A scalable Joint-Space Controller for Musculoskeletal Robots with Spherical Joints

Michael Jäntsch, Christian Schmaler, Steffen Wittmeier, Konstantinos Dalamagkidis and Alois Knoll

***Abstract*—In the long history of robotics research, the most prominent problem has always been, to develop robots that can safely operate in human-centered environments. One way towards the goal of a safe, and human-friendly robot, is to incorporate more and more of the flexibility that can be found in humans, by mimicking the internal mechanisms. In this work we propose a scalable joint-space control scheme based on computed torque control for an anthropomimetic robot. To achieve this, the dynamic system model of the robot is decomposed into hierarchical subsystems, using scalable modeling algorithms where possible. Machine learning techniques were employed to tackle the problem of muscle force to joint torque mapping.**

The developed control scheme has been evaluated using the highly refined simulation of an anthropomimetic robot arm featuring 11 muscles, a revolute elbow joint and a spherical shoulder joint. We show trajectory tracking based on a low-level muscle and a high-level joint control scheme, taking into account the coupling between the joints due to inertial reactions and bi-articular muscles.

***Keywords*—anthropomimetic robot, robot control, distributed control, biomechanics, biorobotics**

I. Introduction

Major progress in robotics turns today's humanoid robots into ever safer, more robust, and more agile agents by the moment. However, it is still a long way until robots can safely operate in unstructured environments. Especially in the area of service robotics, the need arises for robots to work flexibly in a human-centered environment. One way towards this goal is to incorporate more and more of the mechanisms that can be found in humans for robots.

In this work we would like to propose a whole body control strategy for an anthropomimetic robot [1], based on the hierarchical control architecture described in [2]. Anthropomimetic robots are highly bio-inspired and mimic not only the general appearance of the human body, but also the mechanisms of the musculoskeletal system, like bones, joints, and muscles. This calls for a completely new control strategy, because the dynamics are very different from those in standard humanoid robots.

Michael Jäntsch, Christian Schmaler, Steffen Wittmeier, Konstantinos Dalamagkidis and Alois Knoll are with the chair for Robotics and Embedded Systems, Department of Informatics, Technische Universität München, Munich, Germany

michael.jaentsch@in.tum.de
schmaler@in.tum.de
steffen.wittmeier@in.tum.de
dalamagk@in.tum.de
knoll@in.tum.de

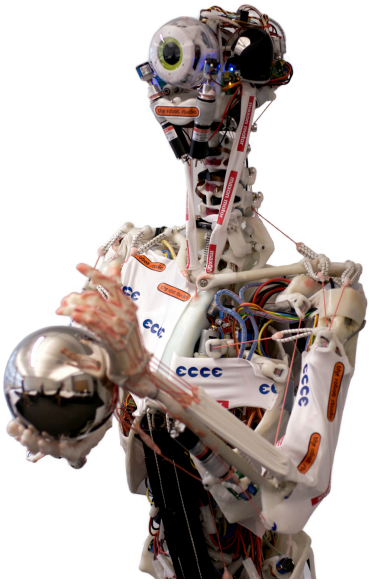

Fig. 1. ECCEROBOT Design Study (EDS). This is the newest platform in the ECCEROBOT family [3]. In this type of robot not only the appearance of the human body is mimicked but also the inner structures, like bones, joints, muscles, and tendons.

In [4, 5], de Sapio et. al. use a highly generic approach to control a human shoulder complex in simulation, based on the work of Thelen et. al. [6]. Since the mechanisms in the human body and in the anthropomimetic robot are comparable, a similar approach is expected to be suitable for controlling our robot. However, these control approaches were only used in the simulation of a human body, without even considering the implementation in a robotic system. Other control schemes for musculoskeletal robot systems, like [7–9], show a very detailed analysis of a specific joint structure and propose a control law based on it. However, [7, 10] do not cover bi-articular muscles, and none of the above investigate the control of ball-and-socket (or spherical) joints. Therefore, a generic approach for musculoskeletal robots, like it was proposed by de Sapio [5] for the simulation of a human body, needs to be found.

