CHALMERS | UNIVERSITY OF GOTHENBURG

*MASTER'S THESIS*

# Manikin in Time; Development of the Virtual Manikin with External Root and Improved Inverse Kinematics

NICLAS DELFS

*Department of Mathematical Sciences*
*Division of Mathematics*
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2012

Thesis for the Degree of Master of Science

# Manikin in Time; Development of the Virtual Manikin with External Root and Improved Inverse Kinematics

Niclas Delfs

Department of Mathematical Sciences
Division of Mathematics
Chalmers University of Technology and University of Gothenburg
SE – 412 96 Gothenburg, Sweden
Gothenburg, May 2012

Matematiska vetenskaper
Göteborg 2012

# Abstract

Simulating manual assembly operations considering ergonomic load and clearance demands requires detailed modeling of human body kinematics and motions, as well as a tight coupling to powerful algorithms for collision-free path planning. The focus in this thesis is kinematics including balance and contact forces, and ergonomically preferable motions in free space. A typical manikin has more than 100 degrees of freedom. To describe operations and facilitate motion generation, the manikin is equipped with coordinate frames attached to end-effectors like hands and feet. The inverse kinematic problem is to find joint values such that the position and orientation of hands and feet matches certain target frames during an assembly motion. This inverse problem leads to an underdetermined system of equations since the number of joints exceeds the end-effectors' constraints. Due to this redundancy there exist a set of solutions, allowing us to consider ergonomics aspects and maximizing comfort when choosing one solution.

The most common approach to handle both forward and inverse kinematics is building a hierarchy of joints and links where one root must be defined. A popular place to define the root is in a body part, e.g. in a foot. This leads to a two-step procedure; (i) determining if re-rooting is necessary, (ii) solving the inverse kinematic problem using the Penrose pseudoinverse.

In this thesis we propose using a fixed exterior root by introducing six additional parameters positioning the lower lumbar - three rotations and three translations. This makes it possible to reposition the manikin without a series of re-rooting operations. Another important aspect is to keep the manikin, affected by internal and external forces and moments, in balance. However, by utilizing the exterior root and its added degrees of freedom it is possible to solve the balance, positioning, contact force and comfort problems simultaneously in a unified way.

A manikin was implemented, and some specific test cases demonstrate the applicability of the presented method and also use randomized goals to show the generality of the solver.

# Keywords

Advanced Biomechanical Models, Inverse Kinematics, Optimization Algorithms

# Contents

# Acknowledgements

Firstly I want to thank my advisor Robert Bohlin for guiding and supporting me throughout the work. Also I give my gratitude to the people at Fraunhofer-Chalmers Research Centre who have supported me. Special thanks goes to Stefan Gustafsson for help with the graphical user interface. Thanks also goes to Johan Carlson at Fraunhofer-Chalmers Research Centre who believes in and supports the manikin project.

Chapter 1

# Introduction

## 1.1 Background

Many companies are working with creating virtual environments where they can build their products, before actually deciding on the industrial facilities needed to do it in the real world. Companies do software prototyping with the goal to reduce time-to-market of a product and increase sustainability in production. There are many things that can be done to reach those goals; some important parts of the production to digitalize would be product assembling, robotic path planning and the factory workers. Software supporting such activities can determine if a product is possible to assemble and how the robots should move. It can also give information of how to reduce wear and tear of the production line by minor design changes of the product. Also, the parts that need to be assembled by humans should be well designed for increased comfortability so work related injuries can be reduced.

Today Computer Aided Design (CAD) is a commonly used environment for visualizing the product and the product assembling, e.g. combining all parts into the final product, and controlling that the assembly path is collision free. This is in some cases done by hand in CAD. Furthermore, the CAD models can be loaded into robot offline programming software for testing of the paths chosen. It is also possible to animate the worker in a digital environment with softwares like Jack [1], by Siemens, and Ramsis [2], by Human Solutions; this is very tedious work. With the software Industrial Path Solutions (IPS) [3], by Fraunhofer-Chalmers Research Centre (FCC), the assembly can be done automatically with rigid bodies since 2003 and with an increasing set of production robots. IPS reduces the time needed to plan paths for robotics and rechecking products after changes.

There is yet no software for easily gaining acceptable motions for digital humans in the assembly line. The thesis [4] describes the basic construction of such software and the aim of this work is to improve and generalize those methods.

## 1.2 Aim

The aim is to improve and expand the Inverse Kinematics (IK) solver from the earlier thesis [4]. In the earlier work the manikin was static in its root joint which is problematic for a simple and efficient IK solver. We will implement an exterior root to see if that solves the problem of a static root joint. Moreover, the pose that is predicted must be in balance to be of use. Therefore, static physics will be implemented and simple controls to move the center of mass. Also new constraints can be created and the old ones from earlier work can be improved, by relaxation. To keep the computation time from increasing we will implement ways to decrease the computational cost for the algorithm.

## 1.3 Disposition

The thesis will start with describing the model structure and how the optimization uses generic constraints. Then, all the different types of constraints are described, and after that the speed up of the algorithm is described in Chapter 4. In Chapter 5 the implementation is described and in Chapter 6 the results are presented. Last follows a discussion of the results and their feasibility, and the possibility to continue the work.

## 1.4 Previous Work

The project of creating a manikin at FCC started with the bachelor thesis [5] where some basic IK solvers where tested. These trials made the ground for the research project Intelligently Moving Manikin in Assembly (IMMA) which is a collaboration of many Swedish manufacturing companies. The basic model of the manikin used in this project is described in the master thesis [4] which is the prequel to this thesis.

## Chapter 2

# Unified Solution with External Root

A common approach to model the human body that can handle both forward and inverse kinematics is by constructing a hierarchy of joints and links where one root must be defined. A popular place to define the root is in a body part, e.g. in a foot. When the user interacts with the manikin by repositioning target frames, i.e. using the IK solver, a two-step procedure is needed; (i) determining if re-rooting is necessary, (ii) solving the IK problem.

This thesis introduces a fixed external root by adding six additional parameters, three rotations and three translations, that positions the lower lumbar. This makes it possible to reposition the manikin without a series of re-rooting operations. Another important aspect is to keep the manikin, affected by internal and external forces and moments, in balance at all times. By utilizing the exterior root and its added degrees of freedom (DoF) it is possible to solve the balance-, positioning-, contact force- and comfort problems simultaneously in a unified way.

## 2.1 Related Work

When using a hierarchical model of links, one root must be defined. Many manikin models presented in the literature use a foot or the lower torso as root. It is also common that the positioning of the root is handled differently than for the other links. In [6][7], the root has a fixed position with respect to a global coordinate system, which restricts the inverse kinematic calculations in an undesired way. Models without the ability to change root usually set the root to the lower torso, see [8]. This, however, gives complications for the balance control.

One way to handle the fixed root is to introduce a high-level algorithm that

changes the tree hierarchy and/or repositions the root when needed. This is done in the works by [9][10] where the root is set in a body part, but when and how to re-root is not described in detail.
In the work [11], an external root with six DoF is implemented with the purpose of matching motion capture data, but the inverse kinematics calculation does not utilize this extra freedom fully.

## 2.2 Unified Solution

This chapter presents the manikin model and the method used to solve the balance-, positioning-, contact force- and comfort problems simultaneously in a unified way, which was published in [12]. The model is described briefly here and for more detail see [4]. Parts of the method is related to the resolved motion-rate method presented in [10].

