Proceedings of the 4th International Symposium on Communications,
Control and Signal Processing, ISCCSP 2010, Limassol, Cyprus, 3-5 March 2010

Motion Capture with Constrained Inverse Kinematics for Real-Time Hand Tracking

Andreas Aristidou, *Student Member, IEEE,* Joan Lasenby

***Abstract*— Articulated hand tracking systems have been commonly used in virtual reality applications, including systems with human-computer interaction or interaction with game consoles. However, building an effective real-time hand pose tracker remains challenging. In this paper, we present a simple and efficient methodology for tracking and reconstructing $3d$ hand poses using a markered optical motion capture system. Markers were positioned at strategic points, and an inverse kinematics solver was incorporated to fit the rest of the joints to the hand model. The model is highly constrained with rotational and orientational constraints, allowing motion only within a feasible set. The method is real-time implementable and the results are promising, even with a low frame rate.**

I. Introduction

In recent years there has been a growing demand for reliable hand motion tracking systems, a technology used to turn the observations of a moving hand into $3d$ position and orientation information. Such information can be used to better analyse hand movements; for hand gesture recognition; to generate virtual figures for films or computer games; for human-computer interaction (HCI) including interaction with game consoles. However, building a fast and effective hand pose tracker remains challenging; the high dimensionality of the pose space, the ambiguities due to self-occlusions and the significant appearance variations due to shading make efficient tracking difficult.

There are many approaches for tracking and configuring the hand model. The hand gesture identification algorithms can be classified into 2 major classes: glove-based and vision-based methods. In general, glove-based methods are real-time, however they are expensive (e.g. P5 Data glove) and only detect limited finger movements with low accuracy. Wang and Popovic [1] and Fredriksson et al [2] proposed methods for hand tracking using a single camera and an ordinary cloth glove which was imprinted with a custom pattern; while this offers a simple, computationally cheap and promising solution, it is still not as reliable as the optical mocap systems. The vision-based methods, on the other hand, are more accurate but they have problems with occlusions, noise and spurious data. Lien and Huang [3] proposed a hand model together with a closed-form Inverse Kinematics solution for the finger fitting process. The $3d$ positions of the markers were obtained using colour markers and stereo vision, and the finger poses were chosen using a search method which finds the best solution amongst all possible positions. While this method is implementable in real-time, it is complex and can fail when different size models are used. De la Gorce et al [4] also proposed a $3d$ hand tracking approach from monocular video. Stenger et al [5], [6] proposed statistical methods, such as an Unscented Kalman Filter and a Hierarchical Bayesian Filter, to track hand motion. Such methods are still far from real-time thus limiting their use. Kaimakis and Lasenby [7] used a set of pre-calibrated cameras to extract the hand's silhouette as a visual cue. The $2d$ silhouette data is then modelled as a conic field and physiological constraints are imposed to improve the reliability of the hand tracking [8].

Marker-based motion capture has been demonstrated in several interactive systems (including but not limited to hand interaction); the results are highly accurate and easy to configure. There are, however, instances where we do not have many markers available or it is impossible to attach 3 markers on each limb segment; the large number of markers needed is often prohibitive. It may therefore be infeasible to reproduce the tracked object animation and reconstruct its skeleton model (i.e. the hand model). Hence, it is necessary to find a new way of capturing the movement of these articulated models, using the minimum possible number of markers. Such a method is presented in this paper, reproducing good estimates of real captured hand motion. Instead of attaching 3 markers on each limb segment, a single marker is attached and captured on each finger (end effector), 1 marker at the chain base (root) and 2 markers at strategic positions to help us define the hand orientation. The markers' positions are tracked using an optical motion capture system, such as [9]. However, prior knowledge about the geometry of the hand, the hand model and the restrictions of each joint are required. Joint constraints are applied to ensure that the hand motion is within a feasible set, giving a visually natural motion of the hand. An Inverse Kinematics solver is also incorporated to estimate the remaining joint positions and to fit them to the hand model. The proposed method is effective and real-time implementable.