II. The ECCEROBOT

The first anthropomimetic robot CRONOS [1], whose technology is being used in this work, is a robot which tries to mimic the human skeleton, as well as the muscular system. The robot bones were made by hand from a thermoplastic which can be hand molded at a temperature of 60°C. The

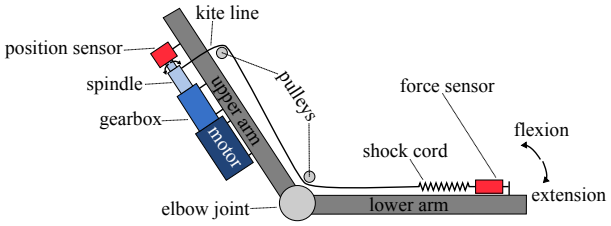

Fig. 2. Actuation principle. An artificial muscle (AM) mimics the human muscle including its elasticity. It consists of a motor that winds kite line on a spindle and hence exerts a force on the robot's bone.

artificial muscles (AMs) consist of a DC motor, kite line, and shock cord. In this type of electric actuator the motor winds the kite line on the attached spindle and hence either innervates or relaxes the AM, depending on the direction of motor rotation. Therefore, force can only be exerted on the attachment points in one direction, i.e. a muscle can only pull, not push. The shock cord adds the flexibility that is also present in a biological muscle (see Fig. 2). The production techniques, as well as the materials used, facilitate the creation of a robot of this complexity. However the impossibility of dismantling and reassembling the robot, as well as lack of a CAD design, pose major challenges in system modeling [11].

Not only the type of actuation is as close to its biological counter part as possible with today's technology, but also the attachment points of the AM. Of course it is (currently) impossible to duplicate all of the well over 250 muscles in the human body [12]. The muscles that were chosen to be incorporated in the current prototype (see Fig. 1) are the ones responsible for larger scale movements, omitting the ones used for fine grained dexterous movements, e.g. in the hands. For a full description of the ECCEROBOT, please refer to [3, 13].

III. Modeling Musculoskeletal Robots

In [7, 8] a model of a tendon-driven robot is obtained directly for the full state space. This leads to a very complex and also non-linear model for which a controller can be found, provided the system is small enough. However, the process of developing these models and corresponding controllers shows that this is highly problematic for larger systems.

To cope with more complex systems, it is possible to decompose them into a hierarchy of simpler subsystems for which separate control methods can be derived. An anthropomimetic robot can be divided in three subsystems. First a model of the comparably stiff robot components—the skeleton—can be obtained like for any conventional robot (Section III-A), second the AMs are modeled (Section III-B), and last a mapping between the two needs to be found (Section III-C).

A. Skeleton Model

For conventional robots the equation of motion can be expressed in one of two canonical forms [14]. In joint space this can be written as follows,

$$\tau = H(q)\ddot{q} + C(q,\dot{q}) + \tau_G(q) \tag{1}$$

giving a relationship between the joint torques τ and the generalized joint coordinates q, $\dot{q}$, and $\ddot{q}$.

Generally, this equation holds for all joint types, however it is mostly used for single degree of freedom (DoF) joints. Like the human body, the ECCEROBOT has revolute, as well as spherical joints that allow rotation around three axis. Rotation in three dimensional space can be represented with three angles in many different ways (e.g. ZYX-Euler-Angles, XYZ fixed Angles). However, all of these come with the problem of possible singularities. Despite making controller design more complicated, the preferred representation of orientation in 3 dimensions, are unit quaternions [15]. Here, rotation is parameterized with four instead of three parameters $Q = [\eta, \epsilon_1, \epsilon_2, \epsilon_3]^T$, while the quaternion norm is confined to $||Q|| = 1$. It consists of a scalar real part η and a 3×1 imaginary part vector ϵ. While obviously increasing the dimensionality of the description, the issue of singularity is eliminated. Furthermore, quaternions can be used for efficiently generating rotational movements at constant speed around a constant rotation axis (e.g. *SLERP* [16]). This poses the additional challenge of representing the canonical equation of motion (1) with quaternions. This can be achieved by replacing the pose vector q by α, containing the quaternion representation of all spherical joints, as well as the angular representation of all revolute joints,

$$\tau = H(\alpha)\ddot{q} + C(\alpha,\dot{q}) + \tau_G(\alpha) \tag{2}$$

while the dimensionality of α is therefore higher than of q. However, a relationship between the derivative of α and the rotational velocities $\dot{q}$ needs to be found. The derivative of a quaternion $\dot{Q}$ as a function of the corresponding rotational velocities ω can be shown to be as follows [14].

