

Estimation of Inverse Kinematics of Arbitrary Serial Chain Manipulators and Human-Like Robotic Hands

Susanne Petsch and Darius Burschka

Abstract—The general solution for inverse kinematics is a problem known for a long time. A lot of problems, algorithms, etc., depend on inverse kinematics. While solutions for specific serial or parallel chain manipulators exist, a good estimation for an arbitrary serial robot within a short computation time is still missing. We provide not only a tool for the *discrete* estimation of *inverse kinematics* of *arbitrary serial chain manipulators*, but also concepts for *inverse kinematics optimization* along entire paths (*adaptive tunneling*) and an approximation of a desired grasp with a human-like robotic hand (*virtual shut grasp*). The latter concept includes an efficient reduction technique of a complex, arbitrary hand, which enables a fast and accurate estimation of the inverse kinematics of an entire hand. In contrast to existing work, our approach is general, since it is neither restricted to certain configurations of the serial manipulator nor to specific structures of the hand. Moreover, our approach does neither rely on proximate starting positions nor does it require specific properties of the objective function concerning the position of the minima. It works even under the presence of multiple local minima in the solution space. Experiments show the performance of our system. The results of the estimation of the inverse kinematics are very accurate and the maximally required joint speeds along the paths are low.

I. MOTIVATION

A robot which is supposed to reach a certain position with its end-effector needs to be moved to an appropriate configuration to place the end-effector in the desired position. In many situations, it is advantageous to have a system which takes the desired end-effector position as input and gives one, several or all possible goal configurations as output. The necessary mapping from the robot's workspace to its joint space is called inverse kinematics. The general solution of inverse kinematics is a known problem for a long time. Sometimes, the inverse kinematics can be computed explicitly for manipulators with certain structures. A general estimation framework for the inverse kinematics of an arbitrary serial robot is still missing. It can already be enough to know, whether a solution exists. In other cases, the knowledge about one, several or all solutions is desirable.

We focus on the discrete analysis of the configuration-space. Such an analysis is, e.g., necessary for the evaluation of an arbitrary manipulator structure. We are interested in the inverse kinematics of an arbitrary serial chain manipulator at a given point without any incremental methods based on intermediate positions or (partial) derivatives (e.g., through

This work was supported by a cooperation with the German Aerospace Center (DLR).

Susanne Petsch and Darius Burschka are with the Machine Vision and Perception Group, Department of Informatics, Technische Universität München. 85748 Garching, Germany
{petsch,burschka}@in.tum.de

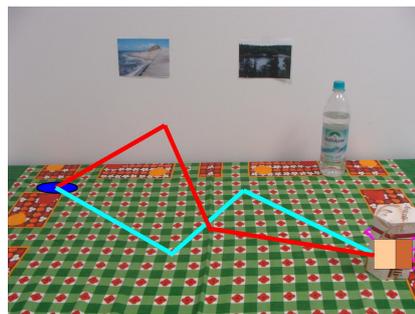


Fig. 1. The figure illustrates the problem of inverse kinematics for a manipulation scenario: The box should be transported. This means, first, that the object has to be grasped by the robotic hand (brown palm, magenta fingers). A serial chain manipulator has to put the hand into an appropriate position, such that a successful grasp is possible. Several configurations of the manipulator could be possible to reach a point (e.g., blue and red configurations of the robot). Then, the manipulator has to reach several desired positions along a trajectory for the transportation (not depicted).

the application of Jacobians). We do not assume any advantageous start configurations or something similar. Consequently, we formulate the problem of inverse kinematics as optimization problem. We look for configurations of the robot which minimize the distance between the robot's end-effector and its desired position (3D: position; 6D: pose/ position with orientation). As for other optimization problems, local minima can occur. Of course, we need to be able to overcome these, since we want to know, whether it is possible to reach the desired positions. Furthermore, we want to be able to find several different solutions without getting stuck in local minima. The solutions could be separated through one or several large maxima. This means, that we need to formulate our problem as a *global* optimization problem. Therefore, the entire configuration space is considered for the optimization. Such a global approach does not exist yet for an arbitrary robot. For example, the widely known Jacobian method depends on the start configuration of the robot. Hence, it can also suffer from local minima. The desired global optimizer has to run within a short computation time. We apply the stochastic approach for global minimization presented in [1], since it meets the above requirements. The optimization in [1] was developed in another context. Therefore, we need to develop a cost function in order to (1) check if a solution exists and to (2) find several solutions if there are more present. Hence, we cannot only minimize the distance between the desired and the real position of the end-effector in our cost function. If we have found one solution, we need to be able to find further solutions.

