A joint-constraint model for human joints using signed distance-fields

Article *in* Multibody System Dynamics · August 2012

DOI: 10.1007/s11044-011-9296-1

CITATIONS
8

READS
183

3 authors:

Morten Engell-Nørregård
University of Copenhagen
5 PUBLICATIONS **26** CITATIONS
SEE PROFILE

Sarah Niebe
University of Copenhagen
9 PUBLICATIONS **66** CITATIONS
SEE PROFILE

Kenny Erleben
University of Copenhagen
63 PUBLICATIONS **514** CITATIONS
SEE PROFILE

Multibody Syst Dyn
DOI 10.1007/s11044-011-9296-1

A joint-constraint model for human joints using signed distance-fields

Morten Engell-Nørregård · Sarah Niebe · Kenny Erleben

Received: 10 May 2011 / Accepted: 16 December 2011

Abstract We present a local joint-constraint model for a single joint which is based on distance fields. Our model is fast, general, and well suited for modeling human joints. In this work, we take a geometric approach and model the geometry of the boundary of the feasible region, i.e., the boundary of all allowed poses. A region of feasible poses can be built by embedding motion captured data points in a signed distance field. The only assumption is that the feasible poses form a single connected set of angular values. We show how signed distance fields can be used to generate fast and general joint-constraint models for kinematic figures. Our model is compared to existing joint-constraint models, both in terms of generality and computational cost.

The presented method supports joint-constraints of up to three degrees of freedom and works well with sampled motion data. Our model can be extended to handle inter-joint dependencies, or joints with more than three degrees of freedom. The resolution of the joint-constraints can be tweaked individually for each degree of freedom, which can be used to optimize memory usage. We perform a comparative study of the key-properties of various joint-constraint models, as well as a performance study of our model compared to the fastest alternative, the box limit model. The study is performed on the shoulder joint, using a motion captured jumping motion as reference.

Keywords Biomechanic modelling · Range of motion · Joint-constraint · Inverse kinematics · Signed distance-fields

M. Engell-Nørregård (✉) · S. Niebe · K. Erleben
Copenhagen University, Universitetsparken 5, 2100 Copenhagen, Denmark
e-mail: mort@diku.dk

S. Niebe
e-mail: niebe@diku.dk

K. Erleben
e-mail: kenny@diku.dk

Published online: 11 January 2012

1 Introduction

When simulating articulated figures, one needs a model that describes the range of motion of the individual joints. An example of such a model is the inverse kinematics (IK) skeleton which is a hierarchy of bones where each bone is connected to a parent bone by a joint [1]. The relative coordinate transforms between connected bones are given by a set of joint parameters. The focus of this paper is the presentation of a novel accurate model for representing legal values of these joint parameters based on experimental kinematic data. Throughout this paper, we use an inverse kinematics procedure to present the joint constraint model, but the constraints could be combined with other procedures, e.g., dynamic simulation.

From a mathematical viewpoint, an IK skeleton is a hierarchy of homogeneous coordinate transformations, where each bone corresponds to a homogeneous transformation. The different joints of the skeleton require varying numbers of parameters for representing a given pose. For instance, the elbow joint only needs a single parameter describing the angle between the upper and lower arm. Joints such as the shoulder or hip joints, have a higher degree of freedom (DOF), and thus require more angle parameters. One could even use a translational joint parameter to model the sliding of the scapula on the thorax in the shoulder complex.

These joint parameters are not unbounded; they are constrained by a highly non-convex, continuous connected, and closed subset of the parameter space. We use the term *feasible pose* when all the joint parameters of a given pose are in the legal parameter space. Joint-constraints describe the boundaries of the feasible region of poses. Figure 1 illustrates the geometric complexity of the feasible regions we face within our application perspective. It

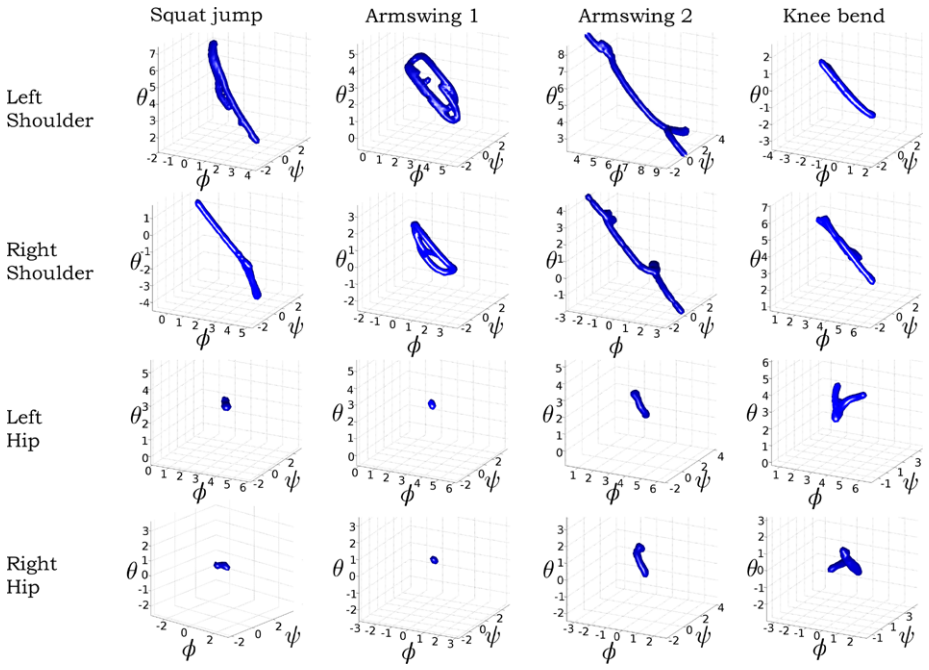


Fig. 1 Examples of joint-constraints for different joint types and different motions. *Each row* corresponds to one joint type and *each column* to one motion. Observe the complex geometry of the feasible region

shows joint-constraints sampled from several motions, found in the Carnegie Mellon University database of motion capture recordings [2].

