants. In other words, by setting the parameter ε we are able to tune the final execution time. It is worth pointing out that the cost $M(\infty)$ can be easily resolved as function of the root λ and more importantly its gradient has a close form that enables to carry out the minimization with reliable gradient-based methods (such as the interior point algorithm).

Fig. 5 compares the trajectories (position, velocity and the measure $M(t)$) of the ideal minimum-jerk model against those obtained with the time-invariant system derived with our approach by selecting $\varepsilon=0.1$ (we retrieved $\lambda^*=-5.322$). As expected the LTI system approximates the ideal minimum-jerk position trajectory, having in particular a slightly faster onset followed by slower convergence to the steady state. At the same time, however, it provides a very good compromise between smoothness and simplicity of implementation.

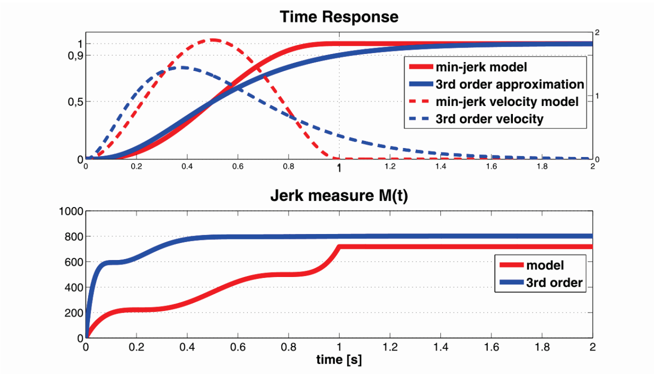

Fig. 5. Comparison between the responses of the 3rd order dynamical time-invariant system found through the minimization (blue) and the responses of the minimum-jerk model (red).

Once the root λ^* is known, it is possible to compute the elements of the dynamic matrix *A* and input matrix *B* and write the controller in the canonical form extended to the case of generic execution time *T* (the description of these steps is out of the scope of this paper):

$$\begin{bmatrix}\dot{x}\\ \ddot{x}\\ \dddot{x}\end{bmatrix}=\underbrace{\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ \frac{a(\lambda)}{T^3} & \frac{b(\lambda)}{T^2} & \frac{c(\lambda)}{T}\end{bmatrix}}_{A}\begin{bmatrix}x\\ \dot{x}\\ \ddot{x}\end{bmatrix}+\underbrace{\begin{bmatrix}0\\ 0\\ -\frac{a(\lambda)}{T^3}\end{bmatrix}}_{B}x_d. \quad (7)$$

The system response in $t=T$ is equal to the 90% of the steady-state value as enforced by the constraint, and the transient can be considered extinguished for $t\geq 1.5\cdot T$.

The first important result achieved by this controller is visible in Fig. 6: it is clear that minimum-jerk controllers can provide, especially for fast movements, smooth velocity profiles that are more similar to the desired human-like prototypes if compared to the profiles generated by the VITE model.

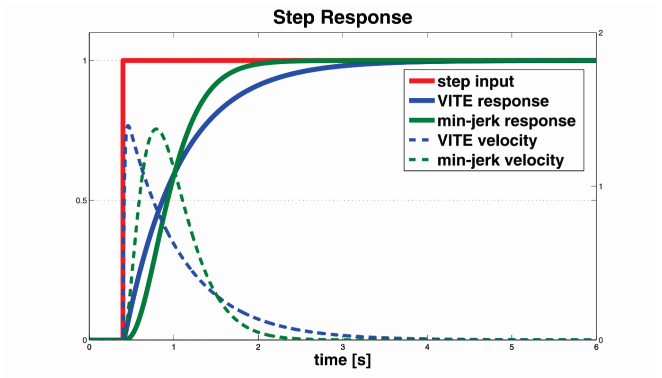

Fig. 6. Comparison between step responses of VITE's (blue) and minimum-jerk controller (green), providing the same velocity peak. The 3rd order system has a faster convergence and an (almost) bell-shaped velocity profile.

C. The Controller: implementation issues

The algorithm follows faithfully the three steps in [2] with two main exceptions as detailed hereafter.