### 2.2.1 The Manikin Model

The manikin model is a simple tree of rigid links connected by joints. Each link has a fixed reference frame and its position is described relative to its parent link by a rigid transformation $T(\theta)$. Here $\theta$ is the joint value that sets the angle or distance to the joint's parent. For simplicity, each joint has one DoF, so a wrist, for example, is composed by a series of joints and links. To position the manikin in space, i.e. with respect to some global coordinate system, we introduce an exterior root at the origin and a chain of six additional links denoted exterior links - as opposed to the interior links representing the manikin itself. The six exterior links consist of three prismatic joints and three revolute joints. Together, the exterior links mimic a rigid transformation that completely specifies the position of the lower lumbar. In turn, the lower lumbar represents an interior root, i.e. it is the ancestor of all interior links.
Note that the choice of the lower lumbar is not critical. In principal, any link could be the interior root, and the point is that the same root can be used through a complete simulation. No re-rooting or change of the tree hierarchy will be needed.
With the joint vector $\Theta = [\theta_1, \dots, \theta_n]^T$, we can calculate all the relative transformations $T_1, ..., T_n$, traverse the tree beginning at the root and propagate the transformations to get the global position of each link. We say that the manikin is placed in a pose, and the mapping from a joint vector into a pose is called forward kinematics. Furthermore, a continuous mapping $\Theta(t)$, where $t \in [0, 1]$, is called a motion.

### 2.2.2 Kinematic Constraints

In order to facilitate the generation of realistic poses that also fulfill some desired rules we can add a number of constraints on the joint vector. These kinematic constraints can be defined by a vector valued function such that

$$g(x) = \mathbf{0} \tag{2.1}$$

must be satisfied at any pose, where $x^T = [\Theta^T, f^T]$ and $f$ is the contact forces and moments and $\Theta$ contains all the joint values. Finding a solution to (2.1) is generally referred to as inverse kinematics. In Chapter 3 we describe some possible constraints, for example balance constraints and positioning constraints.

### 2.2.3 Comfort Function

When generating realistic manikin poses and motions, it is essential to quantify the ergonomic load. To do so a scalar comfort function

$$h(x) \tag{2.2}$$

is introduced, capturing as many ergonomics aspects as is desire. The purpose is to be able to compare different poses in order to find solutions that maximize the comfort. The comfort function is a generic way to give preference to certain poses while avoiding others. Typically, the comfort function considers joint limits, forces and moments on joints, magnitude of contact forces etcetera. Note that it is straightforward to propagate the external forces and moments and the accumulated link masses trough the manikin in order to calculate the load on each joint. The loads on joints are one of the key ingredients when evaluating poses from an ergonomics perspective [13].

It seems like fairly simple rules can be very useful here. Research shows that real humans tend to minimize the muscle strain, i.e. minimize the proportion of load compared to the maximum possible load [14], so by normalizing the load on each joint by the muscle strength good results can be achieved. A suitable comfort function may also depend on the motion speed and how many times the same motion will be repeated.

### 2.2.4 Combined Solution

In this section we present a method for a unified treatment of balance, contact forces, position constraints and ergonomics. Often in practice, the number of constraints is far less than the number of joints of the manikin. Due to this redundancy there exist many solutions, allowing the possibility to consider ergonomics aspects and maximizing comfort when choosing a solution. The combined problem of (2.1) and (2.2) is formulated as follows:

$$\begin{cases} \text{maximize} & h(x) \\ \text{subject to} & g(x) = \mathbf{0} \end{cases} \tag{2.3}$$

This nonlinear optimization problem is solved by iteratively linearizing the problem. Let $x_0$ be the current state. Then, for small $\Delta x$,

$$g(x_0 + \Delta x) \approx g(x_0) + J(x_0)\Delta x$$

where $J(x_0) = \frac{\partial g}{\partial x}$ is the Jacobian at $x_0$. In order to satisfy $g(x_0 + \Delta x) = \mathbf{0}$ while increasing $h(x)$, $\nabla h(x_0)$ is projected onto the null space of J, thus taking the step

$$\Delta x = -J^{\dagger} g(x_0) + \lambda(I - J^{\dagger}J)\nabla h(x_0) \tag{2.4}$$

where $J^{\dagger}$ is the Penrose pseudoinverse and $\lambda$ is a scalar calculated by a line search, taking into account that $\lambda$ should stay within a reasonable domain where the linearization is assumed to be valid. The step $\Delta x$ is also truncated so that each value stays within its joint range. This is described in more detail in [4].

## Chapter 3

# Kinematic Constraints

Instead of changing each joint angle manually until the desired pose is found, an IK algorithm is used to give the user fast pose predictions. Goals need to be defined for the pose predictor to fulfill. In both robotics [15] and digital human modeling [10][4][6], tool center points (TCP) and corresponding target frames are used, where the goal is to make the TCPs coincide with their corresponding target frames. This is a kinematic position constraint but a kinematic constraint can also be a condition for the velocities and accelerations of the links of a manipulator [15]. While position constraints are necessary e.g. for following objects with a hand, they alone do not give realistic poses. The manikin would get out of balance when it moves and therefore constraints that enforce moment balance and force balance are needed. The moment balance is defined as when the sum of the moments induced on the manikin by gravity and all forces ($g_M$) is equal to zero, and force balance is defined as when the sum of all forces ($g_F$) is equal to zero, see section 3.2. In this thesis we use the constraint function $g$, see (2.1), which is a construct of the part constraints functions as follows

$$g(x) = \begin{pmatrix} g_P(\Theta) \\ g_F(f) \\ g_M(x) \end{pmatrix}$$

Since the positioning error $g_P$ does not depend on $f$, we for simplicity write $g_P = g_P(\Theta)$ and similarly $g_F = g_F(f)$. The constraint function $g_P$ will in this text be seen as one position constraint while in implementations it may be a combination of different position constraints.

To solve the system (2.3) with the iterative numerical solver from [4] the derivative

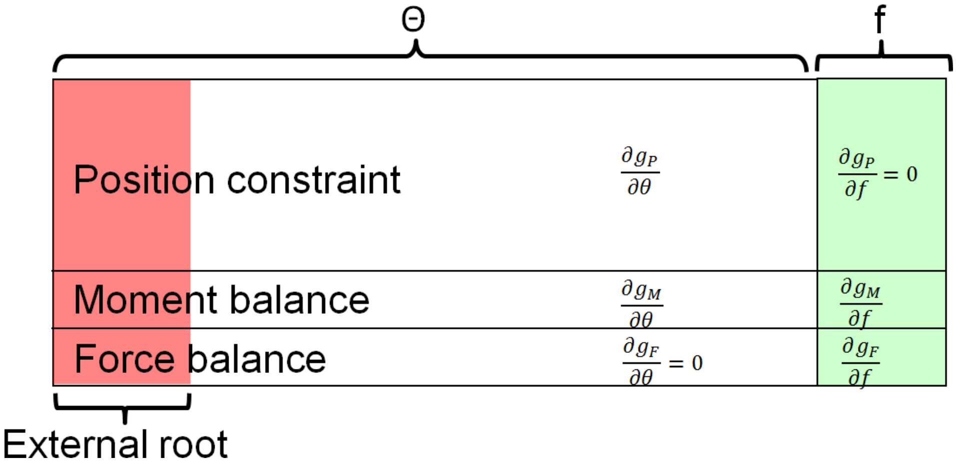

Θ
f
Position constraint
$\frac{\partial g_P}{\partial \theta}$
$\frac{\partial g_P}{\partial f} = 0$
Moment balance
$\frac{\partial g_M}{\partial \theta}$
$\frac{\partial g_M}{\partial f}$
Force balance
$\frac{\partial g_F}{\partial \theta} = 0$
$\frac{\partial g_F}{\partial f}$
External root

Figure 3.1: Schematic view of $\frac{\partial g}{\partial x}$ illustrating how constraints are piled up.

of $g$ is needed, which can be calculated separately for each constraint

$$\frac{\partial g}{\partial x} = \begin{pmatrix} \frac{\partial g_P}{\partial x} \\ \frac{\partial g_M}{\partial x} \\ \frac{\partial g_F}{\partial x} \end{pmatrix}$$

.