$$\dot{Q} = \frac{1}{2}U(Q)\cdot\omega \quad \text{, with} \quad U(Q) = \begin{bmatrix} -\epsilon_1 & -\epsilon_2 & -\epsilon_3 \\ \eta & \epsilon_3 & -\epsilon_2 \\ -\epsilon_3 & \eta & \epsilon_1 \\ \epsilon_2 & -\epsilon_1 & \eta \end{bmatrix} \tag{3}$$

Therefore we define a matrix $A(\alpha)$, as a diagonal matrix, in which each revolute joint is represented by a 1 and each spherical joint is represented by $\frac{1}{2}U(Q)$.

$$\dot{\alpha} = A(\alpha)\cdot\dot{q} \tag{4}$$

A rigid body model of the skeleton in the canonical form (2) can be found using standard techniques like the Newton-Euler Algorithm or more efficiently using the Composite Rigid Body Algorithm [17]. Those algorithms make use of an efficient representation of both kinematic information, like the coordinate transforms between links and joints, and dynamic information, like masses and inertial parameters, to build a model in the shape of (2).

B. Muscle Model

The AMs of the anthropomimetic robot consist of a DC motor, kite line and shock cord (see Fig. 2). A model of an AM can be obtained by combining the standard DC motor model with a model of the gearbox and a linear spring ($F = k\cdot\Delta x$).

The resulting state space model of a muscle can be shown to be

$$\frac{d}{dt}\begin{bmatrix} i_A \\ \omega \\ F \end{bmatrix} = \begin{bmatrix} -\frac{R_A}{L_A} & -\frac{c_e \Phi_F}{L_A} & 0 \\ \frac{c_\tau \Phi_F}{J} & -\frac{\nu}{J} & -\frac{r}{g_r \eta_G J} \\ 0 & \frac{rk}{g_r} & 0 \end{bmatrix} \cdot \begin{bmatrix} i_A \\ \omega \\ F \end{bmatrix} + \begin{bmatrix} \frac{1}{L_A} \\ 0 \\ 0 \end{bmatrix} \cdot v_A + \begin{bmatrix} 0 \\ 0 \\ -k \end{bmatrix} \cdot \dot{x} \tag{5}$$

while v_A, i_A, L_A and R_A are the armature voltage, current, inductance and resistance, respectively. Φ_F is the magnet flux of the stator and c_e and c_τ are the motor constants for back emf and torque. J is the combined motor and gearbox moment of inertia, g_r is the gearbox ratio and r is the radius of the spindle, winding up the nonelastic kite line. The friction in the gearbox and the motor is represented by a viscous component ν and a gearbox efficiency η_G. Finally ω is the rotational velocity of the motor output shaft and x is the linear displacement of the muscle at the anchor point of the muscle on the side of the shock cord (see Fig. 2).

C. Muscle Jacobian

A mapping between the muscle model and the skeleton model can be formulated based on the so called muscle jacobian $L(q)$ [18]. It gives a relation between the derivative of the muscle lengths l with respect to the joint angles q at a certain configuration.

$$L(q) = \frac{\delta l}{\delta q} \tag{6}$$

By using the principle of virtual work this can be transformed to a relation between the muscle forces f (the negative sign arises from the definition of a positive force when pulling) and the joint torques τ [4].

$$\tau = -L^T(q) \cdot f \tag{7}$$

The muscle jacobian can be obtained in different ways. A geometric representation of the muscle lengths subject to the joint angles can be found. By differentiating this representation with respect to the joint angles, the muscle jacobian is obtained. While this is definitely possible for the robot model that is presented in this work, one has to note that we wanted to find a possiblity where muscles that collide with the skeleton can be modeled as well.

In this work we propose to use machine learning to determine the relationship between muscle lengths and joint angles. In this case a function approximator needs to be found to generate the mapping between n skeletal angles and m muscle lengths.

$$l = f(q) \tag{8}$$

As samples can be drawn from the full space of q the function approximator needs only to interpolate between samples, without the need for extrapolation. Well suited for this kind of problem is among others a function approximator based on artificial neural networks (ANN). For this work a multilayer perceptron (MLP) network with a single hidden layer was chosen. The samples of joint angles q (inputs) and corresponding muscle lengths l (outputs) are collected prior to network training. Therefore, learning $f(q)$ poses a supervised learning problem which can be treated using the well-known back-propagation algorithm [19].