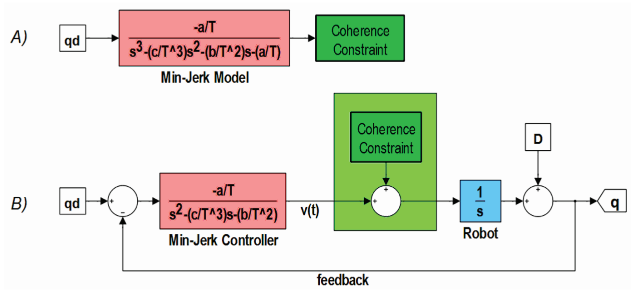

Fig. 7. *A)*: transfer function of the LTI minimum-jerk model. *B)*: an insight of the joint space minimum-jerk controller implemented in closed-loop form.

1) The need to constantly read the feedback q_{fb} motivated the authors of [2] to introduce a P controller with the purpose to keep the generation of trajectories and their execution as two separated functionalities, preventing the evolution of the feedback from interfering with the internal state of the VITE. This brought about a series of drawbacks we have analyzed in section III.B.

In order to adhere to the original diagram of Fig. 3, an alternative solution has been explored that transforms the structure of our model as represented in Fig. 7-*B*, where for sake of clearness only the joint space minimum-jerk controller is presented: from case *A* that corresponds to the state-space model in (7) we migrated to the system *B* that owns exactly the same transfer function $q(s)/q_d(s)$ written in *A*, taking now the actual error between the target and the feedback as input. The pure integrator plays the role of the mechanical plant that integrates the received command and returns the current feedback. All the remaining unmodeled dynamics and uncertainties are represented by the term *D*, whose effects are rejected by the closed-loop system. The disturbance introduced in the minimum-jerk controller by the coherence constraint is compensated similarly: as a matter of fact, the signal computed through the Lagrangian multipliers does not act like a feed-forward component, but rather perturbs the controller. To conclude, the closed-loop structure realizes exactly the scheme of the multi-referential approach and ensures that the current robot's position is correctly fed back in the system.

2) The modulation of weighting matrices that appear in the coherence reinforcement and serve for the handling of *joints limits avoidance* (see section 3.6 of [2]) is achieved by imposing a cosine shaping relation (see Fig. 8). Nonetheless, to better exploit the whole arm workspace it is advisable to assign high priority to the Cartesian controller in a portion of the joint space that is as large as possible. To this end we adopted a different weighting policy made of a flat region connected with hyperbolic tangent functions whose decay rate is much steeper than the original cosine law, as illustrated in Fig. 8, showing out its benefits in the execution of quasi-straight Cartesian trajectories for point-to-point motion as demonstrated by experiments.

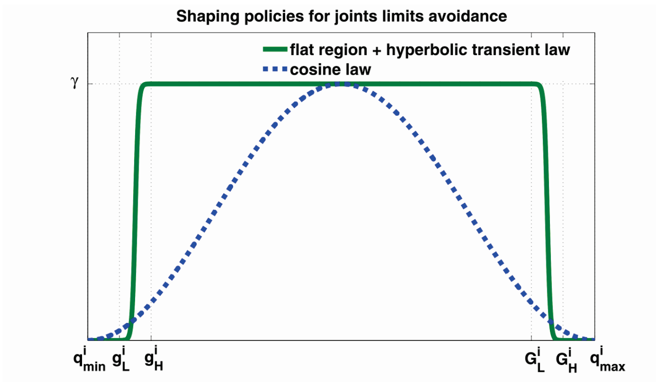

Fig. 8. The implemented shaping policy for the joints limits avoidance (green) and the original cosine law (blue) used in [2].

IV. Experimental Results

According to the RobotCub open-source philosophy the whole software developed by the project partners was made available to the wide community of iCub users; this facilitates the collaboration and promotes the development of new algorithms. As result, a collection of libraries and modules targeting different fields (vision processing, motor development, machine learning, etc) has been circulating among partners of the RobotCub Consortium who can easily reuse code for their research activities without having to be concerned with the implementation details. In this respect it has been almost immediate and much valuable comparing on the same shared platform the performance of our Cartesian controller[2] with the VITE-based system[3] whose modules can

[2] The stand-alone application of Minimum-Jerk Cartesian controller used to obtain the presented experimental results is named *iKinArmCtrl* and relies on the *iKin* library, a general purpose tool for forward and inverse

be downloaded from public repositories. Additionally, we included in the assessment a further controller as an example of a more conventional strategy making use of the typical Damped Least-Squares (DLS) rule [15] coupled with a secondary task that comprises the joints angles limits by means of the gradient projection method [16]. This solution employs the third-party package *Orocos*[4], a tool for robot control that implements the DLS approach (used by some partners of the RobotCub Consortium) and whose public availability and compliance with real-time constraints justified its adoption as one of the reference controllers.