## 3.1 Positioning of the Manikin

One of the more basic constraints of the manikin are the position constraints. They make a TCP coincide with the corresponding target frame, either relative to other links or with respect to the global coordinate system. Typical examples of such constraints keep the feet on the floor, the hands at specific grasp positions, and the eyes pointing towards a point between the hands. Each local joint frame or TCP frame $T_{top}$ in the kinematic model can be transformed to the global system by

$$Q = T_{n_1}(\Theta_{n_1}) \dots T_{n_k}(\Theta_{n_k}) T_{top}. \tag{3.1}$$

where $n_1, n_2, \dots, n_k = n$ are indices to the connected chain of links from the root to link $n$.

### 3.1.1 Basic Position Constraints

For a TCP frame $Q$ and a target frame $Q_T$, the distance between them, also called tracking error, can be expressed in se(3) as

$$g_P(\Theta) = log(Q(\Theta)^{-1} Q_T). \tag{3.2}$$

The derivative of the tracking error is dependent on the joints. In each joint the hierarchy before and after can be seen as rigid bodies as a consequence of 3.1. Let $V_i$ be the vector from the joint center to the TCP, and let $\alpha_i$ be the axis of rotation in local TCP coordinates. Then for the rotation joints we have [4]

$$\frac{dQ}{d\Theta_i} = \begin{pmatrix} \alpha_i \times V_i \\ \alpha_i \end{pmatrix}. \tag{3.3}$$

Because the translation joints give a linear change in the direction of the joint axis, the derivative is

$$\frac{dQ}{d\Theta_i} = \begin{pmatrix} \alpha_i \\ \mathbf{0} \end{pmatrix}. \tag{3.4}$$

### 3.1.2 Different Positioning Constraints

While [4] only incorporates basic position constraints it is of interest to have more possibilities. One extension of the basic constraint is the relaxed constraints, e.g. that the feet are able to move freely on the floor, or that the eyes follow an object [9]. Furthermore, constraints that do not use static points in SE(3) can be of use e.g. constraining a hand to support the other arm [9] or keeping line of sight to the hand while changing gears [16]. These will be explained in the following sections.

#### Ensuring Loops Between TCPs

There are cases where the location of a TCP should coincide with another TCP as $Q_1 = Q_2 T$ where T is an offset transformation from TCP frame $Q_1$ and the TCP frame $Q_2$. From (3.2) with $Q_T$ changed to $Q_2(\Theta)T$ we have the error

$$g_P(\Theta) = \log(Q_1(\Theta)^{-1}(Q_2(\Theta)T))$$

Because each joint changes the model part before and after the link as a rigid body, only the joints in the kinematic loop containing the two TCPs affect the relative distance in SE(3). From equation above and (3.3) the new derivatives are $\frac{dQ_1}{d\Theta_i} - \frac{dQ_2}{d\Theta_i}$ for a joint $i$ that is in the loop, otherwise $\frac{dQ}{d\Theta_i} = \mathbf{0}$.

### 3.1.3 Relaxed Constraints

When defining the targets for TCPs it is sometimes interesting to define a subset $S \subset SE(3)$ instead of a point, e.g. a foot that is bound to be on the floor. This forces the Jacobi elements and the error vector of that constraint to be calculated in $S$. This can be done by using a projection $P$ that takes an element $T$ from SE(3) to $S$ by $PT = T_S \in S$. This can be done by applying a linear operation to the formulation (2.1)

$$g_P(\Theta) = \begin{pmatrix} x(\Theta) \\ y(\Theta) \\ z(\Theta) \\ r_x(\Theta) \\ r_y(\Theta) \\ r_z(\Theta) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

which is transformed into

$$Lg_P(\Theta) = L\mathbf{0}.$$

Any linear operation $L$ can be used. Two operations of special interest are

$$L = \begin{pmatrix} I & \mathbf{0} \end{pmatrix} \Rightarrow \textit{position}$$

and

$$L = \begin{pmatrix} \mathbf{0} & I \end{pmatrix} \Rightarrow \textit{rotation}$$

where $I$ is a three times three identity matrix. When using simple operations to reduce the number of variables we can simply remove the rows from (3.3), (3.4) and (2.1) corresponding to the elements that are removed by the relaxation [9]. There is therefore no need of creating the operation $L$ or the full Jacobian matrix in many cases of relaxations.

## 3.2 Balance of the Manikin

When doing pose predictions with the manikin in [4] the resulting poses sometimes are clearly out of balance; this makes the pose unusable for ergonomical evaluation.

In this thesis the balance is defined as when the total moment and forces of the manikin are zero. We separate between internal and external forces and moments; internal forces are generated inside the links and always neutralized by its neighbor links. Therefore only the external forces are needed for the balance equations. We assume that external forces such as gravity and contact forces, affect the manikin at infinitesimal points that are on a constant place relative to a joint, referred to as center of mass and contact points.

The manikin must be able to handle multiple contact points. This can be implemented in different ways. A simple solution would be to let the user set a weight value $\sigma$ for each support where $\sum_{i=1}^{\#Supports} \sigma_i = 1$, see [10][17]. These $\sigma_i$ are then used as the proportion of the z-axis forces this contact point should hold. This demands an advanced knowledge of human poses to be able to choose these values with the result of an ergonomical pose. In [18] the forces are calculated automatically while calculating the dynamic motion. This method has a limitation in that it needs a number of state response constraints that help generate the motion of the system. It needs different state response constraints for different motions and few of those have been defined. Here an extended version of [10] will be used where the forces are not pre-set, instead they are calculate by extending $x$ with the unknown forces and moments $f$.

### 3.2.1 Balance Equations

To get the total force and the total moment of the body to zero. The total force is

$$g_F(f) = \sum_{i=0}^{n} m_i \hat{g} + \sum_{k=0}^{m} f_k \tag{3.5}$$

where $f_k$ is a contact force affecting the manikin, $\hat{g}$ is the gravity force vector, n is number of links and $m$ is the number of forces. To calculate the total moment we need the moment created by gravity on each link and by the contact forces. The moment created at point $P \in \mathbb{R}^3$ by a force $f_k$ will be the torque resulting from when $f_k$ is affecting a contact point $q$, and is calculated by $(q - P) \times f_k$. The gravitational effect on the manikin will be calculated by summing up each link's moment influence resulting from the gravitation. On each link with the center of mass $c_i$ there is a force from the gravitation times the mass $m_i$ of the rigid body. The total moment on the manikin is then

$$g_M(x) = \sum_{i=0}^{n} (c_i(\Theta) - P) \times m_i \hat{g} + \sum_{k=0}^{m} (q_k(\Theta) - P) \times f_k + t_k \tag{3.6}$$

where $t_k$ is a contact moment.

