

Fast human–robot modelling and related dependability issues

Agostino De Santis, Bruno Siciliano

Abstract—The possibility of fast modelling of an unstructured scene for Human-Robot Interaction (HRI) is to be related to the dependability of the models which are used for robot control. A review of our recent contributions to modelling and control for fast and reliable motion control in HRI is presented in this paper. The models implemented for real-time control have to be complemented with solutions for avoiding discontinuities and propagation of sensory data uncertainty.

Index Terms—human–robot modelling, sensor-based trajectories, uncertainty

I. INTRODUCTION

The adoption of safety and dependability of a human user as evaluation criteria for physical Human-Robot Interaction (pHRI) results in a complete rethinking of many phases of robot modelling and control [1], [2]. With proper modelling for simple geometric computation and real-time multiple-point control, it is possible to choose an arbitrary control point on a robot, and controlling it on suitable trajectories for avoiding collisions [3] and/or scaring motion. Evaluations which are proper of cognitive HRI (cHRI), complementary to those introduced in a pHRI-related framework, can also be helpful for setting some variable parameters. A key point for safe interaction is the dependability of sensory information, which is the basis for computing distances and motion trajectories [4].

Planning/control approaches to scene modelling for HRI can present different strategies: very fine modelling of people and robot can be time-consuming; on the other hand, simple modelling can be conservative, but very fast and possibly integrated into a variety of already implemented control systems, such as force or impedance control for close interaction. The possibility of having a very fast modelling and control of the interaction environment is a key issue for quick motion of interacting robots, giving control to different interfaces and control approaches at the same time.

Modelling and control issues for robot manipulators for HRI discussed in this paper are part of the research carried out in the framework provided by the European project PHRIENDS [5].

Desired features for scene modelling are the speed in the computation, and the possibility of changing control

points and trajectories in real-time, within a certain level of safety. The approach to modelling introduced in [3] is based on skeletal models of interacting human, robot and environment (see Figs. 1, 2, 3). Such an approach, as recalled in the following, offers the possibility to give the possibility of controlling arbitrary points on the robot. After the modelling phase, arbitrary trajectories in the task space can be derived.

In past work [6], experimental results have been presented, related to interaction with the DLR Justin manipulator [7] compared with an industrial robot arm for human–robot cooperation. Related to the problem of collision avoidance, experiments performed for self-collision avoidance of a humanoid manipulator correspond to an ideal case where positions of the possibly colliding bodies are known almost exactly. At the same time, experiments with industrial manipulators showed that, while the technique is effective (head avoidance and goal-directed movements were fulfilled properly), a trade-off with performance is however needed (e.g., slowing down the maximum repulsion motions). In fact, the intrinsic nature of heavy industrial manipulators gives less robustness with respect to consequences of sudden motions in case of malfunctioning (which has to be related with probabilistic aspects). In addition, possible discontinuities due to the nature of modelling have been considered in [8], and the on-going work is devoted to model the impact of sensory uncertainty propagation in the adopted modelling/control approach. The approach in [8] also allows implementation in velocity control.

II. SKELETAL MODELS AND REQUIRED COMPUTATION

An approach which automatically selects a control point on a manipulator, based on sensor information and analytical computation, is based on a simplified model of a kinematic chain. In addition, such arbitrary control points should be computed fast, based on a model of the environment which leads to simple distance computation and trajectory determination. These considerations lead to the so-called “skeleton algorithm”, developed for collision avoidance applications (see [3] and bibliography therein), whose steps include: to build a proper model of the robot, namely the skeleton, useful for analytical computation; to find the control points along the skeleton, via distance computation or explicit user’s decision; to generate trajectories and corresponding joint commands for the controller.

The problem of analyzing the whole volume of the parts of a manipulator is simplified by considering a *skeleton* of the structure, and proper volumes surrounding it. With

This work was supported by the PHRIENDS Specific Targeted Research Project, funded under the 6th Framework Programme of the European Community under Contract IST-045359. The authors are solely responsible for its content. It does not represent the opinion of the European Community and the Community is not responsible for any use that might be made of the information contained therein.