Furthermore, we want to determine a convenient solution of the inverse kinematics along an entire path, e.g., for the manipulation of an object. Of course, it is desirable to achieve consecutive configurations which are close to each other. Therefore, we introduce the concept of *adaptive tunneling*. Up to now, we formulated our optimization problem in a manner, such that we find one or more solutions for a single point. We are looking for these solutions in the entire search space, e.g., from 0 to 360 degree for each joint. Now, each configuration should be close to its ancestor. Hence, we limit the search space to a region around the configuration of the ancestor. A small search space is desirable to achieve consecutive configurations which are really close to each other. At the same time, a small space reduces the chance to find a good solution to attain the goal. Therefore, the size of the search space has to be adapted appropriately to ensure consecutive configurations close to each other, while the desired points are reached. Our concept of adaptive tunneling provides a framework to achieve this goal.

Of course, it is desirable to estimate also the inverse kinematics of more complex systems, while keeping the run time very small. However, the more DoF have to be solved by the optimizer, the longer it will take. An example is a human-like hand with 20 DoF which should grasp an object. If the desired positions of the finger tips and the thumb's tip are known, the position and the orientation of the hand as well as its 20 DoF have to be determined in such a manner, that the five tips can reach the goal positions. In order to deal with this complex problem, we build on the concept of the "virtual finger" [2], which is a combination of real fingers. An object can be grasped through a shut of the object between a thumb and a virtual finger. We call such a grasp a *virtual shut grasp*. The application of the virtual shut grasp reduces the number of DoF significantly from all DoF of the hand (e.g. 20) to a small number of DoF in form of a serial chain (7 DoF in our example, which is shown later).

To sum up, we provide very general tools for the global estimation of inverse kinematics (1) at single points for arbitrary serial chain manipulators, (2) along entire paths with consecutive configurations for arbitrary serial chain manipulators, and (3) of a complex human-like robotic hand.

The proposed discrete analysis of the configuration-space was already successfully applied in structure analysis [3]. Another application area was path optimization for abstractly represented tasks [4]. Up to now, the estimation methods were not described in detail. Hence, we want to present the detailed approach, since it renders very useful services.

The paper is organized as follows. After the overview of the related work, we present the estimation of the inverse kinematics for a single point, along a path and for a human-like robotic hand. Afterward, information about the data and the implementation as well as the experimental results are provided. We end with conclusions.

II. RELATED WORK

A lot of work exists in the field of inverse kinematics. Many applications and problems depend on inverse kinemat-

ics, e.g., [5]. A general introduction to the problem of inverse kinematics can be found in [6]. Solutions for special manipulators can already be found there. For example, Pieper's solution [7] is well-known. Moreover, there are also solutions for special redundant manipulators, e.g., [8]. Others make use of the null-space, which comes along with a redundant manipulator, e.g., [9]. Combinations of different methods have been developed (analytic, numerical methods; including optimization), e.g., in [10] or [11]. The Jacobian (including its inverse, resp., pseudo inverse) is also often used as, e.g., in [12]. In general, optimization methods have already been applied on the problem of inverse kinematics (e.g., [13]). The determination of the inverse kinematics of parallel chain manipulators has been presented as well (e.g., [14]). Inverse kinematics and path planning were integrated in [15] to move a manipulator arm from an initial configuration.

The concept of the virtual finger is originally used in the context of mapping a human grasp to a grasp for a robotic hand [2]. We adapt the original idea to our approach for the estimation of inverse kinematics. We just use a thumb and one virtual finger. In our case, the virtual finger is a combination of the real fingers without the thumb.