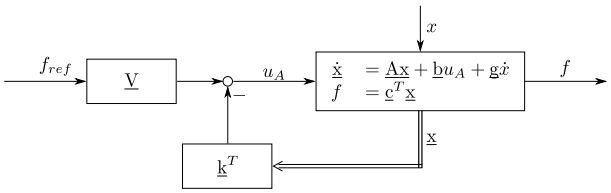

Fig. 3. Force Controller. A state space force controller k with a pre-filter V is shown for the state space system $\{A, b, c^T, g\}$ of the muscle model.

The ANN approximation of (8) can be differentiated with respect to the joint angles q, using the difference quotient to obtain the muscle jacobian.

$$L(q) = \frac{f(q + \Delta q) - f(q)}{\Delta q} \tag{9}$$

When the pose of the robot is not represented by q, but by α as in (2), the learning algorithm will produce a function $f(\alpha)$, and therefore the difference quotient produces

$$L_\alpha(\alpha) = \frac{\delta l}{\delta \alpha} \tag{10}$$

By solving (10) for δl and dividing it by δt, we get an equation where $\dot{\alpha} = \frac{\delta \alpha}{\delta t}$ can be replaced by (4).

$$\frac{\delta l}{\delta t} = L_\alpha(\alpha)\frac{\delta \alpha}{\delta t} = L_\alpha(\alpha) \cdot A(\alpha)\frac{\delta q}{\delta t} \tag{11}$$

Therefore the muscle jacobian $L(q)$ can be written as.

$$L(q) = \frac{\delta l}{\delta q} = L_\alpha(\alpha) \cdot A(\alpha) \tag{12}$$

The muscle jacobian which is hereby obtained provides the missing mapping between the rigid body chain (2) and the AMs (5).

IV. Control

In the following, a hierarchical control structure (cascade) is developed to control the full robot. In a cascade, controllers for the subsystems can be developed independently, whenever dynamics of the inner control loop are at least an order of magnitude faster than the dynamics of the outer control loop [20]. In the following, a controller for the faster inner system—the muscle force control—is synthesized first, and subsequently a controller for the full robot body is developed.

The control approach developed in the following section is distributable in a manner, where fast force control loops can run with a frequency of $500\,\mathrm{Hz}$ on distributed nodes, and the whole body control algorithm runs with a much slower frequency on the central computer. The implementation of this control architecture has been described in detail in [2].

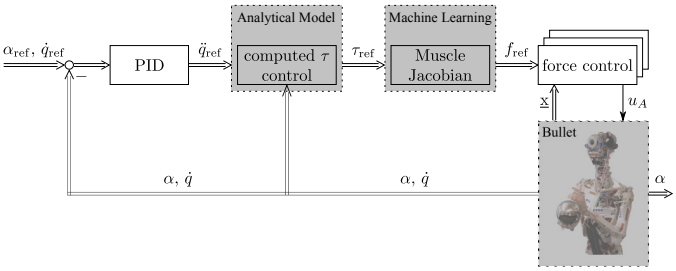

Fig. 4. Whole Body Control. The control scheme uses an analytical model of the skeleton, along with the learned muscle jacobian to calculate reference forces for the individual muscle force controllers (see Fig. 3)

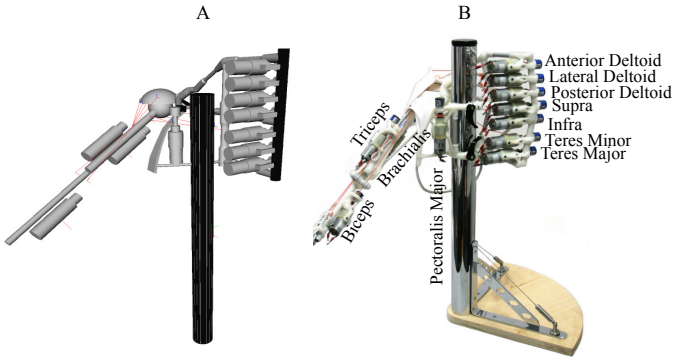

Fig. 5. Anthropomimetic Robot Arm. Fig. A depicts the simulation model rendered in CALIPER [24] of the anthropomimetic robot arm shown in Fig. B with 11 AM and 4 skeletal DoF which was manufactured by the ECCEROBOT consortium.