A. De Santis and B. Siciliano are with PRISMA Lab, DIS, Università di Napoli Federico II, Italy, <agodesa, siciliano>@unina.it

reference to the DLR Justin manipulator, such a skeleton is reported in Fig. 1.

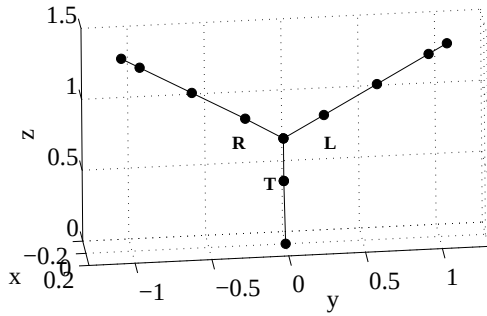


Fig. 1. A skeleton can be found by drawing segments between the Cartesian positions of some crucial joints

Segments “span” the kinematic structure of a manipulator (see Fig. 1), and a variable-radius surrounding volume is created. The underlying idea is that a solid of revolution can express the shape of a link: this can be partially modified considering different behaviours at different angles around the segment.

The resulting multiple volumes form a virtual region which has to approach the real volume of the considered part of a manipulator. Automatic skeleton building from Denavit–Hartenberg (DH) parameters is possible. In fact, a standard DH table gives the possibility of computing the *Cartesian position of each joint*. These positions can constitute the ends of the segments of the skeleton (*nodes*), and some of these possible nodes will be discarded if they coincide with other nodes already present in the skeleton. It is important to consider segments which cover the *spine* of all the mechanical parts: this has a reflex on DH tables when manipulator links have parts on both sides of a revolute joint as, e.g., for counterbalances or for allocating motors.

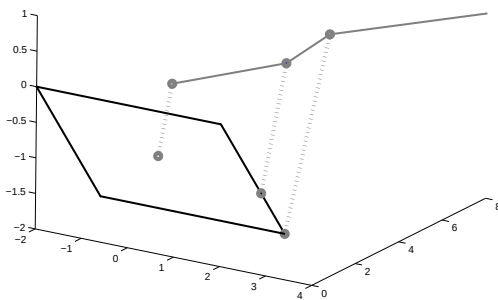


Fig. 2. The distances from segments (gray) to a rectangular regions (black) are computed as a straightforward extension with respect to the introduced distance formulas for segments

Built the skeleton, the focus is then on distance evaluation from the segments on the robot (bounding volumes) to the environment: this is the basis for motion control in an unstructured domain. The complete environment has to

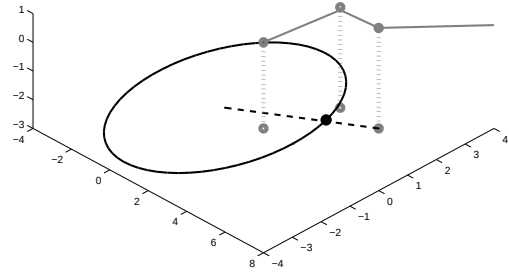


Fig. 3. Distances from a circular regions are computed by considering distances to a plane, and then from there to the center of the considered region

be modelled with geometric figures. In the design of the skeleton, some heuristics can help in discarding useless computation; nevertheless, the general case of computing all possible distances between simple objects like segments, regions of a plane (circles, rectangles) or points has the only limit of the time complexity for modelling the whole operation environment. The distances between these simple objects can be obtained via analytical formulas ([4]). Simple case of distance computation with segments are reported in [3, 6].

For more complex environments, the idea is to obtain planar figures for analytical computation. The computation of distances from planes represents a straightforward generalisation. Consider the equation of a plane in the form:

$$ax + by + cz + d = 0. \quad (1)$$

The distances between the extremal points of a segment on the robot and such a plane are then considered on the direction orthogonal to the plane, which is actually colinear to the vector $[a, b, c]^T$. The repelling force will act on one of the ends of the robot’s segment. If the closest point on the plane is outside from the considered polygonal regions, distances from the sides of the polygon are computed.