In contrast to existing work, we aim to estimate the inverse kinematics of an arbitrary serial chain manipulator and an arbitrary human-like hand (resp., position and orientation). The solution has to be general, so that it can be applied to any serial chain manipulator, resp., robotic hand even under the presence of redundancy. At first, we want to estimate whether a solution exists. If one or more solutions exist, we want to determine them. Otherwise, we are interested in a solution close to the goal position. We do not assume any pre-knowledge about (possibly) advantageous or known (start-)configurations. Nevertheless, local minima may not cause any problem. In contrast to local approaches (e.g., null-space), it has to be possible to find several optima, even if they are significantly separated. Although local optimizer can possibly find a solution of inverse kinematics, we need to apply a global optimization method to achieve the previously described aims. Additionally, the computation time should stay within a reasonable boundary.

It is important to distinguish our approach from other fields like path planning. We do not aim to, e.g., find a path between two points or analyze the entire workspace of the robot. We are interested in the *discrete* analysis of inverse kinematics for one or more *given* points.

III. APPROACH

We want to determine appropriate joint angles θ_j for all joints j in the joint space in such a manner, that the robot's end-effector is able to reach a desired position p_d in the workspace. This is called inverse kinematics. The mapping from the robot's joint space into the workspace can be described by a function $F: p_d = F(\Theta)$ with $\Theta = \{\theta_j\}$. We describe the robotic system in the DH-convention suggested by Denavit and Hartenberg [16] in the form shown in [6]. The orientation o of the robot's end-effector is described as Z-Y-X Euler angles [6].

The robot is supposed to reach the desired point p_d . The currently reached, real point is labeled with p_r . The euclidean distance e describes the remaining distance between p_d and p_r . A desired position is reached, if e is smaller than a certain tolerance t . A trajectory consists of n desired points p_i including the start point and end point. The configuration $C_i = \{\theta_{j,i}\}$ for all joints j of the robot corresponds to point p_i in the trajectory. The euclidean distance between two configurations is labeled with e_C . The total number of joints is M .

A. Inverse kinematics for a single point

In order to estimate the inverse kinematics for a single point, we want to minimize the distance between the real and the desired position. Therefore, we search for the minimum in the following objective function O_k :

$$O_k = \|p_r - p_d\| + r \quad (1)$$

with residual r . The residual r is zero for the search of the first solution of O_k . If more solutions should be found for the same point p_d , we need to search for further, different minima of O_k . Configurations which are close to already determined configurations should be avoided. Therefore, r is increased, when a new configuration is close to a previous, already known one. The minimal distance, that is necessary to declare two configurations as *different*, is labeled with u . If the euclidean distance e_C of the joint configurations is smaller than the distance u , the residual r is increased. r is, then, increased for each joint j in the configuration, if its distance $e_{C,j}$ is smaller than u/M :

$$r := r + \gamma \cdot \|e_{C,j} - u/M\| \quad (2)$$

with γ as a scaling factor. The higher the scaling factor, the higher the punishment of known configurations.

If we want to determine the 3D position of the end-effector as well as its orientation, we extend the previous objective function O_k to the following objective function O_o :

$$O_o = \|p_r - p_d\| + \alpha \cdot \|o_r - o_d\| + r \quad (3)$$

with o_d as desired orientation of the end-effector and o_r as the current orientation of the end-effector. The factor α is used as scaling, since the terms have different ranges.

If we want to transport an object, e.g., a cup filled with coffee, we are interested in keeping the object upright, but the orientation around the vertical axis of the object is allowed to change. Then, the objective function O_o needs just to be adapted slightly. Instead of including the orientations around all axes, we just use the orientations around the horizontal axes of the object in O_o . All described axes in this paragraph are, of course, in an object-centric point of view.

It should be pointed out, that the formulation of the objective functions is independent of the number of DoF of the manipulator. For example, the inverse kinematics of a redundant manipulator could be estimated as well. If the result of the first estimation within the objective function is higher than allowed, no solution is found. In this case, we get automatically the configuration of the robot which is closest

to the goal, since we search for the global minimum of the objective function.