As can be seen, the balance constraint is divided into two parts, the moment constraint and the force constraint. These two can be divided into $\Theta$ or $f$ dependence and can be calculated separately, see figure 3.1.

### 3.2.2 Differentiation of Moment and Force Constraints

Because of independence between $\Theta$ and $f$ in $x$ we will differentiate (3.6) in two parts. In the differentiation with respect to $\Theta$ we use the math from the section on angular velocity in [4]. Let $C(j)$ be the indices to the subtree rooted at link $j$. Then the rotation joints' derivatives are

$$\frac{\partial g_M}{\partial \Theta_j} = \sum_{i \in C(j)} (\alpha_j \times (c_i - p_j)) \times m_i \hat{g} + \sum_{k=0}^{m} (\alpha_j \times (q_k - p_j)) \times f_k \tag{3.7}$$

where $\alpha_j$ is the rotation axis of joint $j$. The translation joints give no angular velocity and therefore give

$$\frac{\partial g_M}{\partial \Theta_j} = \alpha_j \times m_j \hat{g} + \sum_{k=0}^{m} \alpha_j \times f_k. \tag{3.8}$$

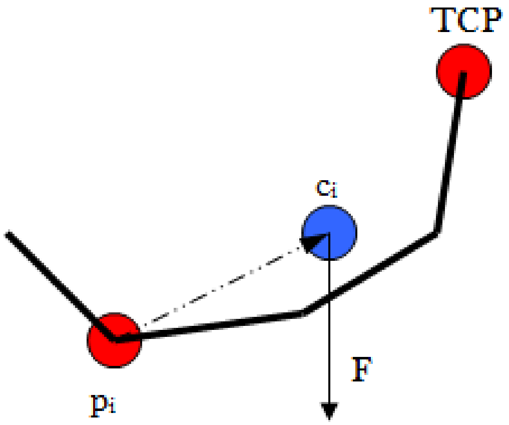


Figure 3.2: The torque in a joint $i$ generated by gravity on the center of mass $c_i$.

When differentiating with respect to the forces $f$, the first summation expression in (3.6) is equal to zero and by rewriting $l_k = q_k - P$ as the skewed matrix $\hat{l_k}$ therefore

$$\frac{\partial g_M}{\partial f_k} = \hat{l}_k. \tag{3.9}$$

This is a three times three matrix because each contact point force is dependent on three variables. Similarly, when differentiating with respect to $t_k$

$$\frac{\partial g_M}{\partial t_k} = I_3.$$

## 3.3 The Comfort Function

One of the goals of this thesis is to make ergonomical poses. Therefore a comfort function, $h$, is introduced which gives an ergonomical load that depends on the joint values $\Theta$ and moments, $h = h_\Theta + h_M$. In the IK algorithm the gradient $\frac{\partial h}{\partial x}$ is needed for finding a more ergonomical pose.

In [19] one ergonomical aspect is the joint angles; joint values that are small compared to joint value maximum are preferred. This is achieved by using the tangent function that can be made to approach infinity close to maximum

$$h_\Theta(\Theta) = -\sum_{i=0}^{n} \tan^2\left(\frac{\pi\Theta_i}{2\Theta_{max,i}}\right)$$

so

$$\frac{\partial h_\Theta}{\partial \Theta_i} = \frac{-\pi}{\Theta_{max,i}} \tan\left(\frac{\pi\Theta_i}{2\Theta_{max,i}}\right) / \cos^2\left(\frac{\pi\Theta_i}{2\Theta_{max,i}}\right).$$

We also want to reduce the strain from high moments in the joints. Here a joint can only be affected by moments in the plane normal to the rotation axis. We introduce $M_i(x)$ as the moment vector in joint $i$ and $\alpha_i$ is the rotation axis which are used in

$$h_M(x) = -\sum_{i=0}^{n} (\alpha_i \cdot M_i)^2. \tag{3.10}$$

To calculate $M_i$ two points are introduced: $p_i$ is the joint position and $q_j$ is the contact point of force $f_j$. With $D(i)$ as the set of contact forces and contact moments applied to the links referred to in $C(i)$

$$M_i(x) = \sum_{k \in C(i)} (c_k(\Theta) - p_i) \times m_k \hat{g} + \sum_{k \in D(i)} (q_k(\Theta) - p_i) \times f_k + \sum_{k \in D(i)} t_k. \tag{3.11}$$

By differentiation of (3.10)

$$\frac{\partial h_M}{\partial x_i} = -2 \sum_{j=0}^{n} (\alpha_j \cdot M_j) \left( \frac{\partial \alpha_j}{\partial x_i} \cdot M_j + \alpha_j \cdot \frac{\partial M_j}{\partial x_i} \right)$$

where $\frac{\partial \alpha_j}{\partial \Theta_i}$ is $\alpha_i$ if joint $i$ is revolute and $i \in C(j)$ otherwise $\mathbf{0}$. Furthermore, $\frac{\partial \alpha_j}{\partial f_i}$ and $\frac{\partial \alpha_j}{\partial t_i}$ are always $\mathbf{0}$. For convenience we differentiate (3.11) with respect to $\Theta, f$ and $t$ separately. When differentiating with respect to $\Theta$ the velocity of a point $b_j \in \mathbb{R}^3$ moving around an axis $\alpha_i$ in a point $p_i$ is

$$\frac{\partial b_j}{\partial \Theta_i} = \begin{cases} \alpha_i \times (b_j - p_i) & \text{if } j \in C(i) \\ \mathbf{0} & \text{otherwise} \end{cases}$$

which is described in detail in [4]. Using this and $j \in C(i)$ we get

$$\frac{\partial M_i}{\partial \Theta_j} = \sum_{k \in C(i)} \frac{\partial c_k}{\partial \Theta_j} \times m_k \hat{g} + \sum_{k \in D(i)} \frac{\partial q_k}{\partial \Theta_j} \times f_k$$

and when differentiating with respect to $f$ and $t$ we have

$$\frac{\partial M_i}{\partial f_k} = \begin{cases} \hat{q_k} - \hat{p_i} & \text{if } k \in D(i) \\ \mathbf{0} & \text{otherwise} \end{cases}$$

$$\frac{\partial M_i}{\partial t_k} = \begin{cases} \mathbf{1} & \text{if } k \in D(i) \\ \mathbf{0} & \text{otherwise} \end{cases}$$

Chapter 4

# Decreasing the Computational Cost

The computation time of the pose predictions is greatly influenced by the choice of the IK solver and the model. A global optimization algorithm is often computationally expensive compared to a local solver while it may find a better optimum. An increase of DOFs in the model will increase the computational cost of pose predictions independently of the solver. [4] chose to go with a Newton inspired method, Resolved Motion Rate (RMR), as the IK solver and there are a number of possible ways to reduce the amount of flops needed. It was concluded that a large part of the computation time of RMR was spent on the SVD. Changing the SVD to QR factorization as in [20] would introduce problems because of the ill-conditioning of the matrix when moving close to singularities. The SVD can be damped in a easy way called DLS (Damped Least Square) [21] without increasing the amount of flops significantly. Therefore, in this thesis we will continue using the SVD and improve the algorithm that would be of benefit for any decomposition or solver used in future. The number of DoFs will be reduced while still maintaining the flexibility of the manikin.