In addition, the tools for the continuous motion on the skeleton [8] avoiding jumps in the reference signals for the controller have to be implemented also, for discarding abrupt changes of closest points in the case of parallel segments, or segments parallel to a plane, or for reducing the number of control points.

In order to generate proper reference motion for the control points, in general, potential fields or different techniques can be used in order to generate the forces or velocities which will produce the desired motions. In ([3]) it is discussed how, e.g., repulsion forces can be derived from a potential function. These forces can be naturally used to compute avoidance torques at the manipulator joints via the Jacobian transpose. Nevertheless, it should be pointed out that suitable repulsion velocities could be likewise generated in lieu of forces.

III. DH-BASED MODELLING AND MODIFICATIONS TO DIFFERENTIAL KINEMATICS

Robot modelling via segments can benefit of an additional tool: the description of an arbitrary position on robot links with a set of DH parameters.

In other words, the possibility of choosing an arbitrary point on the skeleton is based on the consideration that every point on the structure can be considered as a control point. Such a point can be identified by a set of DH parameters. If one considers a manipulator and its direct kinematics equation, changing the value of its DH parameters results in the kinematics equations of another manipulator, whose end-effector is located before the real end-effector: that is equivalent to moving the control point of the structure.

For later computation of the desired joint velocities or torques, it is necessary to compute the Jacobians associated with the control points. In detail, since control points always lie on the spine of the robot links, the direct kinematics and the Jacobian computation can be carried out in a parametric way for a generic point p_i , which is located after the i -th joint. Considering the homogeneous transformation relating the $(i-1)$ -th frame (corresponding to the i -th joint) to the next frame, by simply replacing the DH parameter corresponding to the link length with the distance to the considered control point, a new “shorter” manipulator is considered for control. The values of direct kinematics and Jacobian for the specified control point have then to be considered by setting to 0 the DH values corresponding to frames located below the control point in the kinematic chain. If the displacements on the links vary continuously and sequentially, from the tip of the robot towards the base and vice versa, the whole skeleton can be spanned.

The interesting implementation in velocity control will be presented, since the differential kinematics equation has an important modification, due to the fact that not only the joint angles, but also other kinematic parameters in the DH table may change during the task. The additional suggested tools for a velocity-level implementation are the choice of a modular Jacobian, and the proper management of a moving control point.

For the purpose of control, the Jacobian matrix is the cornerstone: similarly to the previous discussion, a symbolic Jacobian can be used, where the kinematic parameters in the DH table change as described above, allowing the motion of the control point. The dimensions of such a matrix change, depending on the available degrees of freedom (DOFs) before the control point. The crucial aspect is that derivation of the differential kinematics equation is affected by the motion of the multiple control points. This can be also taken into account in a symbolic expression, as discussed below.

A. Differential kinematics with moving points

Considering the direct and differential mappings with the standard DH parameters, the usual differential kinematics

equation does not take into account the possibility of varying those kinematic parameters other than joint values. A complete model is as follows. The direct kinematics equation can be written in the form

$$p_i = k(q_i) \quad (2)$$

where the vector q_i contains the vectors of the standard DH variables d_i , a_i , θ_i , α_i . The differential mapping, discarding possible variations of the values in α_i , is the following

$$\begin{aligned} \dot{p}_i &= \frac{dk}{dt} = \frac{\partial k}{\partial \theta_i} \frac{\partial \theta_i}{\partial t} + \frac{\partial k}{\partial a_i} \frac{\partial a_i}{\partial t} + \frac{\partial k}{\partial d_i} \frac{\partial d_i}{\partial t} = \\ &= J_{\theta,i}(\theta_i, a_i, d_i) \dot{\theta}_i + J_{a,i}(\theta_i, a_i, d_i) \dot{a}_i + \\ &\quad J_{d,i}(\theta_i, a_i, d_i) \dot{d}_i. \end{aligned} \quad (3)$$

Given a control point p_i , the matrices $J_{a,i}$ and $J_{d,i}$ in (3) are the Jacobians which express the contribution to the motion of the control point of the variations of the DH values which characterise the control point. Moreover, θ_i expresses the vector of the joint values which contribute to the motion of the control point. Notice that $J_{\theta,i}$ is the ordinary Jacobian up to the control point, for a given set of DH parameters.