B. Adaptive tunneling

We described already in the motivation, that we want to estimate the inverse kinematics of an arbitrary serial chain manipulator along a trajectory. Of course, it is desirable to get robot configurations which are close to each other along the trajectory to achieve smooth motions. Hence, we do not need to evaluate the entire search space. We just change the search space S of the optimizer. At the beginning, we use the entire search space ranging from 0 to 360 degree for each joint to determine the robot's configuration at the start position of the trajectory. For each following point p_i with $i > 0$, the search space is limited to a smaller search space S'_j :

$$S'_j = [\theta_{j,i-1} - l_j; \theta_{j,i-1} + l_j] \quad (4)$$

for each joint j . $\theta_{j,i-1}$ is the configuration of joint j at the previous point p_{i-1} . The variable l_j limits the search space around the previous configuration for joint j . At first, l_j is set to a small constant l_c . If the manipulator is able to reach the desired position, we are done. Otherwise, the limit l_j is adapted:

$$l_j := l_j \pm \delta \quad (5)$$

with

$$\delta = m \cdot \|(e - t)/M\| \text{ and } \|e\| > t \quad (6)$$

The sign in Eq. 5 is positive for the upper bound and negative for the lower bound. The adaption factor δ (Eq. 6) depends on the distance e between the desired and the real position. Hence, if the goal position is far away from the currently reachable positions, the search space is extended by a large step to overcome the gap to the goal. In the case of an adaption, $\|e\| > t$ must hold. Each l_j is, then, increased by the (proportionate, scaled) difference between the current distance e and the tolerance t . The distance $e - t$ is divided by M , since l_j limits the joint space for one joint j . The scaling m determines the step size of the adaption. The smaller m , the closer the consecutive configurations can be. However, the run time can increase at the same time, since more iterations can be necessary to reach the desired points. The process of adaption can be repeated until either a solution is found or if $S = S'$ without any solution. The objective functions O_k and O_o themselves (see Section III-A) are not changed. We just modify the search space.

If we concatenate the search spaces along the trajectory, we can illustrate them as a tunnel. The diameter of the tunnel depends on the size of the search space. This concept is independent of the step size between consecutive points or (partial) derivatives. The concept of adaptive tunneling is shown in Fig. 2. It supports consecutive manipulator configurations close to each other, while avoiding extensive computation times.

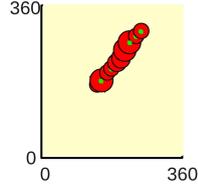


Fig. 2. Adaptive tunneling: The figure shows a tunnel in a 2D search space for a 2 DoF manipulator (overall range: 0° - 360° for each joint). The green dots are consecutive configurations. The red circles show the (possibly adapted) search space around the each configuration. Each consecutive configuration is searched within the red circle of its ancestor.

C. Virtual shut grasp

A virtual shut grasp is built up as follows. We use the thumb of the real hand and construct a “virtual finger” [2]. A virtual finger is a combination of real fingers. In our concept, it is the mean of all real fingers (mean of, resp., base, link length, orientation). We concatenate the real thumb and the virtual finger to a serial chain (from the tip of the thumb over the hand carpus to the tip of the virtual finger). An object can be grasped through a shut of the object between a thumb and a virtual finger. We call such a grasp a *virtual shut grasp*. Now, we attach the tip of the thumb to its 3D goal position. Then, we just need to determine the thumb’s orientation and the small number of DoF in the serial chain, so that the tip of the virtual finger reaches its desired position (the mean of the goal positions of all fingers).

Additionally, we need to make sure, that the serial chain has the right orientation. It has to be oriented in a such manner, that the real fingers are able to reach their goal positions afterward. Hence, we attach an “orientation stick” to the root $P_{v,root}$ of the virtual finger. The stick is aligned with the joint axis of the root of the virtual finger and points towards the root $P_{l,root}$ of the finger, which is the last one or defined as the last one. We use the little finger as the last finger. The orientation stick can be formulated as the vector $\overrightarrow{P_{v,root}P_{l,root}}$. This stick has to be parallel to an “aim stick” and it has to point in the same direction. The desired aim stick is the vector from the goal position of the virtual finger to a point $P_{g,goal}$ ($\overrightarrow{P_{v,goal}P_{g,goal}}$). To construct the point $P_{g,goal}$, we imagine a plane P_{vFP} within which the tip of the virtual finger can move. If a goal position $P_{f,goal}$ of a real finger is on the same side of the plane P_{vFP} as the goal position of the last finger $P_{l,goal}$, we introduce a new point $P'_{f,goal} = P_{f,goal}$. Otherwise, we mirror $P_{f,goal}$ on P_{vFP} to create $P'_{f,goal}$. The mean of all $P'_{f,goal}$ and $P_{l,goal}$ is the desired point $P_{g,goal}$, which is used for the aim stick. Fig. 3 illustrates the principle of the virtual shut grasp.