II. Articulated Hand Model

Human motion is typically represented as a series of different configurations of a rigid multibody mechanism consisting of a set of segments connected by joints. These joints are hierarchically ordered and have one or more degrees of freedom (DoF). The DoFs represent the rotations relative to their parent joints, up to the root joint, for which the position and orientation are represented with respect to a reference coordinate system. Most motion capture systems reconstruct figures by tracking several markers placed over the body of

A. Aristidou and J. Lasenby are with the Department of Engineering, University of Cambridge, Cambridge, UK, CB2 1PZ. E-mail: aa462@cam.ac.uk and jl221@cam.ac.uk

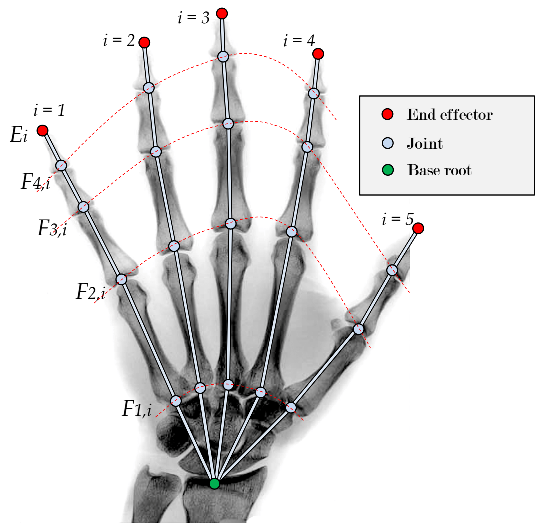

Fig. 1. The hand's model geometry used in our implementation.

TABLE I
HAND JOINT CONFIGURATION

	DoF	Rotational-x (degrees)		Rotational-y (degrees)		Orientational
		θ_1	θ_3	θ_2	θ_4	
$F_{1,i}$ $i = 1, \ldots, 4$	1	-	-	-	-	No twist
$F_{1,i}$ $i = 5$	2	20	20	30	40	No twist
$F_{2,i}$ $i = 1, \ldots, 4$	2	10	85	15	15	No twist
$F_{2,i}$ $i = 5$	2	5	30	10	10	No twist
$F_{3,i}$ $i = 1, \ldots, 5$	1	10	95	-	-	No twist
$F_{4,i}$ $i = 1, \ldots, 4$	1	10	90	-	-	No twist
Total	25					

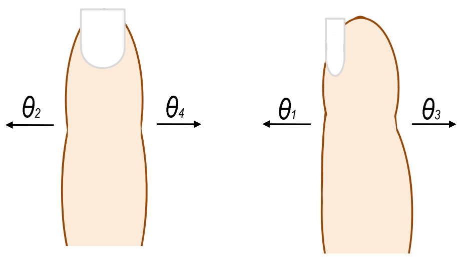

Fig. 2. Graphical representation of the angles $\theta_1, \ldots, \theta_4$ which define the rotational constraints of each joint.

the performer. Thus, in order to configure a human pose, it is important to position the end effectors in strategic positions as they are more easily specified by an animator and tracked by motion capture systems.

A joint, in the most general case, has 3 DoF. A bone rotation can be limited by factorising it into two rotations: one "simple rotation" that moves the bone to its final direction vector and one that represents the twist around that final vector. The proposed hand model consists of 25 joints and has in total 25 DoFs. Figure 1 shows a graphical representation of the hand's geometry used in our experiments and table I lists the degrees of freedom for each joint as well as its rotational and orientational limits. Figure 2 presents the angles $\theta_1, \ldots, \theta_4$ which define the rotational limits of each joint $F_{i,j}$. Fingers do not twist, thus only rotational constraints are applied, locking the joint orientation to be identical to its initial configuration with the root, so that the twisting configuration between joint and root will not change over time.