There are proper ways [9] for reducing the number of nonnull values in the d_i and a_i vectors of DH parameters. Often, this is intrinsically forced by the manipulator's design. As a simple case, consider a manipulator kinematics where only some values in the vector d_i change. In this situation, the way to compute the joint variables for a moving point on the skeleton of the robot is the following:

$$\dot{\theta}_i = J_{\theta,W,i}^\dagger (\dot{p}_i - J_{d,i}(\theta_i, a_i, d_i) \dot{d}_i) \quad (4)$$

where the subscript W for the pseudoinverse of the Moore–Penrose Jacobian matrix $J_{\theta,W,i}^\dagger$ (corresponding to the control point p_i) is referred to possible joint involvement weighing. Based on these simple modifications, multiple-point control, which has shown to be central in interaction with robots, can be accomplished easily both in force and velocity control.

The main issue is that the control points, with the associated Jacobian, move on the robot; therefore, this motion is taken into account in the differential kinematics.

The need for a modular expression of the introduced Jacobians comes naturally from the number of matrix multiplications which are necessary for an arbitrary number of DOFs.

B. Continuity of moving control points

When a control point is computed automatically, e.g., via distance evaluation from the closest obstacle (or goal) to the skeleton of the manipulator, there is the possibility that its position changes in a discontinuous fashion. Consider as an example the case of multiple obstacles approaching an articulated robot. Moreover, some heuristics or the need for a reduced number of control points can result

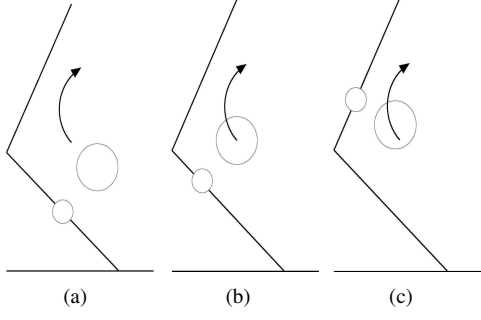


Fig. 4. The control point (small circle) on the skeleton, automatically computed as the closest to a moving object (big circle), can give a discontinuity to reference trajectories, moving abruptly on a new segment

in some sudden change of control point, and therefore of the corresponding DH values.

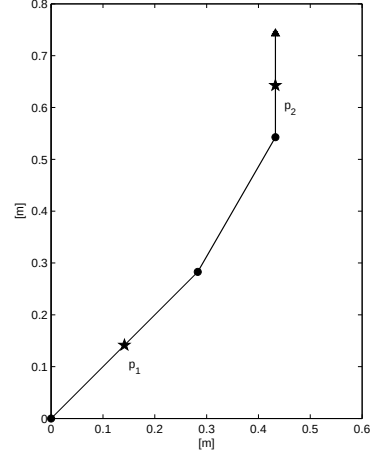
In order to avoid this problem, the DH values for the control point have to be forced to vary with continuity and *in the right sequence*: this corresponds to moving to the next control point *always lying on the skeleton*. This can be achieved, e.g., via spline interpolation resulting in moving the current control point towards next node of the skeleton, and then from there to the new control point, via sequent Cartesian position of the joints, i.e., nodes of the skeleton ([4]). The sequence of variation of the DH parameters is important for simulating such a motion on the spine of the links, and the use of spline interpolation is suggested for specifying values of the higher-order derivatives of DH values.

In order to reach the new control point via interpolation, a delay is to be considered before the change of the control point is performed. This delay has to be compatible with parameters related to the current motion of the robot such, e.g., the time-to-collision.