## 4.1 DoF Reduction

Some developers of digital humans decrease the number of joints to increase the speed of pose predictions e.g. by reducing the spine to just a few joints [9]. While this decreases the number of flops needed it is not of interest because the flexibility of the manikin should be maintained. The idea is to let a group of variables be controlled by fewer variables. Therefore groups of joints are created, like the spine's three parts, lumbar, thoracic and cervical, and also the shoulders. The number of flops needed for SVD is around $2n^2m - \frac{2}{3}n^3$ which increases almost linearly in the number of columns when the number of columns, m, is much larger than the number of rows, n. Therefore, a reduction of columns is of interest. Furthermore, there is no need to rebuild the model for this reduction because it can be added to the total model. Introducing the vector $\beta$ which contains variables that are related to $x$ by $x = B\beta$, where $B$ is a linear operator. Then the new constraint function is formulated as $\hat{g}(\beta) = g(x(\beta))$ and the new Jacobian $\hat{J}$ can be derived from $J$ by the chain rule

$$\tfrac{\partial \hat{g}}{\partial \beta} = \tfrac{\partial g}{\partial x}\tfrac{\partial x}{\partial \beta} = JB = \hat{J}$$

### 4.1.1 The Spine

The three parts of the spine are modeled separately and each part has three bend variables [22] $\beta_i$ where $i$ defines which axis the spine is bending around. Each $\beta_i$ is simply mapped to all corresponding values in $\Theta$. Here is an example of what the matrix $B$ can look like

$$B = \begin{pmatrix} I & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & I \end{pmatrix}$$

where $I$ denotes identity matrices, which maps the unchanged variables. The $B$ matrices are simple to create by derivation.

The lumbar has 15 joints and the thoracic has 36 joints while the cervical has 21 joints and after the reduction the total spine variables are reduced from 72 variables to 9, while the full model is reduced from 162 to 99 variables. This should reduce the flops for the SVD by a little more than a third.

### 4.1.2 The Shoulders

A shoulder is more complex than the spine in the sense of making it move naturally by a number of variables that defines all joint values. In this thesis the shoulder model contains nine joints and the goal is to show that it is possible to use the above stated DoF reduction method on the shoulders to speed up the IK solver.

As an example the relation formulas of [23] are used and below is an short description of the equations. Let the parameterization for rotations be $R(\gamma, \beta, \alpha) = R_x(\gamma)R_y(-\beta)R_z(\alpha)$ and the indices $c$, $s$ and $h$ stand for clavicle, scapula, and humerus, respectively. Furthermore, let $\alpha_h \in [-10, 90]$ and $\beta_h \in [-90, -30]$ be independent variables and the others calculated from these with the equations

$Humerus: \gamma_h = -45 + \alpha_h(1 - (\beta_h + 90)/360) + (135 - \alpha_h/1.1)\sin(0.5(\beta_h + 90)(1 + \alpha_h/90))$

$$Clavicle: \begin{cases} \alpha_c = -50 + 30\cos(0.75(\beta_h + 90)) \\ \beta_c = 24(1 - \cos(0.75(\beta_h + 90)))(0.5 + \alpha_h/90) + 9 \\ \gamma_c = 15(1 - \cos(0.75(\beta_h + 90))) + 3 \end{cases}$$

$$Scapula: \begin{cases} \alpha_s = 200 + 20\cos(0.75(\beta_h + 90)) \\ \beta_s = -140 + 94\cos(0.75(\beta_h + 90)(1 - \gamma_h/270)) \\ \gamma_s = 82 + 8\cos((\alpha_h + 10)\sin(0.75(\beta_h + 90))) \end{cases}$$

Worth to note is that $\gamma_h$ is the rotation of the upper arm while $\alpha_h$ and $\beta_h$ are lifting in sagittal and coronal plane. These formulas only give natural postures for a small proportion of all possible postures. The equations from [23] reduce each shoulder from nine to two DoFs but a more general set of equations may require more DoFs.

Chapter 5

# Implementation

The implementation has been done into the framework of IMMA that was developed in [4]. The model has been upgraded to handle the external root and its translation joints while the IK solver has been rewritten. A short review of the program's general structure will be given and after that the new IK solver will be described.

## 5.1 Programming Language and Libraries

There are many programming languages to choose from that would be suitable for creating a digital human. C++ is a versatile language and is used by FCC and was therefore chosen as the language used for implementing the framework of IMMA. In [4] we found no reason to change language and therefore C++ is also used while refining the manikin. Also, to save time, libraries were used for such things as basic math functions and array structures. The libraries were provided by FCC.

## 5.2 General Structure of the Program

The manikin program is built with the human biomechanical skeleton in mind. The foundation of the manikin is a class that represents the joints and the rigid bodies between them, called links. These links hold all the information that is local to the link and have no knowledge of its environment or the interrelationship between links. Then there is one class that handles all rules for connecting these links as a biomechanical model. There is also a pose prediction class that stands alone from the manikin and only uses the biomechanical class as a function of $\Theta$.

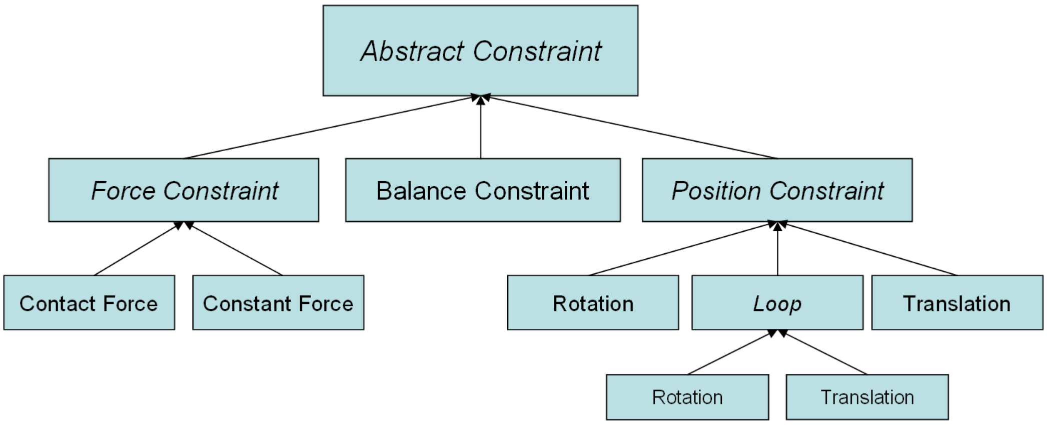


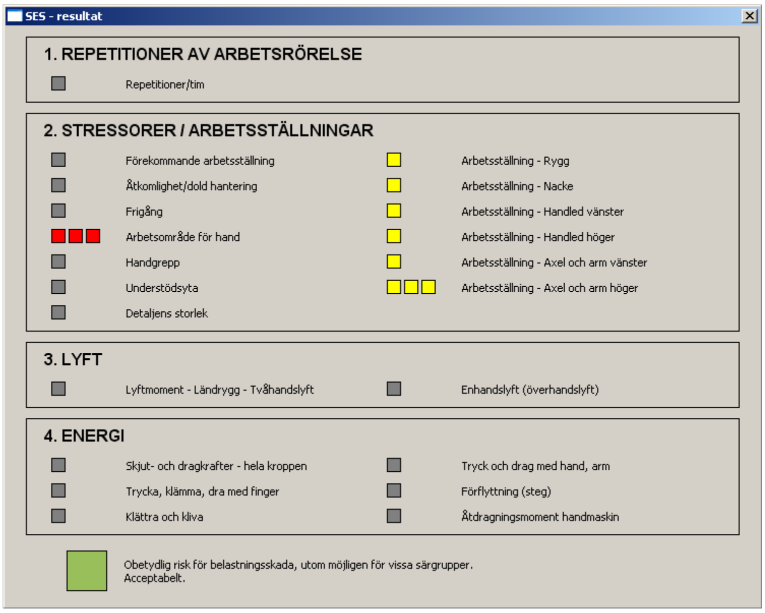
Figure 5.1: Example of a constraint hierarchy.