A. Force Control

For the synthesis of the force control for a single muscle it has to be noted that an anthropomimetic robot is underactuated, as there are less actuation variables than system states. For this reason the limb movements are assumed to be unknown disturbances while designing the force control. In the following the muscle model derived in Section III-B is used to obtain a force control algorithm.

Among P, PD, and state space control, the latter was chosen based on its superior performance at a discrete control frequency of $500\,\text{Hz}$. Hence the control law was synthesized using Ackermann's formula [21]. This can be done directly in the discrete space domain and hence taking the control frequency into account at design time. Ackerman's formula can be used to calculate state space gains k (see Fig. 3) by moving the closed loop poles of a discrete state space model to the desired values p [20]. A discrete state space model can be obtained easily from the continuous state space model in (5).

State space control frequently suffers from a steady state offset as there is no I element. Therefore, a pre-filter V (see also Fig. 3) is needed for compensation, which can be designed using the state space system $\{A, b, c^T\}$ to determine V by establishing the closed-loop behavior of the system to have a zero steady state error [20].

$$\begin{bmatrix} M_x \\ M_u \end{bmatrix} = \begin{bmatrix} A & b \\ c^T & 0 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ I \end{bmatrix} \tag{13}$$

$$V = M_u + k \cdot M_x \tag{14}$$

B. Whole body control

For standard robotic systems there are various control methods based on the canonical form of the skeleton model. Here, the method of computed torque control is used [14]. It utilizes (2) to calculate the joint torques τ necessary to produce desired joint accelerations $\ddot{q}_{\text{ref}}$, given the system states q and $\dot{q}$. In case of a perfect system model the applied joint torques will always lead to the desired joint accelerations.

The reference joint acceleration can be obtained by any control law. In this case we chose to use a PID controller to get asymptotic stability and to reduce the steady state offset.

$$\ddot{q}_{\text{ref}} = P \cdot \Delta q + D \cdot \Delta \dot{q} + I \cdot \sum_{i=0}^{i=\frac{t}{\Delta t}} \Delta q_i \tag{15}$$

Obtaining Δq for revolute or linear joints is as simple as subtracting the two vectors q and q_{ref}. However, for quaternions Q and Q_{ref}, the corresponding ΔQ is not the difference, but the rotation between the two, which is defined as follows.

$$\Delta Q = Q_{\text{ref}} * Q^{-1} \tag{16}$$

Since the zero rotation in quaternions is represented as $\Delta Q = [1 \quad 0 \quad 0 \quad 0]$, the error function Δq can written as follows [22],

$$\Delta q = \eta \epsilon_{\text{ref}} - \eta_{\text{ref}} \epsilon - S(\epsilon_{\text{ref}}) \epsilon \tag{17}$$

where $S(\cdot)$ represents the skew-symmetric operator.

Solving (7) for the muscle forces f is underdetermined, as there are more muscles than DoF in the joints. The need for additional constraints arises. This problem can be treated by formulating a quadratic optimization problem [23]. Here the objective function is the square of the euclidean distance between the forces, which is minimized, subject to two constraints. First, the forces are to apply a certain reference joint torque, and second, the muscle forces have a lower bound (muscles can only pull, not push). By minimizing the muscle forces, using these two constraints, it is guaranteed that the resulting forces (a) lead to the desired behavior and (b) the internal forces of the system are kept at a minimum.

$$\min_f ||f||^2 \tag{18}$$

$$\text{subject to} \begin{cases} -L^T(q) \cdot f = \tau \\ f > f_{min} \end{cases}$$

The forces that are obtained in this manner are then used by the force controller introduced in Section IV-A.

V. Results

The results in the following section were obtained in simulation, with the model of an anthropomimetic robot arm with 11 AM, a revolute elbow joint and a spherical shoulder joint (see Fig. 5). However, the control scheme is kept generic enough to be easily applied to larger systems.