It is assumed that the hand geometry, meaning the initial joint configuration of the hand, is known. The end effector positions are captured using an optical motion capture system. Using inverse kinematics, we then tracked and reconstructed the hand poses over time. The markers are identified so that we know a priori on which finger each marker is placed. It is also important to know the orientation of the hand in order to efficiently incorporate constraints. This can be achieved by using markers which return both position and orientation information or by attaching 2 extra markers which will define the hand orientation. In this work, we used the 2 extra markers option. The data used in our experiments are captured from a markered optical motion capture system. Since our hand model does not have a mesh which defines its external shape, constraints related to the hand model, such as self-collisions, are not considered.

III. Inverse Kinematics

It is time-consuming for an animator to manually set all the DoFs of a virtual character. It is therefore more sensible to use a simulation mechanism, such as an Inverse Kinematics (IK) solver, to situate limbs according to their known end effector positions. The IK techniques require only positions and orientations of certain joints, usually named end effectors, to be specified by the animator and the remaining DoFs are automatically determined according to criteria that depend on the IK variant. The most popular IK methods use variants of the *Jacobian Inverse*, such as the Jacobian Transpose, Damped Least Squares (DLS), Selectively Damped Least Squares (SDLS) and several extensions [10], [11], [12], [13], [14], [15]. These methods linearly model the end effectors' movements relative to instantaneous system changes in link translation and joint angle. While Jacobian inverse solutions produce smooth postures, most of these approaches suffer from high computational cost, complex matrix calculations and singularity problems. Pechev in [16] proposes an IK solution from a control prospective which is computationally efficient and does not suffer from singularity problems. Recently, [17] and [18] proposed a Sequantial Monte Carlo Method (SMCM) and Particle filtering approach respectively. Neither method suffers from matrix singularity problems and both perform reasonably well. However, these statistical methods have high computational cost. [19], [20] used mesh-based IK techniques to configure the animated shapes. Mesh-based IK learns a space of natural deformations from example meshes. Using the learned space, they generate new shapes that respect the deformations exhibited by the examples, and satisfy vertex constraints imposed by the user. However, these methods require an off-line training procedure, their results are highly dependent on the training data and limited only to those models and movements on which the system has been trained. A well known IK method is the Cyclic Coordinate Descent (CCD) algorithm, which was first introduced by [21] and then biomechanically constrained by [22]. CCD is a heuristic iterative method with low computational cost for each joint per iteration, which formulates an IK solution very quickly. However, CCD suffers from unrealistic animation, even if manipulator constraints have been added, and often produces motion with erratic discontinuities. In this work, the joints have been configured using the FABRIK algorithm; FABRIK (Forward And Backward Reaching Inverse Kinematics) [23], [24] is a reliable iterative algorithm which uses points and lines to solve the IK problem. It has been successfully used for marker prediction and CoR estimation [25]. It divides the problem into 2 phases, a forward and backward reaching approach, and it supports biomechanical constraints, both rotational and orientational. It is fast, computationally efficient and provides visually smooth results.

FABRIK starts from the end effector position and works forwards, adjusting each estimated joint along the way until the root is reached. Thereafter, it works backward in the same way, in order to complete a full iteration. This method, instead of using angle rotations, treats finding the joint locations as a problem of finding a point on a line; hence, time and computation can be saved. A full iteration of the FABRIK algorithm is illustrated in pseudo-code in Algorithm 1.

The algorithm is repeated, for as many iterations as needed, until the end effector is sufficiently close to the target position. FABRIK always converges to any given chains/goal positions, when this is possible. If there are constraints which do not allow the chain to bend enough or if the target is not within the reachable area, there is a termination condition which compares the previous and the current position of the end effector, and if this distance is less than an indicted tolerance, FABRIK terminates its operation.

FABRIK can also cope with cases where the model has multiple chains and end effectors. This extension can be achieved by separating and treating the model as many different serial chains, connected with sub-bases. FABRIK can be easily adapted to support joint restrictions by readjusting the target position and orientation, on each step, to satisfy the joint biomechanical limits.