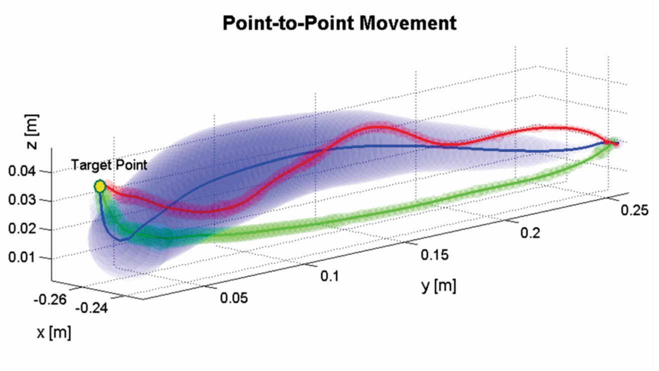

Fig. 9. Point-to-point Cartesian trajectories executed by the three controllers: the VITE-based method produces on average the blue line, the minimum-jerk controller result is in green, the DLS system using Orocos in red. Bands containing all the measured paths within a confidential interval of 95% are drawn in corresponding colors. Controllers settings are: T=2.0 s for the minimum-jerk system, α=0.008, β=0.002, K_P=3 for the VITE, and μ=10^{-5} for the damping factor of the DLS algorithm.

In the first experiment we put to the test the three selected schemes in a point-to-point motion task wherein the iCub arm was actuated in 7-DOF mode and whose end-effector was controlled both in position and orientation. It came out that paths produced by our controller and the DLS-based system are well restricted in narrow tubes of confidence intervals and are quite repeatable; conversely the VITE is affected by a much higher variability.

Fig. 9 highlights what reported for a set of 10 trials of a typical reaching task where the right hand is moved from the rest posture towards a location in front of the iCub waist with the palm turned downward; Tab. 1 summarizes the measured in-target errors for the three cases: all the controllers behave satisfactory, but the DLS achieves lower errors because operates continuously on the current distance from the target $\mathbf{x}_d$, being virtually capable of canceling it at infinite time. On the contrary, strategies based on the interaction with an external solver bind the controller module to close the loop on an approximation $\tilde{\mathbf{x}}_d$ of the real target that is determined by the optimization tolerances as in (1).

TABLE I

Controller	Position Error	Orientation Error	Mean Radius of Trajectory Band
VITE-based	$1.3 \pm 1.4 \cdot 10^{-3}$ mm	0.041 ± 0.05 rad	10 ± 10.8 mm
Min-Jerk-based	$3.0 \pm 1.4 \cdot 10^{-3}$ mm	0.048 ± 0.008 rad	2.5 ± 1.5 mm
DLS-based	$1.3 \pm 1.4 \cdot 10^{-3}$ mm	0.016 ± 0.028 rad	2.0 ± 1.36 mm

Tab. 1. Mean errors along with the confidence levels at 95% computed when the target is attained. An average measure of the variability of executed path is also given for the three controllers.

Regarding the analysis of human-likeness, the new proposed Cartesian controller outperforms both the traditional and the VITE-based solution thanks to the regulator design – so near to the ideal minimum-jerk model – and also as consequence of the wider working region where the task space module can function due to the replacement of the shaping policy. It is indeed clear from Fig. 9 how the trajectory commanded by the minimum-jerk controller (green line) approaches much more a quasi-straight path whereas the red and the blue lines oscillate before reaching the target. This important aspect becomes evident also once the velocity of the end-effector in the operational space is drawn as shown in Fig. 10: the velocity profiles have been computed in post-processing from the indirect acquisition of the end-effector coordinates by reading the joints encoders values (i.e. $x(t) = K(q(t))$ and then have been filtered to remove the high-frequency components (with a cutting frequency of 2.5 Hz). The superposition with the curve of the ideal minimum-jerk prototype (sketched black line), identified by knowing the starting and the ending points of the motion, underlines the good level of similarity of our implementation (green line) and, at the same time, the discrepancy of the other two methods which show a very sharp onset and a remarkable asymmetry of the response.