## 5.3 Pose Prediction

In the previous IK solver the DLS [4] solver class created the full Jacobian and the error vector before it solved the system. This made it very cumbersome to implement new constraints and give the users more options for each constraint. Furthermore, the complexity of the model class was increasing fast with the introduction of contact points, the force variables, the relaxed constraints and the reduction rules. These problems will be even more acute in the future when more constraints are introduced. As a new structure we mimic the generic optimization problem in section 2.2.4. All the constraints derive from an abstract parent class that defines the basic functions needed for all constraints. Important functions that are in common are

- A function to fill the part of the Jacobian that is reserved for the specific constraint.
- A function to fill the part of the error vector that is reserved for the specific constraint.
- A function to give the influence to the $\nabla h(x)$ vector.
- A function that returns the number of variables and constraint dimensions that are used by the constraint.

These are the main functions of all constraints even if more may be needed. From this parent class the inherited classes derive the structure and add the constraint specific code, se fig 5.1 as example.
The DLS solver that was implemented in [4] now only receive an array of these constraints and gets all model dependent information from them. Now when adding a new type of constraint, no change to the solver or model is needed.



SES - resultat

1. REPETITIONER AV ARBETSRÖRELSE

■ Repetitioner/tim

2. STRESSORER / ARBETSSTÄLLNINGAR

| | | | |
|---|---|---|---|
| ■ | Förekommande arbetsställning | 🟨 | Arbetsställning - Rygg |
| ■ | Åtkomlighet/dold hantering | 🟨 | Arbetsställning - Nacke |
| ■ | Frigång | 🟨 | Arbetsställning - Handled vänster |
| 🟥🟥🟥 | Arbetsområde för hand | 🟨 | Arbetsställning - Handled höger |
| ■ | Handgrepp | 🟨 | Arbetsställning - Axel och arm vänster |
| ■ | Understödsyta | 🟨🟨🟨 | Arbetsställning - Axel och arm höger |
| ■ | Detaljens storlek | | |

3. LYFT

| | | | |
|---|---|---|---|
| ■ | Lyftmoment - Ländrygg - Tvåhandslyft | ■ | Enhandslyft (överhandslyft) |

4. ENERGI

| | | | |
|---|---|---|---|
| ■ | Skjut- och dragkrafter - hela kroppen | ■ | Tryck och drag med hand, arm |
| ■ | Trycka, klämma, dra med finger | ■ | Förflyttning (steg) |
| ■ | Klättra och kliva | ■ | Åtdragningsmoment handmaskin |

🟩 Obetydlig risk för belastningsskada, utom möjligen för vissa särgrupper.
Acceptabelt.

Figure 5.2: A motion was created and its ergonomical value was evaluated by SES [24] and presented in Swedish as in the picture.

It is now also possible for the user to change the number of constraints or the relaxation of them on the fly.

## 5.4 Ergonomical Evaluation

Because the IMMA project has as a goal to improve the ergonomical evaluations in production we need a way to efficiently measure the ergonomical load on postures. In companies like Scania, Volvo and Saab, a mockup of the environment is constructed, where the workers can do the assembly operation of interest while being recorded. Then the motions are compared to a worksheet that usually include

- Bend on back
- Working area for the hand
- Shoulder pose
- Forces

and maybe more. Instead of letting the user of the manikin approximate all poses and filling in a worksheet, the computer can calculate all answers exactly on the constructed pose, see figure 5.2 for Scania's ergonomical standard (SES) [24] implemented.

# Chapter 6

# Results

In order to demonstrate the applicability of the method using an exterior root and a unified treatment of balance, contact forces, position constraints, reduction, and ergonomics, we present some test cases. In all cases we have used the same units: meters, radians, Newtons, and Newton meters, for the quantities length, angles, forces, and moments, respectively. Furthermore, the comfort function in our trials penalizes joint values near the limits and in the start pose all joint angles are zero, see figure 6.1.

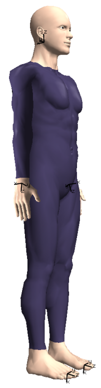


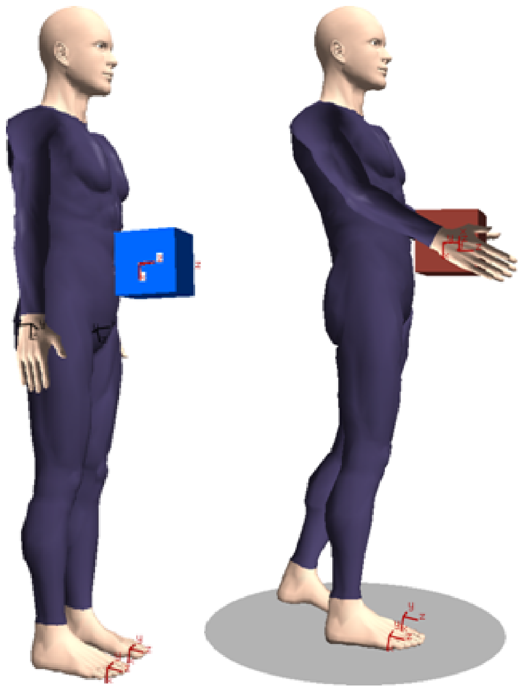
Figure 6.1: The starting pose of the manikin, all angles are set to zero.

## 6.1 Test Cases

### 6.1.1 Box Lifting

In the first test case we have added kinematic constraints to hands and feet. The hands should match the grasp positions on the box with five degrees of freedom locked - the palms are allowed to rotate around the surface normal. The feet are allowed to slide on the floor and rotate around the z-axis, orthogonal to the floor - the remaining three degrees of freedom are locked. The target placement can be seen in figure 6.2a. This setup results in 16 constraints for the link positions, and another six constraints to keep the body in balance. The final pose is shown in figure 6.2b. The hands are turned from the target frames and both feet have moved from initial positions, this without any need of re-rooting.

Since no link is completely locked, this test case would be impossible to solve using a fixed root and problematic to solve using a changing root. This case shows the strength of the framework for the unified solution presented in this



(a) The targets shown relative to the start pose.

(b) Solution pose.

Figure 6.2: The first test case. The hands and feet are partly constrained.

thesis.

The second case has the same start pose as the first case, but the box is positioned slightly higher, the target frame for the right foot is set at a different height, and the left foot is allowed to stand on its toes. See figure 6.3 for the final pose.

### 6.1.2 Randomized Target Testing

The third test case is to validate the functionality of the physics and the optimization of the implementation. This is done by taking randomized targets for the hands and then let the solver find feasible poses. The position targets for hands will be taken randomly in space with the restrictions that the hands will not cross each other and that they should not be further apart than that a restricting box can fit around them; this box is used in industry as the accepted workspace. We will use the same definition of the box as in [4] which was

- Upper bound at shoulder.