FABRIK can be easily extended to support rotational and orientational constraints; the main idea is to re-position and re-orient the target to be within an allowed range bound. This can be accomplished by checking whether the target is within the valid bounds, at each step of FABRIK, and if it is not, to guarantee that it will be moved accordingly. We assume that the hand has only rotational limits (the hand fingers do not twist). The allowed range of motion is defined by the angles $\theta_1, ..., \theta_4$, which represent the minimum and maximum allowed rotation of each joint about the x and y-axes, respectively. Figure 3 gives a graphical representation of the implemented constraints and the irregular cone describing the rotational motion bounds. More information on how FABRIK works, how joint limitations can be incorporated and how it can be extended to problems with multiple end effectors can be found in [23] and [24].

```
Input: The joint positions $\mathbf{p}_i$ for $i = 1, ..., n$., the target
       position $\mathbf{t}$ and the distances between each joint
       $d_i = |\mathbf{p}_{i+1} - \mathbf{p}_i|$ for $i = 1, ..., n-1$.
Output: The new joint positions $\mathbf{p}_i$ for $i = 1, ..., n$.
% The distance between root and target
$dist = |\mathbf{p}_1 - \mathbf{t}|$
% Check whether the target is within reach
if $dist > d_1 + d_2 + ... + d_{n-1}$ then
    % The target is unreachable
    for $i = 1, ..., n-1$ do
        % Find the distance $r_i$ between the target $\mathbf{t}$ and the joint
        position $\mathbf{p}_i$
        $r_i = |\mathbf{t} - \mathbf{p}_i|$
        $\lambda_i = d_i / r_i$
        % Find the new joint positions $\mathbf{p}_i$.
        $\mathbf{p}_{i+1} = (1 - \lambda_i)\,\mathbf{p}_i + \lambda_i \mathbf{t}$
    end
else
    % The target is reachable; thus, set as $\mathbf{b}$ the initial position of
    the joint $\mathbf{p}_1$
    $\mathbf{b} = \mathbf{p}_1$
    % Check whether the distance between the end effector $\mathbf{p}_n$ and
    the target $\mathbf{t}$ is greater than a tolerance.
    $dif_A = |\mathbf{p}_n - \mathbf{t}|$
    while $dif_A > tol$ do
        % STAGE 1: FORWARD REACHING
        % Set the end effector $\mathbf{p}_n$ as target $\mathbf{t}$
        $\mathbf{p}_n = \mathbf{t}$
        for $i = n-1, ..., 1$ do
            % Find the distance $r_i$ between the new joint position
            $\mathbf{p}_{i+1}$ and the joint $\mathbf{p}_i$
            $r_i = |\mathbf{p}_{i+1} - \mathbf{p}_i|$
            $\lambda_i = d_i / r_i$
            % Find the new joint positions $\mathbf{p}_i$.
            $\mathbf{p}_i = (1 - \lambda_i)\,\mathbf{p}_{i+1} + \lambda_i \mathbf{p}_i$
        end
        % STAGE 2: BACKWARD REACHING
        % Set the root $\mathbf{p}_1$ its initial position.
        $\mathbf{p}_1 = \mathbf{b}$
        for $i = 1, ..., n-1$ do
            % Find the distance $r_i$ between the new joint position
            $\mathbf{p}_i$ and the joint $\mathbf{p}_{i+1}$
            $r_i = |\mathbf{p}_{i+1} - \mathbf{p}_i|$
            $\lambda_i = d_i / r_i$
            % Find the new joint positions $\mathbf{p}_i$.
            $\mathbf{p}_{i+1} = (1 - \lambda_i)\,\mathbf{p}_i + \lambda_i \mathbf{p}_{i+1}$
        end
        $dif_A = |\mathbf{p}_n - \mathbf{t}|$
    end
end
```

Algorithm 1: A full iteration of the FABRIK algorithm

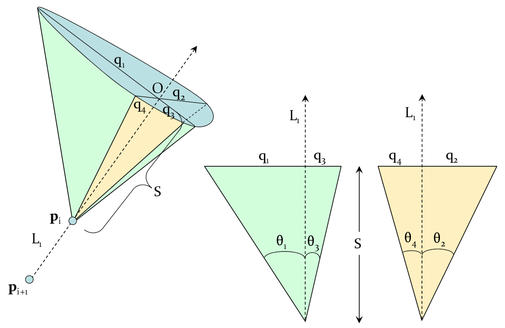

Fig. 3. A joint, $\mathbf{p}_i$, with its associated irregular cone which defines the allowed range of motion.