The determination of the virtual shut grasp gives us the orientation of the thumb as well as the position and orientation of the hand carpus. Afterward, just the joints of the real fingers need to be estimated (independently of each other). It is important to point out, that our approach is independent of the number of fingers and the form of the hand.

A robotic hand can be attached to the end-effector of a

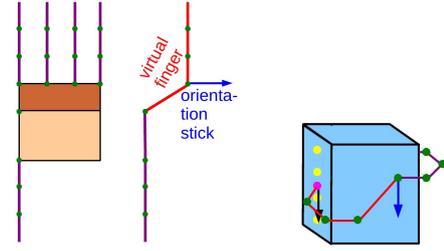


Fig. 3. Human-like robotic hand and virtual shut grasp. *Left*: Illustration of the human-like robotic hand with the brown palm of the hand, the dark magenta fingers and the green joints. *Middle*: Model of the human-like robotic hand with the red virtual finger and the blue orientation stick. *Right*: Illustration of a grasp with the virtual finger and the orientation stick: The yellow circles are the desired positions of the real fingers. The virtual finger should reach the light magenta mean of these positions. The blue orientation stick should be kept in the right orientation: It should be parallel to the black aim stick and point into the same direction.

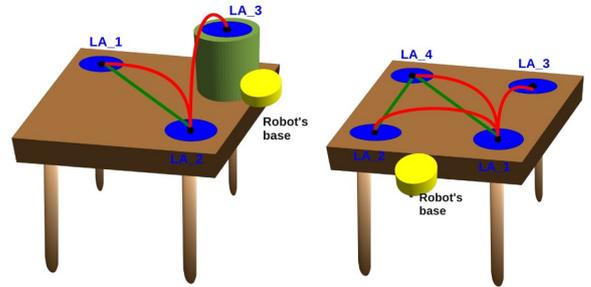


Fig. 4. Fig. 4. Data set I (left) and II (right): The lines/ curves between the blue characteristic places (LA) refer to trajectories of lifted objects (red) and trajectories of pushed objects (green). The robots’ base positions are symbolized in yellow.

robotic serial chain manipulator. If we want to use a hand and a manipulator for an action, e.g., object transportation, we can go on as follows. First, we determine the joint parameters as well as the orientation and position of the hand to grasp the object. Afterward, we just need to deal with the serial chain manipulator. We treat the hand carpus as the manipulator’s end-effector, which has, first, to position the hand for the grasp and, second, to keep the object upright during the transportation. Its inverse kinematics can be estimated as described in Section III-B.

IV. EXPERIMENTS

We test our approach on a manipulation scenario, in which the robot has to grasp an object at a certain position and to move it to another position in an upright orientation.

A. Data and Implementation

The experiments are processed on two data sets extracted from human actions in manipulation experiments (see [17]). The human transported objects between characteristic places. Fig. 4 illustrates both data sets used in the experiments. The manipulations take place in an area of about 40 cm x 40 cm.

The implementation is done in C/C++. The stochastic approach for global minimization was presented in [1]. We use the implementation by Oliver Ruepp [18]. The used computer has a Intel(R) Core(TM)2 Duo CPU.