- Lower bound at knuckles when the arms are relaxed at the sides.

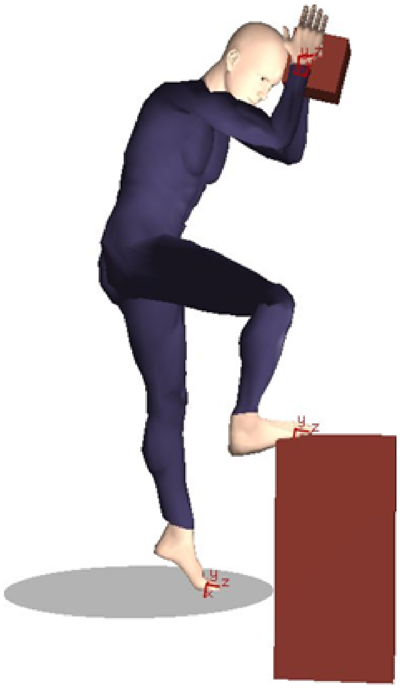


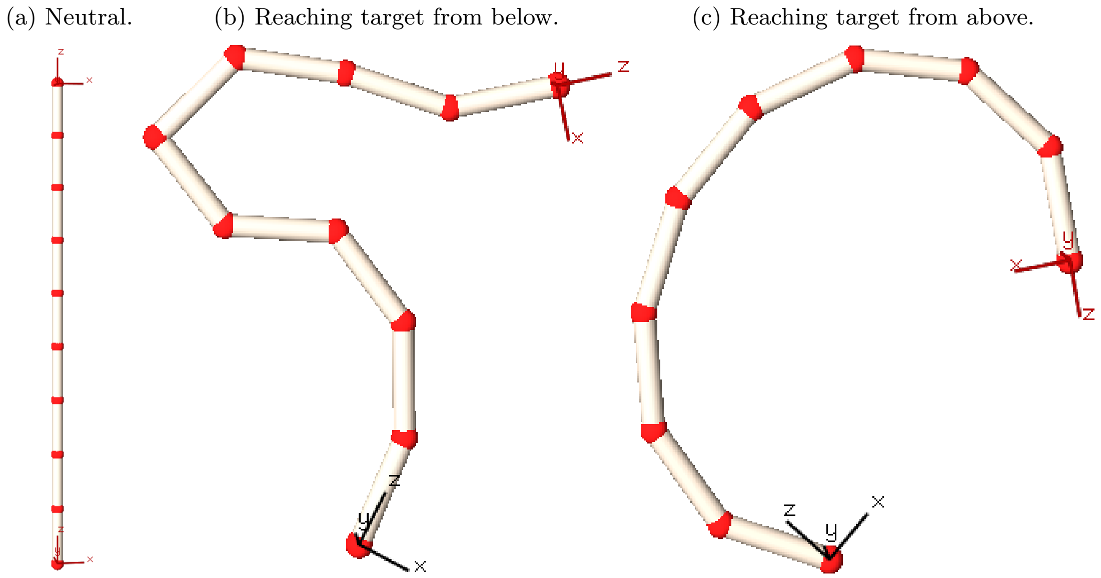
Figure 6.3: Second test case. Right foot is constrained to the box, otherwise the manikin is constrained as in test case one.

- Left and right bounds are at elbows when both arms are extended at maximum to the sides.
- Back bound at the body's vertical line.
- Front bound at the knuckles when the arms are extended at maximum, forward.

The manikin is free to move on the floor but the heels and toes need to be on the floor. Furthermore, the tests are done with a number of different max-deviations on the target rotation compared to neutral hand position, this because a high deviation increases the risk of unnatural positions. For each deviation level 1000 trials are done where success is only dependent on the constraint function. The results can be seen in tables 6.1, 6.2, and 6.3, which show the effect of the physics and the speed up.

## 6.2 Physics

To display the effect of the physics a simple snake model is also used. This is because it is easier to see whether the model is in balance or not when only one chain of rigid bodies affects the center of mass. The snake model consists of nine segments that can be rotated 90 degrees in any direction. Furthermore, it has


(a) Neutral.

(b) Reaching target from below.

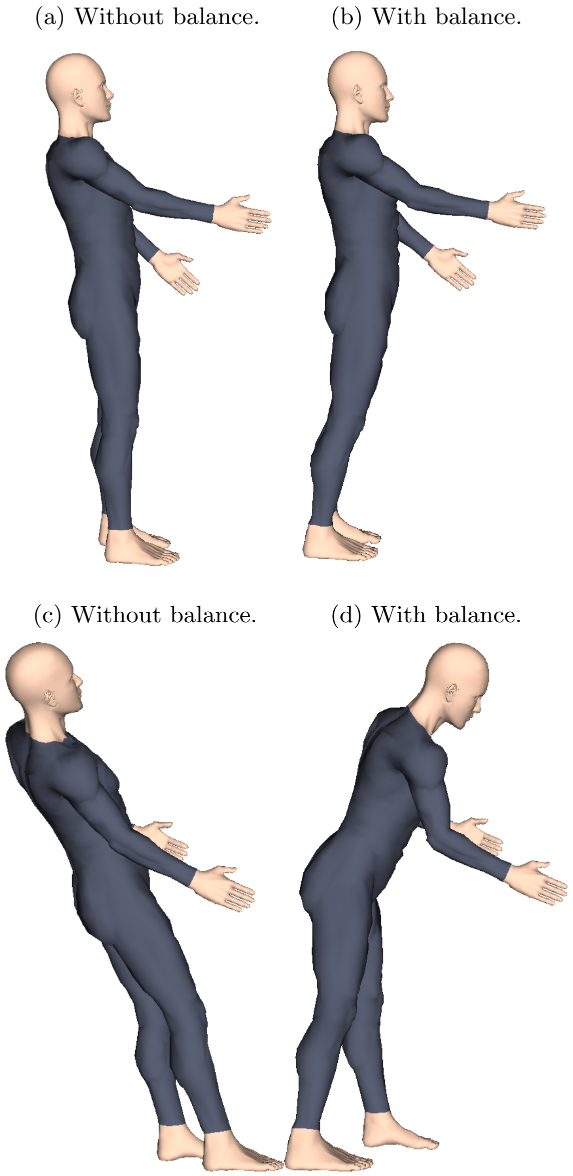
(c) Reaching target from above.

Figure 6.4: Different poses that are all in balance.

| Max Dev. | *Success* | Max t | Average t |
|---|---|---|---|
| $\pi/8$ | 919 | 1.067 | 0.333 |
| $\pi/4$ | 885 | 0.980 | 0.361 |
| $\pi/3$ | 889 | 1.030 | 0.370 |
| $\pi/2$ | 832 | 0.996 | 0.423 |
| $2\pi/3$ | 802 | 0.992 | 0.464 |
| $\pi$ | 744 | 0.997 | 0.532 |

Table 6.1: Success rates and solution times with physics and with reduction, based on 1000 trials.

one floor TCP that allows it to move freely on the floor and one top TCP that has a full six dimensional constraint to its target. The moments and forces shall be equal to zero for the model to have found balance. The starting pose and two goal poses can be seen in figure 6.4. These poses help to verify that the balance constraints are correct and also show that it can solve rather complex poses. Applying the physics constraints on the manikin results in much more realistic poses compared to the ones without the balance constraint, see figure 6.5. The balance constraint increase the number of rows in the Jacobian by six and the number of columns by three for each contact force. This increases the computation time but does not affect the success rate, this can be seen when comparing tables 6.1 and 6.2.