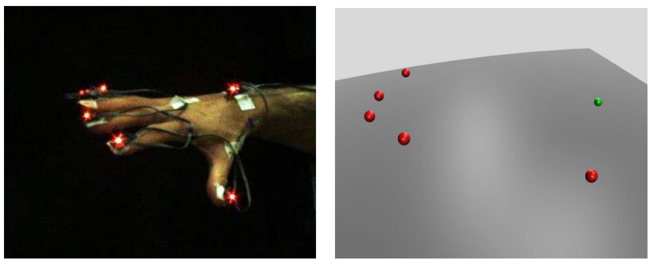
(a) (b)

Fig. 4. (a) The true hand pose, (b) the markers as seen from the motion capture system.

IV. Experimental Results

Experiments were carried out using a 10 camera Phasespace motion capture system capturing data at 100Hz [9]. The proposed algorithm can process up to 76 frames per second. Runtimes were measured with custom MATLAB code on a Pentium 2 Duo 2.2GHz. Data was also captured using a colour video camera, in order to compare the reconstruction quality between the estimated and the true hand postures. Figure 4 shows an example of the hand with attached markers and how the markers are seen from the motion capture system.

Our methodology is simple, it has low computational cost and performs well, returning smooth motion restricted only to a feasible set. The reconstruction quality can be checked visually by comparing the resulting poses with data captured using a colour video camera, as seen in figure 5. The results are promising, returning natural poses which meet the hand's rotational and orientational constraints. Figure 6 shows an example of continuous tracking and reconstruction using a dataset captured at a frame rate of 10Hz. The resulting motions do not suffer from oscillation or discontinuities and each finger smoothly moves to the target, having a posture within the feasible set.

The reconstruction quality varies at different frame rates. Clearly, as the frame rate becomes higher, the reconstruction quality is improved. However, even at a low frame rate (3 frames per second) the reconstruction quality of the proposed methodology is visually natural and biomechanically correct. Since our hand model is well constrained, these differences are minimal and difficult to observe in a single figure. The time needed for the IK solver to fit the joints to the model also varies for data captured at different frame rates. By reducing the frame rate, the distance between target and end effector is increased hence, FABRIK requires more time to track the target positions. Table II lists the average time needed per frame to track the hand under different frame rates.

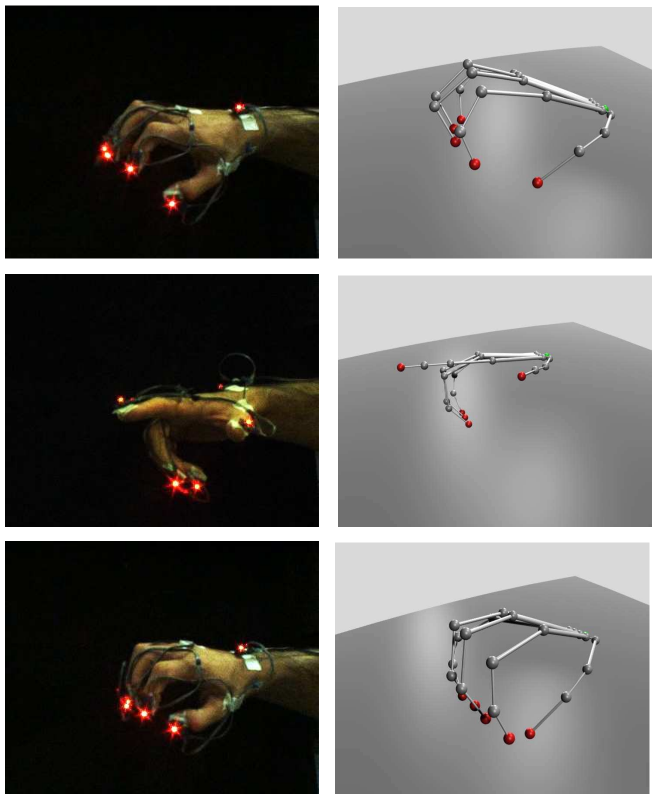
(a) (b)

Fig. 5. An example of hand reconstruction using our methodology at 100Hz frame rate. (a) The true hand pose, (b) the reconstructed figure. The resulting poses are visually natural and biomechanically correct.