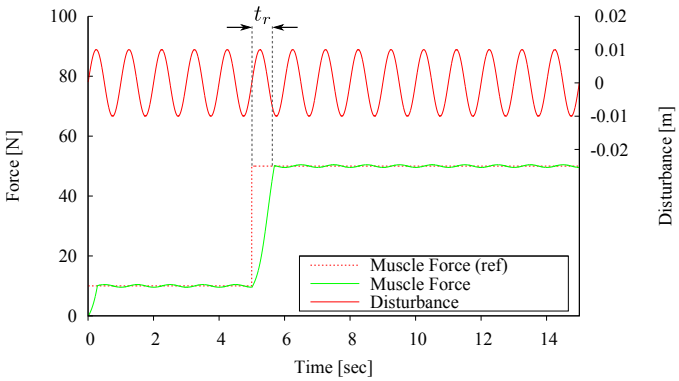

Fig. 6. Force Control Performance. The simulated muscle force and its reference is plotted, while a sinusoidal disturbance is applied with a frequency of 1 Hz and an amplitude of 0.01 m. (Using Hooke's law, this can be estimated to be 5 N.) The amplitude of the control variable is damped to 0.45 N. The step of the reference force of 40 N leads to a rise time $t_r \approx 630$ ms.

The simulation is based on CALIPER [24], a generic simulation framework based on the open source physics engine Bullet[1]. This simulator was specifically developed to reflect the robotic system introduced in Section II. Hence, a robotic skeleton with different joint types, including static as well as dynamic (Coulomb and viscous) friction can be simulated [11]. Furthermore it allows for a detailed simulation of the muscular system, including a refined version of the motor dynamics specified in Section III-B, additionally featuring Coulomb friction. Furthermore, not only the robot itself but also the control architecture as described in [2] is modeled, so the force control loops are executed asynchronously both to the simulation and the high-level control, as they would in a distributed system [11].

A. Force Control

The performance of the state space force control, developed in Section IV-A, was tested outside of the full robot set up. For this purpose the force control, as well as a model of the muscle setup was simulated in MATLAB/Simulink with different reference, as well as disturbance signals.

Fig. 6 shows the controller performance, while being disturbed by a sine wave. The system dynamics and seem rather slow for a robotic system, but one has to bear in mind that due to the flexibility of the muscle, the motor needs to actively drive a certain distance to apply the new reference force.

B. Muscle Jacobian

The muscle jacobian is obtained using the method described in Section III-C, while the major challenge is to be able to cover the full joint space. Therefore, all samples used to approximate the function $l = f(q)$ are obtained by two different methods: (i) the simulated joints are moved to different joint angles and (ii) random movements are performed through motor babbling. These two processes lead to an overall number of 103 055 data points consisting of joint angles and corresponding muscle lengths.