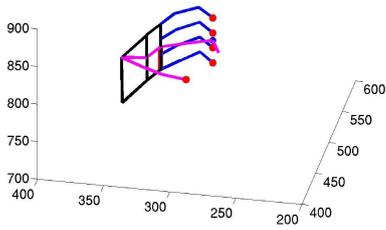


Fig. 5. Exemplary virtual shut grasp (Data set II). The illustrated hand has a black palm, blue real fingers and a magenta model of the hand with the virtual finger, the thumb and the red orientation stick. The red dots are the aim positions of the real fingers, the blue one is the goal of the virtual finger. The black axes of the global coordinate frame give an impression about the magnitude of the hand (unit: mm). As it can be seen, the virtual finger does not exactly reach its aim position. However, the real fingers are achieving their aim positions precisely.

The used serial chain manipulator has 6 DoF with a resp. link length of 300 mm. Each link is perpendicular to its resp. consecutive link. The human-like hand has a thumb and four fingers with three DoF each (resp. link length: 30 mm). The hand has one additional DoF between the root of the thumb and the roots of the fingers (see also Fig. 3). Each finger joint has a range from 0 to 120 degree.

The tolerance for a hit is set to $t = 20$. The variable γ is set to $\frac{1}{3}$ to punish configurations smoothly, when they are close to each other. The scaling $\alpha = 300$ is large, since the range of the second term in Eq. 3 is much smaller than the first one. Two configurations are treated as different, if the euclidean distance e_C in between is at least $u = 0.2$ rad. The search for further solutions is stopped, if either the result of the corresponding objective function is higher than $2.5 \cdot t$ or a maximal number of iterations is reached (200 in our experiments). We choose $l_c = 0.01$ and $m = 0.2$, in order to increase the search space just in very small steps to ensure smooth motions (see Eq. 4-6). Our trajectories consist of $n = 20$ consecutive points.

B. Results

We analyze a transportation task for each data set. Moreover, we repeat each experiment five times to check the stability of the experiments.

At first, we estimate the inverse kinematics of the human-like hand. An exemplary virtual shut grasp is depicted in Fig. 5. Nearly all of the grasps are done successfully. There is just one grasp among all experiments and repetitions, which could not be applied in reality: Since we do not have implemented a collision detection yet, the fingers are intersecting themselves in form of a loop. A collision detection could be added to the objective function in Section III-A as punishment of undesired intersections. Once the hand has successfully grasped the object, we just need to position the end-effector of the serial chain manipulator. It has to be positioned in such a manner, that the computed grasp can be performed. Some of the solutions are illustrated in Fig. 6.

Afterward, the desired manipulation is performed with the serial chain manipulator. The hand needs just to stay in the known grasp position to hold the object. We repeat the process of adaptive tunneling for different start

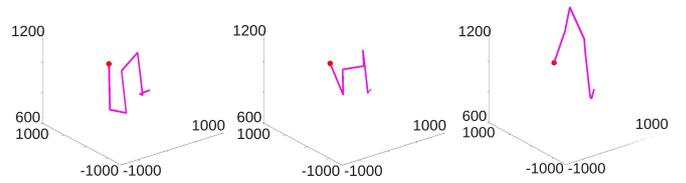


Fig. 6. Exemplary configurations of the magenta manipulator with the red base for the same position of the end-effector (data set I). For illustration, a stick stub was added to the end-effector instead of the hand to show its orientation more clearly. The black axes of the global coordinate frame are given as orientation between the subfigures (unit: mm).

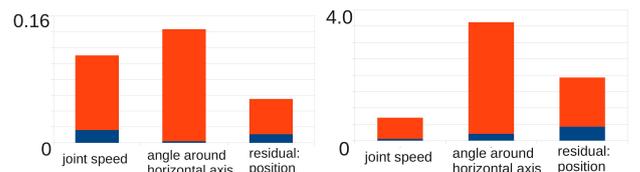


Fig. 7. Statistical results of the best (left) and the worst (right) results of adaptive tunneling for data set II. Each figure shows the blue mean and the orange maximum of the joint speed (in rad/time unit), the angle around the horizontal axes (in degree) and the residual of the position (in 10 mm).