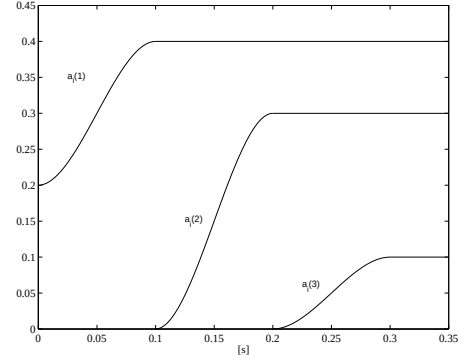
The possibility of smoothly moving the control point is useful also for forcing its motion on the skeleton in case of distance computation between parallel segments, where the computed closest point can move abruptly from an end to the other of the segment of the skeleton, in case of motion of an obstacle segment passing through a configuration which results in a parallelism with respect to the segment on the manipulator's skeleton.

Both first- and second-order inverse kinematics schemes [10] can be easily modified for taking into account the presence of the moving control point with variable kinematic parameters. In the case of second-order algorithms, the derivatives of the additional Jacobian matrices which have been introduced have to be computed also.

With reference to Fig. 4, it can be seen that a single control point which is automatically computed as the closest to a collision, based on environment modelling, can move abruptly on the articulated structure. The use of different filters for the motion on the control point depends on the control algorithm too: if the control uses higher order derivatives of the position, motion has to be smooth enough to ensure continuity of these data.



(a)



(b)

Fig. 5. When a control point's position is expected to change, e.g. from p_1 to p_2 (a), the desired new DH parameters are to be reached sequentially and with continuity (b)

Figure 5 shows an example of variation of the DH parameters which identify the control point for a three-link planar manipulator, whose lengths are 0.4 m, 0.3 m, 0.2 m. In such a manipulator, only the values in the vector $\mathbf{a}_i$, i.e., $a_i(1)$, $a_i(2)$, $a_i(3)$, change for a control point p_i on the skeleton. If the control point moves from the middle of the first link (p_1) to the middle of the third link (p_2), the resulting continuous and sequential change in the DH parameters, obtained via cubic spline interpolation, is reported. The time for reaching the desired final value for each parameter has been set always equal to 0.1 s. This time interval is a parameter for the spline interpolator. In this case, the change of three parameters results in reaching control point after 0.3 s. Meanwhile, the control point is moving on the skeleton and its motion is taken into account as described above.

IV. UNCERTAINTY AND DEPENDABILITY

The presented approach has been demonstrated with experiments related to real-time, fast whole-body collision avoidance ([3, 6]), and with simulations [8] for the moving control point described with DH parameters. It has not been demonstrated how to explicitly relate uncertainty to dependability.

Dependability issues in pHRI, with emphasis on the robot side, is related to the key problems of sensor capabilities and data fusion for inferring a correct characterisation of the scene and of the people in the robot environment. Dependability of complex robotic systems in anthropic domains during normal operation is threatened by different kinds of potential failures or unmodeled aspects in sensors, control/actuation systems, and software architecture, which may result in undesirable behaviours. Due to the critical nature of pHRI, dependability must be enforced not only for each single component, but for the whole operational robot [5]. A number of issues in the above discussion are related to the dependability of the interaction. The approach to modelling and reactive control using skeletal models allows fast modelling the scene. Having a fast model is good for prompt reaction to environmental changes, but the automatic computation of control points, depending on proprioception and exteroception, can result both in wrong evaluations and sudden motion on the manipulator's structure, and in delays. With reference to collision avoidance, the provided tools for fast modelling and control, only in an ideal case guarantee absence of impacts, when the positions of the possibly colliding parts are known with a precision coming from the dependability of encoders, as well as the accuracy of the kinematic model. When exteroception is used for crucial point detection, following automatic computation, e.g., of distances, results in selecting a control point. This control point can be wrong, the distance computation can then be wrong, and also the resulting reactive forces for attraction or repulsion. The error in sensing can affect the position of the control point in two ways: at first, just changing its position on the same segments of the ideal computed control point; however, another possibility is the change of the control point on another segment.

In sum, the following sequence of events can be considered:

- the position of an external point p_f (e.g., centroid of a user's face) is sensed;
- there is an error in position estimation, namely, the real centroid is in $p_f - \partial p_f$;
- the computed control point is different from the ideal one (distance from p_f);
- it can take more or less time to reach this new control point via the spline interpolator, from the previous control point: possibly, a change of segment occurs;
- according to the steps of the skeleton algorithm, computed distances and corresponding reactive trajectories are different from the ideal case;
- these errors can threaten safety and performance.