[1]see www.bulletphysics.com

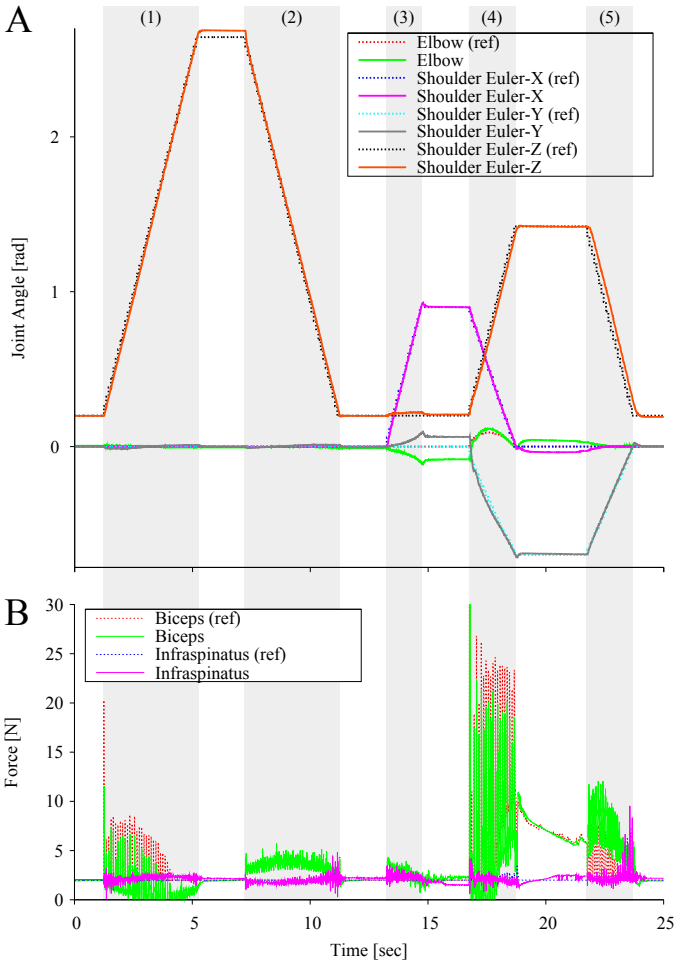

Fig. 7. Trajectory Tracking. Joint angles (Fig. A) and Forces of two muscles (Fig. B) are plotted. First, only the elbow is moved (1-2), keeping the shoulder fixed, second a shoulder adduction (3) is performed, and finally a composite movement affecting all joint angles at the same time is performed (4-5). (Please note that XYZ-Euler angles are used for illustration only.)

Best results were obtained, when the ANN was divided into simpler learning subtasks, relieving it of the task of identifying the structure of the robot. Therefore, muscles that span specific joints are represented by a network that takes only the corresponding joint angles as inputs. In the anthropomimetic robot arm that is used in this work, there is only one bi-articular muscle. This muscle (the Biceps) is therefore represented by a separate neural network, taking the quaternion of the shoulder joint and the angle of the elbow as an input. Muscles that only span the elbow joint (Triceps and Brachialis) are incorporated in a second network. The rest of the muscles (see Fig. 5) that only affect the shoulder are covered by a third network. The network sizes were determined experimentally and were chosen to have 20, 5 and 30 hidden units for the biceps, the elbow and the shoulder network, respectively.

C. Trajectory Control

The whole-body control scheme covered in Section IV-B is tested for stability with the learned muscle jacobian, by using it to follow given trajectories. We consider movements

of all joints, including combined movements (see Fig. 7-A). The maximum error of each of the angles during the whole trajectory is $[0.1121 \quad 0.0596 \quad 0.1142]^T$ for the shoulder joint and 0.1475 for the elbow joint angle, while the RMSD[2] evaluates to $[0.0302 \quad 0.0135 \quad 0.0230]^T$ and 0.0399, respectively.

Fig. 7-B depicts the forces resulting for the same trajectory. The reference forces obtained by solving the optimization problem (see Section IV-B) show high frequency changes, during the transition between poses, but the force controller is able to follow. During the steady state phases between the movements the reference forces move to a constant level. A major concern in tendon driven robotics is that tendons should never be allowed to go slack. While the control approach ensures that commanded forces are always greater than the minimum force, it is observed that forces do go to zero (see Fig. 7-B), which implies that tendons go slack for short periods of time. However, the maximum time a tendon force has a force of zero is $84.3\,\mathrm{ms}$ in this experiment. This time is too short for tendons to go slack enough to get tangled or jerk the arm in a manner that would be problematic for the control performance.

A video, showing the simulated robot arm performing this trajectory, is available online [13].

VI. Conclusions and Future Works

A. Conclusions

In this work we developed a scalable controller for musculoskeletal robots that is based on standard techniques from the field of robotics, while the complex relationship between muscles and the skeleton is automatically acquired. It has been shown before that the use of the composite rigid body algorithm scales well also to very large robotic systems in the order of 50 DoF and more [25]. This and the usage of machine learning techniques for obtaining the muscle jacobian makes it possible to control large robotic systems. Simulation results proved that the proposed control scheme performs well for a musculoskeletal system with bi-articular muscles and spherical joints.

B. Future Work

In the future we would like to verify the functionality of the proposed control scheme in robotic systems with more DoF, like the full upper-torso, developed in the ECCEROBOT project. Naturally, this will not be limited to simulated systems, as functional models become available. By doing this it will also become necessary to study the effect of muscles that collide with the robot's skeleton. Finally we would like to extend this scheme to handle not only joint space but also operational space control.

VII. Acknowledgments

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 - Challenge 2 - Cognitive Systems, Interaction, Robotics - under grant agreement no. 231864 - ECCEROBOT

[2]root mean square deviation

References

[1] O. Holland and R. Knight, "The Anthropomimetic Principle," in *Adaptation in Artificial and Biological Systems*, 2006.

[2] M. Jäntsch, S. Wittmeier, and A. Knoll, "Distributed control for an anthropomimetic robot," in *Proc. IEEE/RSJ Int Intelligent Robots and Systems (IROS) Conf*, 2010, pp. 5466–5471.