configurations, if the start position can be reached with different configurations. The desired manipulation consists of a 3D trajectory, on which the object has to be kept upright (e.g., a cup with coffee), while the joint speed should stay low. Fig. 7 shows the average and maximum of, resp., the joint speed, the angle around the horizontal axes and the residual of the position for data set II. The best result is shown on the left, the worst on the right. Even the worst result has very desirable low values. This shows clearly, that the direct optimization of the angle around the horizontal axes and the residual in position (see Eq. 3) succeeds. The desired low joint speeds are achieved indirectly through the limitation of the search space in the adaptive tunneling method. The results for data set I are even better. Fig. 8 shows an exemplary development of joint speeds. On the right, we see, that one trajectory is done at an extremely low speed, but in a zig-zag-pattern. This can be explained through the application of the Stochastic Optimization: We search for the solution in a very small search space at the beginning. Within this space, the solution is picked arbitrarily. The joint speed of the trajectory in Fig. 8 is so small, that the solution is already found within this extremely small search space. Some trajectories have a peak in the development of the joint speed, but even this peak is very low with 0.5 rad. Fig. 9 illustrates some consecutive configurations of the manipulator along the trajectory. As we can see, its motions are smooth.

Concerning the computation time and the stability of the solutions, we achieve satisfying results. The estimation of the inverse kinematics of one single point takes between 15 and 60 seconds, depending on the number of possible solutions (between 20 and 80 solutions for a point in our experiments). The adaptive tunneling is started with the three best solutions of the inverse kinematics of the start positions, the best one according to the statistical values in Fig. 7

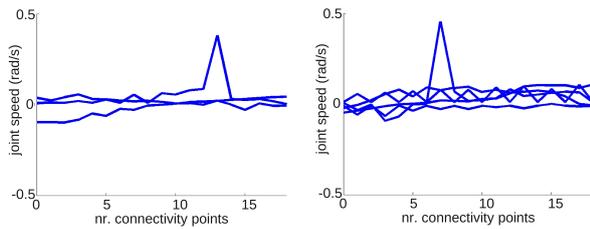


Fig. 8. Exemplary joint speeds along the trajectories. The x-axis can be seen as the time axis, if 1 s is needed to move to a consecutive point. *Left:* Speed of joint 1 in data set I. *Right:* Speed of joint 4 in data set II.

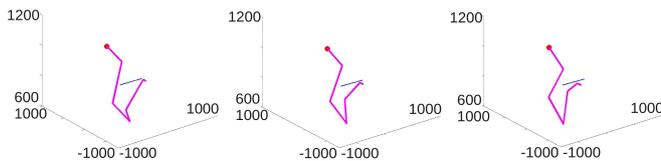


Fig. 9. Consecutive configurations of the manipulator along the blue trajectory at position 1, 5, 10 (data set I). Illustration similarly to Fig. 6.

is chosen at the end. The adaptive tunneling for the entire trajectory is done in 45-90 seconds. The inverse kinematics of the entire human like robotic hand is estimated within 5-10 seconds. We tested several local optimization methods (e.g., Matlab Optimization Toolbox: simplex search method, Trust-Region Dogleg method, Levenberg-Marquardt method) on the same data sets, but all of them got stuck in local minima. It was not possible to find the inverse kinematics for *all* desired points with any of these methods. This forced us to search for a *global* estimation of inverse kinematics. The small peaks in the speed profiles (see, e.g., Fig. 8) are negligible, if one considers, that it was not possible to find solutions with another method.

The number of solutions is stable for the inverse kinematics of a single point in data set I with just two outliers (just about 25 solutions found instead of about 40). For data set II, the estimation of the inverse kinematics seems to be more difficult for two places: About 50 solutions are found in ca. 50% of the repetitions, while just the half of this amount is determined in the other repetitions. The amount of possible solutions depends also on the definition of “different solutions”, e.g., the tolerance t and the minimal distance u between two different configurations. Moreover, the number of possible solutions can differ, depending on the real distance between two configurations. Nevertheless, the large amount of solutions in our experiments shows, that we find many possible solutions, even in the worst case. As described, the three best results of the inverse kinematics of the start position are evaluated before the adaptive tunneling. The best result is taken at the end. A third of the other solutions have single higher outliers in the joint speed about 1.5 - 2.5 rad/ time unit, the other values are approximately the same. The repetitions show similar results, just one outlier occurs in one repetition of one trajectory for the joint speeds (about 3 rad/ time unit). The estimation of the inverse kinematics of the hand is very stable, there is just one outlier (residual of 10 mm for a finger; normally a residual about

1.5 mm) among all repetitions, which is still very small.