(a) Without balance. (b) With balance.

(c) Without balance. (d) With balance.

Figure 6.5: Change of the pose when balance is introduced.

| Max Dev. | *Success* | Max t | Average t |
|---|---|---|---|
| $\pi/8$ | 916 | 0.716 | 0.224 |
| $\pi/4$ | 898 | 0.715 | 0.236 |
| $\pi/3$ | 850 | 0.719 | 0.265 |
| $\pi/2$ | 834 | 0.820 | 0.285 |
| $2\pi/3$ | 806 | 0.733 | 0.319 |
| $\pi$ | 752 | 0.723 | 0.362 |

Table 6.2: Success rates and solution times without physics and with reduction, based on 1000 trials.

| Max Dev. | *Success* | Max t | Average t |
|---|---|---|---|
| $\pi/8$ | 856 | 1.841 | 0.834 |
| $\pi/4$ | 829 | 1.870 | 0.887 |
| $\pi/3$ | 829 | 1.840 | 0.893 |
| $\pi/2$ | 753 | 1.863 | 0.988 |
| $2\pi/3$ | 699 | 1.853 | 1.063 |
| $\pi$ | 612 | 1.852 | 1.213 |

Table 6.3: Success rates and solution times with physics and without reduction, based on 1000 trials.

## 6.3 Decreased Computational Cost by Variable Reduction

To investigate the impact of variable reduction, the spine was reduced to nine variables, the hands to one variable each for grasping, and each shoulder to two DoF. The decrease in time is a little more than 50% and also the success rate increased, see tables 6.1 and 6.3. The success rate increase may be because the system contains fewer local minima after variable reduction.

# Chapter 7

# Discussion

This work is a part of the IMMA project where the aim is to support ergonomical evaluation of operators in the manufacturing industry. The purpose of this master thesis is to improve on the manikin made in [4] by implementing basic physics and speed improvements. The manikin presented here has the possibility to be manipulated in Euclidean space instead of in the joint space. The manipulation of the manikin in Euclidean space is done by positioning a target frame for a TCP.

The position targets are parameterized by a point in SE(3) but can be relaxed and therefore constrain one to six DoF each. The poses that are generated by the IK solver are now in balance which improves the usability of the manikin greatly. The speed up method that was implemented, succeeded to reduce the time used for pose prediction by more than 50%. At the same time the improvements of the manikin model and implementation of physics almost doubled the computation time. This makes the computation times in this thesis similar to the ones in earlier work.

## 7.1 Future Work

In this thesis we improved the manikin by realizing most of the future goals set in [4], but still there are a lot of functionality that could be added. For the IMMA project, collision handling is an important step towards better IK solutions and is therefore a future goal. When collision handling is added the manikin should be connected to Industrial Path Solutions (IPS) to make use of their top end path planner. Also, to improve usability and come closer to our goal of non-expert software an automated contact point finder would be of great use.

# Bibliography

[1] Jack. url: http://www.plm.automation.siemens.com. [cited at p. 3]

[2] Ramsis. url: http://www.human-solutions.com. [cited at p. 3]

[3] Industrial path solutions. url: http://www.industrialpathsolutions.com. [cited at p. 3]

[4] Delfs. N. Manikin in time: Development of a virtual manikin with inverse kinematics and comfort. Master's thesis, University of Gothenburg, 2011. [cited at p. 3, 4, 6, 8, 9, 11, 12, 13, 15, 17, 21, 22, 26, 31]

[5] Delfs N.; Mårdberg P.; Gustafsson S. Robotik och inverse kinematik. bachelor thesis at chalmers, gothenburgs university. 2009. [cited at p. 4]

[6] Benitez. A; Huitzil. I. Interactive platform for kinematic control on virtual humans. *Electronics, Communications and Computer 2010*, 2010. [cited at p. 5, 9]

[7] Janzen. L; Metaxas. D. Recursive dynamics and optimal control techniques for human motion planning. *Computer Animation*, 1999. [cited at p. 5]

[8] Tom. I. E; Navasiolava. N. A. Two-level behavior control of virtual humans. *RTO-MP-088*, 2003. [cited at p. 5]

[9] Badler. I. N; Philips. B. C; et al. *Simulating Humans: Computer Graphics, Animation And Control.* Oxford University press, 1999. [cited at p. 6, 11, 12, 17]

[10] Baerlocher P. *Inverse Kinematics Techniques Of The Interactive Posture Control Of Articulated Figures.* PhD thesis, École polytechique fédérale de Lausanne, 2001. [cited at p. 6, 9, 12, 13]

[11] Qu S; Wei Y; et al. Pose synthesis of virtual character based on statistical learning. *CNMT*, 2009. [cited at p. 6]

[12] Bohlin. R; Delfs. N; et al. Unified solution of manikin physics and positioning - exterior root by introduction of extra parameters. *Digital Human Modeling*, 2011. [cited at p. 6]

[13] Westgaard. R. H; Aars. A. The effect of improved workplace design on the development of work-related musculo-skeletal illnesses. *Applied Ergonomics*, 16, 1985. [cited at p. 7]

[14] Rasmussen. J; Damsgaard. M; et al. Anybody - a software system for ergonomic optimization. *Proceedings of the Fifth World Congress on Structural and Multidisciplinary Optimization*, 2003. [cited at p. 7]

[15] McKerrow P.J. *Introduction To Robotics.* McKerrow, 1991. [cited at p. 9]

[16] Reed. P. M; Faraway. J; et al. The humosim ergonomics framework: A new approach to digital human simulation for ergonomic analysis. *Digital Human Modeling for Design and Engineering Conference*, 2006. [cited at p. 11]

[17] Boulic. R. Complex character positioning based on a compatible flow model of multiple supports. *IEEE Computer Society*, 3(3), 1997. [cited at p. 12]

[18] Yujiang. X; Hyun-Joon. C; Joo. H. Kim; Rajankumar. B et al. Predictive dynamics: An optimization-based novel approach for human motion simulation. *Structural and Multidisciplinary Optimization*, 41(3), 2009. [cited at p. 12]

[19] Lämkull D. *Computer Manikins In Evaluation Of Manual Assembly Tasks.* PhD thesis, Department of Product and Production Development Production System, Chalmers University of Technology, 2009. [cited at p. 14]

[20] Luciano. C; Banerjee. P. Avatar kinematics modeling for telecollaborative virtual environments. *Winter Simulation Conference*, 2000. [cited at p. 17]

[21] Samuel R. B; Jin-Su Kim. Selectively damped least squares for inverse kinematics. *Journal of Graphics Tools, vol. 10, no. 3*, 2004. [cited at p. 17]

[22] Monheit G; Badler NI. A kinematic model of the human spine and torso. *IEEE Computer Graphics And Applications*, 11(2):29–38, MAR 1991. [cited at p. 18]

[23] Hogfors C.; Peterson P.; et al. Biomechanical model of the human shoulder. *Journal of Biomechanics*, 24(8):699–709, 1991. [cited at p. 18, 19]

[24] Scania AB. Ses (scania ergonomistandard). 2008. [cited at p. 23]

Printed and Bound at
Department of Mathematical Sciences
Chalmers University of Technology and University of Gothenburg
2012