The proposed method is simple and real-time implementable. It requires 1.13msec per frame for tracking and fitting 25 joints, meaning it can process on average 76 frames per second when the data frame rate is high, and 52 frames per second for a low frame rate dataset.

The resulting poses produced by the FABRIK algorithm are highly related to previous states; therefore, the final joint configuration might be different when the IK problem is solved with the end effectors in different initial positions but with similar final states. Nevertheless, these differences are minimal and can be further decreased by incorporating a hand model which considers, in addition to the hand rotational and orientational constraints, constraints related to the hand movement. One limitation of the work presented here is that we have assumed all markers are always visible. However, we are able to handle occluded markers using existing techniques to predict target positions [25], and hence use these predicted positions in the IK algorithm.

TABLE II
AVERAGE TIME NEEDED AT DIFFERENT FRAME RATES.

	Time per frame (msec)
Frame rate at 100Hz	1.1363
Frame rate at 10Hz	1.4152
Frame rate at 3Hz	1.9728

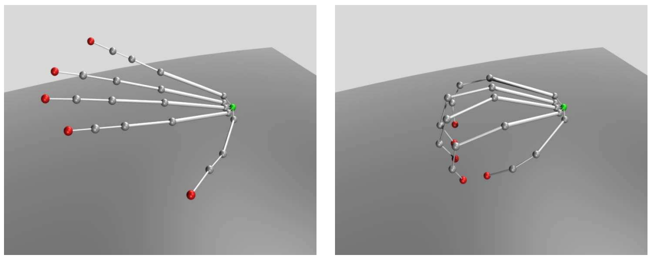
Fig. 6. An example of continuous tracking and reconstruction at a frame rate of 10Hz.

V. CONCLUSIONS AND FUTURE WORK

This prototype version is a first step towards an effective real-time hand motion tracking and reconstruction system. The results are promising, producing visually natural and biomechanically correct movements. The system can process up to 76 frames per second, meaning it is real-time implementable. Even with a low frame rate, the proposed methodology tracks the hand motion smoothly, without oscillations and with high reconstruction quality.

In future work, a more sophisticated model will be implemented which takes into account, in addition to the joint rotational and orientational restrictions, constraints related to the hand model, such as self-collisions, inertia, flexion etc. This extension will provide more accurate results, ensuring that the hand will have more natural poses, without violating any biomechanical or model constraint.

ACKNOWLEDGMENT

The authors would like to thank Dr. J. Cameron for his help capturing the data using the Phasespace [9] motion capture system and for his valuable and positive comments.

REFERENCES

[1] Robert Y. Wang and Jovan Popović. Real-time hand-tracking with a color glove. In *ACM SIGGRAPH 2009*, pages 1–8, New York, NY, USA, 2009. ACM.

[2] Jonas Fredriksson, Sven Berg Ryen, and Morten Fjeld. Real-time 3d hand-computer interaction: optimization and complexity reduction. In *Proceedings of the 5th Nordic conference on Human-computer interaction*, pages 133–141, New York, NY, USA, 2008. ACM.

[3] Cheng-Chang Lien and Chung-Lin Huang. Model-based articulated hand motion tracking for gesture recognition. *Image and Vision Computing*, 16(2):121–134, 1998.

[4] Martin de La Gorce, Nikos Paragios, and David J. Fleet. Model-based hand tracking with texture, shading and self-occlusions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008.

[5] Björn Stenger, Paulo R. S. Mendonça, and Roberto Cipolla. Model-based 3d tracking of an articulated hand. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 310–315, Los Alamitos, CA, USA, 2001. IEEE Computer Society.