It has been discussed above how successive changes of segments have to be considered together with the time delay which is needed for motion on the skeleton. The system is dependable at least when these changes of control points do not result in delays which can be comparable to the maximum time before a collision occurs.

A higher level module is to be considered, for detecting situations (as in Fig. 4) where a (bounded) error on the

position can result in a change of segment. In case of a control point changing several times its position on a different segment, a possible solution is counting the number of switches, and to stop the robot in case of strange chattering behaviour, or forcing the control point on one of the two segments for a certain time.

The usual tools for uncertainty propagation for the differentiable model can be used, since discontinuities are smoothened as shown in Section III. A comprehensive discussion about these possible model is an on-going work.

V. SUMMARY

Summarizing the previous discussion, the introduced modelling and control techniques assume that skeletons are obtained from correct sensory data, which describe planes, polygons, segments. A correct sequence of control points will be therefore derived, with corresponding velocity or force trajectories. Moreover, it has been shown how to model a control point via DH parameters, and smoothen possible transitions between segments via spline interpolation. According to the presented algorithms, wrong sensory data may cause errors in the choice of the control point, depending on wrong positioning of identified geometrical objects. In particular, one has to evaluate the impact of sensory errors on wrong reference trajectories to the controller. The main on-going work for improving the introduced approach is the evaluation of the effects of modelling errors on desired computed forces/velocities. A more complete characterization is in progress, also using Virtual Reality techniques for safer simulations. The ultimate goal of this research is a possible definition of robustness and resilience properties for HRI tasks with industrial robots.

REFERENCES

- [1] A. De Santis, B. Siciliano, A. De Luca, A. Bicchi, "An atlas of physical Human-Robot Interaction", *Mechanism and Machine Theory*, Elsevier, vol. 43, pp. 253-270, 2008.
- [2] "Safely among us", *IEEE Robotics and Automation Magazine*, vol. 11(2), Special Issue on Dependability of Human-Friendly Robots, 2004.
- [3] A. De Santis, A. Albu-Schäffer, C. Ott, B. Siciliano, G. Hirzinger, "The skeleton algorithm for self-collision avoidance of a humanoid manipulators", *2007 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Zürich, CH, 2007.
- [4] A. De Santis, *Modelling and Control for Human-Robot Interaction, Research Doctorate Thesis*, Università degli Studi di Napoli Federico II, Italy, 2007.
- [5] PHRIENDS Research Project Home Page, <http://www.phriends.eu>.
- [6] A. De Santis, B. Siciliano, "Reactive collision avoidance for safer human-robot interaction", *5th IARP/IEEE RAS/EURON Workshop on Technical Challenges for Dependable Robots in Human Environments*, Roma, I, 2007.
- [7] C. Ott, O. Eiberger, W. Friedl, B. Bäuml, U. Hillenbrand, C. Borst, A. Albu-Schäffer, B. Brunner, H. Hirschi, S. Kiehlöfer, R. Konietzke, M. Suppa, T. Wimböck, F. Zacharias, G. Hirzinger, "A humanoid two-arm system for dexterous manipulation", *2006 IEEE International Conference on Humanoid Robots*, Genova, I, 2006.
- [8] A. De Santis, B. Siciliano, "Inverse kinematics of robot manipulators with multiple moving control points", *11th International Symposium on Advances in Robot Kinematics*, Batz-sur-Mer, F, 2008.
- [9] L. Sciacivco, B. Siciliano, *Modelling and Control of Robot Manipulators*, 2nd Ed., Springer-Verlag, London, UK, 2000.
- [10] F. Caccavale, S. Chiaverini, B. Siciliano, "Second-order kinematic control of robot manipulators with Jacobian damped least-square inverse: Theory and experiments", *IEEE/ASME Transactions on Mechatronics*, vol. 2, pp. 188-194, 1997.