V. CONCLUSION

We proposed a concept for the estimation of the inverse kinematics of arbitrary serial chain manipulators and a human-like robotic hand. We achieve good results for the inverse kinematics of the serial chain manipulator. The experiments with the efficiently reduced model of a complex, arbitrary hand show very good results of the estimation of the inverse kinematics.

Further applications of the described estimation approach can be found in [3], [4]. The estimations there include further features like another joint type (prismatic joint). The basic idea is the same as described here.

REFERENCES

- [1] C. Papazov and D. Burschka, “Stochastic global optimization for robust point set registration,” *Computer Vision and Image Understanding*, vol. 115, December 2011.
- [2] D. Lyons, “A simple set of grasps for a dextrous hand,” in *IEEE ICRA*, vol. 2, 1985, pp. 588–593.
- [3] S. Petsch and D. Burschka, “Analysis of manipulator structures under joint-failure with respect to efficient control in task-specific contexts,” in *Proceedings of the IEEE ICRA*, May 2013. To appear.
- [4] S. Petsch and D. Burschka, “Path optimization for abstractly represented tasks with respect to efficient control,” in *Proceedings of the IEEE ICRA*, May 2013. To appear.
- [5] F. Zacharias, C. Borst, and G. Hirzinger, “Capturing robot workspace structure: Representing robot capabilities,” in *IEEE/RSJ IROS*, 2007, pp. 3229 – 3236.
- [6] J. J. Craig, *Introduction to Robotics - Mechanics and Control*. Prentice Hall, 2005.
- [7] D. Pieper and B. Roth, “The kinematics of manipulators under computer control,” in *Proceedings of the Second International Congress on Theory of Machines and Mechanisms*, vol. 2, 1969, pp. 159–169.
- [8] H. Mohamed, S. Yahya, M. Moghavvemi, and S. Yang, “A new inverse kinematics method for three dimensional redundant manipulators,” in *ICCA-SICE*, 2009, pp. 1557 – 1562.
- [9] A. Liégeois, “Automatic supervisory control of the configuration and behavior of multibody mechanisms,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 7, no. 12, pp. 868–871, 1977.
- [10] D. Tolani, A. Goswami, and N. Badler, “Real-time inverse kinematics techniques for anthropomorphic limbs,” *Graphical models*, vol. 62, pp. 353–388, 2000.
- [11] R. Knoetschke and G. Hirzinger, “Inverse kinematics with closed form solutions for highly redundant robotic systems,” in *IEEE ICRA*, 2009, pp. 2945–2950.
- [12] R. Roberts and A. Maciejewski, “Nearest optimal repeatable control strategies for kinematically redundant manipulators,” *IEEE Transactions on Robotics and Automation*, vol. 8, no. 3, pp. 327–337, 1992.
- [13] A. Goldenberg, B. Benhabib, and R. Fenton, “A complete generalized solution to the inverse kinematics of robots,” *IEEE Transactions on Robotics and Automation*, vol. 1, no. 1, pp. 14–20, 1985.
- [14] F. Pierrot, A. Fournier, and P. Dauchez, “Towards a fully-parallel 6 dof robot for high-speed applications,” in *IEEE ICRA*, 1991, pp. 1288–1293.
- [15] D. Bertram, J. Kuffner, R. Dillman, and T. Asfour, “An integrated approach to inverse kinematics and path planning for redundant manipulators,” in *IEEE ICRA*, 2006, pp. 1874–1879.
- [16] J. Denavit and R. S. Hartenberg, “A kinematic notation for lower-pair mechanisms based on matrices,” *Journal of Applied Mechanics*, pp. 215–221, 1955.
- [17] S. Petsch and D. Burschka, “Representation of manipulation-relevant object properties and actions for surprise-driven exploration,” in *IEEE/RSJ IROS*, 2011, pp. 1221–1227.
- [18] O. Ruepp, “Recovery of structure and motion from monocular images under poor lighting and texture conditions,” Ph.D. dissertation, Technische Universität München, 2012.