[6] Björn Stenger, Arasanathan Thayananthan, Philip H.S. Torr, and Roberto Cipolla. Model-based hand tracking using a hierarchical bayesian filter. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28(9):1372–1384, Sept. 2006.

[7] Paris Kaimakis and Joan Lasenby. Gradient-based hand tracking using silhouette data. In *Proc.of the 3rd International Symposium on Visual Computing (ISVC)*, volume 1, pages 24–35, Lake Tahoe, NV/CA, USA, November 26-28 2007.

[8] Paris Kaimakis and Joan Lasenby. Physiological modelling for improved reliability in silhouette-driven gradient-based hand tracking. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19–26, Miami, FL, USA, June 25 2009.

[9] PhaseSpace Inc:. Optical motion capture systems, http://www.phasespace.com.

[10] A. Balestrino, G. De Maria, and L. Sciavicco. Robust control of robotic manipulators. In *Proc. of the 9th IFAC World Congress*, volume 5, pages 2435–2440, 1984.

[11] W. A. Wolovich and H. Elliott. A computational technique for inverse kinematics. *The 23rd IEEE Conf. on Decision and Control*, 23:1359–1363, December 1984.

[12] John Baillieul. Kinematic programming alternatives for redundant manipulators. In *Proc. of the IEEE International Conf. on Robotics and Automation*, volume 2, pages 722–728, March 1985.

[13] Charles W. Wampler. Manipulator inverse kinematics solutions based on vector formulations and damped least-squares methods. *IEEE Trans. on Systems, Man and Cybernetics*, 16(1):93–101, 1986.

[14] Y. Nakamura and H. Hanafusa. Inverse kinematic solutions with singularity robustness for robot manipulator control. *Trans. ASME, J. of Dynamic Systems, Measurement, and Control*, 108(3):163–171, September 1986.

[15] Samuel R. Buss and Jin-Su Kim. Selectively damped least squares for inverse kinematics. *J. of Graphics Tools*, 10(3):37–49, 2005.

[16] Alexandre N. Pechev. Inverse kinematics without matrix invertion. In *Proc. of the 2008 IEEE International Conf. on Robotics and Automation*, pages 2005–2012, Pasadena, CA, USA, May 19-23 2008.

[17] Nicolas Courty and Elise Arnaud. Inverse kinematics using sequential monte carlo methods. In *Proc. of the V Conf. on Articulated Motion and Deformable Objects*, volume 5098, pages 1–10, Mallorca, Spain, 2008.

[18] Chris Hecker, Bernd Raabe, Ryan W. Enslow, John Deweese, Jordan Maynard, and Kees van Prooijen. Real-time motion retargeting to highly varied user-created morphologies. *ACM Trans. on Graphics (TOG)*, 27(3):1–11, 2008.

[19] Robert W. Sumner, Matthias Zwicker, Craig Gotsman, and Jovan Popović. Mesh-based inverse kinematics. *ACM Transactions of Graphics*, 24(3):488–495, 2005.

[20] Kevin G. Der, Robert W. Sumner, and Jovan Popović. Inverse kinematics for reduced deformable models. In *ACM SIGGRAPH*, pages 1174–1179, New York, NY, USA, 2006. ACM.

[21] Li-Chun Tommy Wang and Chih Cheng Chen. A combined optimization method for solving the IK problems of mechanical manipulators. *IEEE Trans. on Robotics and Automation*, 7(4):489–499, 1991.

[22] Chris Welman. Inverse kinematics and geometric constraints for articulated figure manipulation. *Master Dissertation, Simon Fraser University, Department of Computer Science*, 1993.

[23] Andreas Aristidou and Joan Lasenby. FABRIK: a fast, iterative solver for the inverse kinematics problem. *Submitted to Graphical Models*, February 2010.

[24] Andreas Aristidou and Joan Lasenby. Inverse kinematics: A review of existing techniques and introduction of a new fast iterative solver. Technical Report F-INFENG/TR. 632, CUED, 2009.

[25] Andreas Aristidou and Joan Lasenby. Real-time marker prediction and CoR estimation in optical motion capture. *Submitted to Image and Vision Computing*, February 2010.