[3] H. G. Marques, M. Jäntsch, S. Wittmeier, O. Holland, C. Alessandro, A. Diamond, M. Lungarella, and R. Knight, "ECCE1: The first of a series of anthropomimetic musculoskeletal upper torsos," in *Proc. 10th IEEE-RAS Int Humanoid Robots (Humanoids) Conf*, 2010, pp. 391–396.

[4] V. De Sapio, J. Warren, O. Khatib, and S. Delp, "Simulating the task-level control of human motion: a methodology and framework for implementation," *The Visual Computer*, vol. 21, no. 5, pp. 289–302, 2005.

[5] V. De Sapio, K. Holzbaur, and O. Khatib, "The control of kinematically constrained shoulder complexes: physiological and humanoid examples," in *Proc. IEEE International Conference on Robotics and Automation ICRA 2006*, 2006, pp. 2952–2959.

[6] D. G. Thelen, F. C. Anderson, and S. L. Delp, "Generating dynamic simulations of movement using computed muscle control," *Journal of Biomechanics*, vol. 36, no. 3, pp. 321–328, 2003.

[7] S. C. Jacobsen, H. Ko, E. K. Iversen, and C. C. Davis, "Antagonistic control of a tendon driven manipulator," in *Proc. Conf. IEEE Int Robotics and Automation*, 1989, pp. 1334–1339.

[8] V. Potkonjak, B. Svetozarevic, K. Jovanovic, and O. Holland, "Biologically-inspired control of a compliant anthropomimetic robot," in *Proceedings of the 15th IASTED International Conference on Robotics and Applications*, 2010.

[9] H. Kino, S. Kikuchi, T. Yahiro, and K. Tahara, "Basic study of biarticular muscle's effect on muscular internal force control based on physiological hypotheses," in *Proc. IEEE International Conference on Robotics and Automation ICRA '09*, 2009, pp. 4195–4200.

[10] V. Potkonjak, B. Svetozarevic, and O. Holland, "Control of Compliant Anthropomimetic Robot Joint," in *Symposium on Computational Geometric Methods in Multibody System Dynamics*, 2010.

[11] S. Wittmeier, M. Jäntsch, K. Dalamagkidis, and A. Knoll, "Physics-based Modeling of an Anthropomimetic Robot," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems IROS 2011*, 2011.

[12] E. R. Kandel, J. H. Schwartz, and T. M. Jessel, *Principles of Neural Science*, 4th ed., J. Butler and H. Lebowitz, Eds. McGraw-Hill, 2000.

[13] Embodied Cognition in a Compliantly Engineered Robot (ECCEROBOT). [Online]. Available: www.eccerobot.org

[14] B. Siciliano and O. Khatib, *Springer Handbook of Robotics*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[15] B. Xian, M. S. de Queiroz, D. Dawson, and I. Walker, "Task-space tracking control of robot manipulators via quaternion feedback," vol. 20, no. 1, pp. 160–167, 2004.

[16] K. Shoemake, "Animating rotation with quaternion curves," *SIGGRAPH Comput. Graph.*, vol. 19, no. 3, pp. 245–254, Jul. 1985.

[17] R. Featherstone, "A Divide-and-Conquer Articulated-Body Algorithm for Parallel O(log(n)) Calculation of Rigid-Body Dynamics. Part 1: Basic Algorithm," *The International Journal of Robotics Research*, vol. 18, no. 9, pp. 867–875, 1999.

[18] V. M. Zatsiorsky, *Kinetics of Human Motion*. Human Kinetics Publishers, 2002.

[19] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[20] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital Control of Dynamic Systems*, T. Hyde, Ed. Prentice Hall, 1997.

[21] J. Ackermann, *Sampled-Data Control Systems*. Springer, 1985.

[22] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics - Modelling, Planning and Control*. Springer, 2009.

[23] V. De Sapio, J. Warren, and O. Khatib, *Predicting reaching postures using a kinematically constrained shoulder model*. Springer Netherlands, 2006, ch. 3, pp. 209–218.

[24] S. Wittmeier, M. Jäntsch, K. Dalamagkidis, M. Rickert, H. G. Marques, and A. Knoll, "CALIPER: A Universal Robot Simulation Framework For Tendon-Driven Robots," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems IROS 2011*, 2011.

[25] R. Featherstone, *Rigid Body Dynamics*. Springer Science+Business Media, LLC, 2008.