Tab. 2 sums up the jerk measures computed in the Cartesian space: a factor of 43.8% is gained with respect to the VITE system and even a factor of 69% is achieved against the DLS.

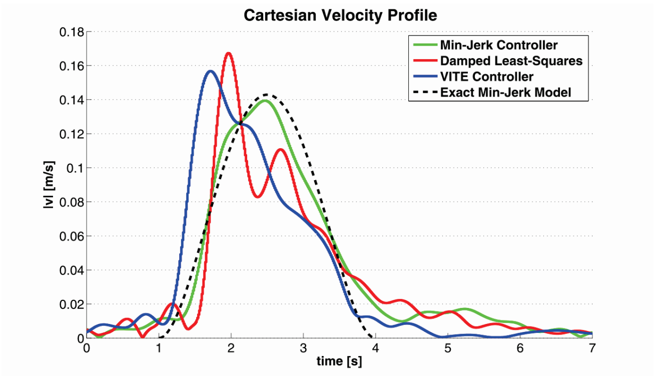

Fig. 10. Magnitude of the end-effector velocity in the operational space: blue for VITE, red for DLS and green for minimum-jerk controller. The point-to-point task begins at t=1s.

kinematics of serial link chains. It is available from the repository: https://robotcub.svn.sourceforge.net/svnroot/robotcub/trunk/iCub.

[3] A version of the M. Hersch's code that accepts target points expressed in the iCub standard reference frame can be downloaded from the repository: https://robotcub.svn.sourceforge.net/svnroot/robotcub/trunk/papers/pattacini2010.

[4] The kinematic component of Orocos project is reachable here: http://www.orocos.org/kdl.

The second evaluation was concerned with the dynamical characteristics of the DLS and minimum-jerk controllers. In particular, we verified their capabilities to track in position a quite fast reference trajectory given in the operational space while keeping the orientation of the end-effector constant; unfortunately, we did not manage to test the VITE algorithm as we experienced that the implemented CCD solver was not fast enough to run in real-time.

TABLE II

Controller	$\int_1^{+\infty}\left(\frac{d^2 v_{\text{cart}}}{dt^2}\right)^2 d\tau \Big/ \int_1^{+\infty}\left(\frac{d^2 v_{\text{exact model}}}{dt^2}\right)^2 d\tau$
VITE-based	71.86
Min-Jerk-based	40.33
DLS-based	131.06

Tab. 2. Relative measures of the jerk in the operational space given as the ratio between the integral of the squared second derivative of the Cartesian velocity for the three controllers and the same quantity computed for the exact minimum-jerk model.

The target passed to the controllers evolved along a lemniscate-shaped trajectory (Fig. 11), completing one cycle with two different time periods: T_P in {30, 15} seconds. In the first experiment (T_P=30 s), the minimum-jerk Controller ran with the parameter T set to 2.0 s and accomplished the task considerably well; the DLS method deviated somehow from the reference and did not perform with the same accuracy. When the target moved faster (T_P=15 s), the minimum-jerk Controller still behaved better, in the sense that the gap between the executed curve and the reference was lower compared to the DLS case, as we reduced the parameter T to 1.5 s in order to get a quicker response; on the other side, the DLS reactivity remained unchanged lacking of an analogous built-in tuning.

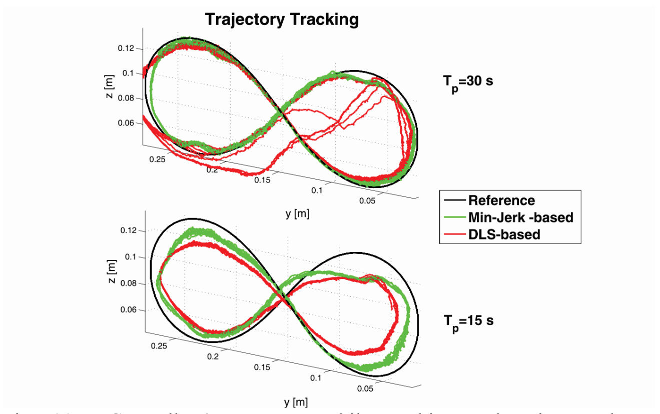

Fig. 11. Controllers' responses while tracking a lemniscate shape: minimum-jerk controller in green, DLS in red. The resulting trajectories of 10 trials are shown for the two time periods T_P.

Notably, it is crucial to mention for this kind of test that IpOpt is able to comply with the stringent real-time constraints and eventually allows to close the loop of Fig. 2: the average solving time of (1) for the case T_P=15 s was 17±28 ms, having the solver running at 33 Hz on a multi-core Intel (R) Xeon with 2.27 GHz of clock frequency.

V. CONCLUSION

We designed a novel and general purpose Cartesian controller capable of moving the robot end-effector in the operational space in a way most similar to the behavior that humans express during reaching tasks, performing motions that follow quasi-straight trajectories with bell-shaped velocity profiles. By inheriting the concept of multi-referential approach from a pre-existing VITE-based model, the controller's scheme has been improved through an optimization process that enabled to achieve better results in terms of similarity to human-like movements with respect to the VITE system and traditional control strategies. This was demonstrated by carrying out an experimental assessment of the different techniques on the same robotic platform.

REFERENCES

[1] L. Sciavicco, B. Siciliano, (2005). *Modelling and Control of Robot Manipulators* (Second Edition).

[2] M. Hersch, A.G. Billard, "Reaching with multi-referential dynamical systems," *Autonomous Robots*, Springer-Verlag, pp. 71–83, 2008.

[3] A. Wätcher, L.T. Biegler, "On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming," *Mathematical Programming* 106 (1): pp. 25–57, 2006.

[4] D. Bullock, S. Grossberg, "Neural dynamics of planned arm movements: Emergent invariants and speed-accuracy properties during trajectory formation," *Psychological Review* 95 (1): pp. 49–90, 1988.

[5] W. Abend, E. Bizzi, P. Morasso, "Human arm trajectory formation," *Brain* 105, pp. 331–348,1982.

[6] S. Chiaverini, B. Siciliano, O. Egeland, "Review of the Damped Least-Squares Inverse Kinematics with Experiments on an Industrial Robot Manipulator", *IEEE Transactions on Control Systems Technology* 2 (2), 1994.

[7] A. Liegeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Transactions on Systems, Man, and Cybernetics*, 7 (12): pp. 868–817, 1977.

[8] G. Metta, D. Vernon, L. Natale, F. Nori, G. Sandini, "The iCub humanoid robot: an open platform for research in embodied cognition," *IEEE Workshop on Performance Metrics for Intelligent Systems*, Washington, USA, 2008.

[9] T. Flash, N. Hogan, "The coordination of arm movements: an experimentally confirmed mathematical model," *Neuroscience* 5, pp. 1688–1703, 1985.

[10] B. Hoff, M.A. Arbib, "A model of the effects of speed, accuracy and perturbation on visually guided reaching," In: Control of arm movement in space: neurophysiological and computational approaches (R. Caminiti, P.B. Johnson, Y. Burnod, eds), pp. 285–306.

[11] R. Shadmehr, S.P. Wise, 2005: Supplementary documents for "Computational Neurobiology of Reaching and Pointing". Available: http://www.shadmehrlab.org/book/minimumjerk.pdf

[12] L.C. Wang, C. Chen, "A combined optimization method for solving the inverse kinematics problems of mechanical manipulators," *IEEE Transactions on Robotics and Automation* 7 (4): pp. 489–499, 1991.

[13] A. Parmiggiani, M. Randazzo, L. Natale, G. Metta, G. Sandini, "Joint torque sensing for the upper-body of the iCub humanoid robot," *IEEE International Conference on Humanoid Robots*, 2009.

[14] A.J. Ijspeert, J. Nakanishi, S. Schaal, "Movement Imitation with Nonlinear Dynamical Systems in Humanoid Robots," *IEEE International Conference on Robotics and Automation*, 2002.

[15] A.S. Deo, I.D. Walker, "Robot Subtask Performance with Singularity Robustness using Optimal Damped Least-Squares," *IEEE International Conference on Robotics and Automation*, 1992.

[16] H.-Y. Lee, B.-J. Yi, Y. Choi, "A Realistic Joint Limit Algorithm for Kinematically Redundant Manipulators," *IEEE International Conference on Control, Automation and Systems*, 2007.