The method we present is comparable to the method presented in [3]. However, the use of Euler angles and distance-fields results in a constant time projection operator, making our method a faster, more accurate, and attractive approach. Our model, the signed distance-field joint-constraint model, is generalized, supports highly nonconvex joint-constraints, has a simple geometric interpretation, and shows no performance problems that would prohibit use in interactive applications.

The work presented in this paper is highly motivated by tracking human poses using computer vision and machine learning based methods [4]. Within this application perspective, there is a need for a joint-constraint model that is computationally fast and gives an accurate description of the feasible region of a single specific human motion. An early version of this research, with an animation focus, was presented in [5].

2 Related work

Several researchers have investigated the range of motion of joints [6, 7], and numerous models exist, e.g. [3, 8, 9]. The box-limit model is the prevalent model, used in motion file formats such as Acclaim's asf/amc [10]. Animation tools such as Maya [11] and Blender [12] also use the box-limit model. With box-limits, the individual parameters are bounded within a minimum and maximum of allowed values, thus the feasible region forms a product of independent intervals. As this is a linear model with a convex feasible region, the box-limit model is easy to combine with optimization methods [13]. However, there are two major drawbacks of box-limits: They tend to result in a loose fitting boundary and they fail to include interdependencies between joint parameters. In practice, these short-comings are handled by tweaking the box-limits for specific motions, thus for a running motion one set of values would be used whereas, for a jumping motion, another set of joint parameters would be used. Despite these shortcomings, the box-limit model is widely used. As shown in previous work, box-limits can be determined automatically from motion capture data [4, 14]. The alternatives to the box-limit model offer more descriptive models, at the cost of increased complexity. The signed distance-field model, presented in this paper, takes a geometric approach. For each joint, we model the geometry of the boundary of the feasible region of the joint parameters. Our model is local, in the sense that we only model joint parameter dependencies for each individual joint. The model requires a two-manifold feasible region, so it can be embedded as the zero level-set of a signed distance-field, allowing us to handle any nonconvex joint-constraint regions. In this paper, we have used the shoulder joint as the primary test case because it exhibits some of the more complex behavior of joint-constraints, and thereby stresses the joint-constraint model used. The legal parameter space of the human shoulder joint is bounded by a nonconvex nonlinear joint-constraint [8, 15].

The authors of [8] use a sinus-cone model from [16], a human shoulder is modeled by a hierarchical IK skeleton. The scapula-thoracic joint is modeled by breaking the closed chain and using the scapula as an end-effector constrained to the surface of an ellipsoidal thorax. The sinus-cone model is more general than the box-limit model. A reach-cone model based on an idea from Korein [17] is presented by Wilhelms and Gelder [9]. This is extended in [18] where a general joint component framework is described. In [19], a spline based implicit joint model is suggested for multibody dynamics. Due to the implicit nature of the model, the geometry of the boundary of the feasible region can be modeled as box-limits in the spline parameter space, interdependency of parameters is omitted. Shoulder

Table 1 A comparison study of key-properties of various joint-constraint models

Model	Model representation	Data representation	Projection operator
Box-Limits	Explicit	Vector of intervals	Constant time projection
Sinus-Cones	Implicit	Implicit function	Root-search problem, Iterative Newton method
Reach-Cones	Explicit	Set of connected tetrahedra	Linear search for closest tetrahedron
Spline Joints	Para-metric	B-splines	Not available
Quaternion Boundary Fields	Implicit	Radial basis functions (RBF)	Root-search problem, Iterative Newton method
Signed Distance-Field	Implicit	Signed distance-field	Constant gradient projection

Model	Feasibility test operator	Model capabilities
Box-Limits	Constant time verification of enclosing interval	Convex/Boxed
Sinus-Cones	Evaluation of closed form solution	Convex/Ellipsoid
Reach-Cones	Linear search for enclosing tetrahedron	Nonconvex, but no holes
Spline Joints	Infeasible poses are not allowed	General nonconvex
Quaternion Boundary Fields	Global support of RBF convolution of all samples	General nonconvex, difficulties with holes
Signed Distance-Field	Constant time lookup of distance value	General nonconvex

joint-constraints are modeled in [3] using quaternion field boundaries. The orientation of the shoulder joint is sampled from motion capture data in quaternion space. Radial basis-functions are used to reconstruct an implicit surface representation of the boundary of the feasible quaternions. Our approach has similarities with this method, as it uses an implicit surface and supports general nonconvex joint-constraints. However, our projection operator is superior as it is a constant time operation, i.e., it is not an iterative procedure.

The approach of [3] is further developed in [20] where a hierarchical model of joints is presented. It seems that this approach can only handle a single parent hierarchy as there is no information about deeper hierarchies. In both papers, the main focus is on the machine learning part of building the joint-constraints, while our focus is on the joint-constraint model itself.

Table 1 is a summary of a comparison study between key-properties of the above mentioned methods and our signed distance-field (SDF) joint-constraint model. All the compared models can essentially be seen as geometric models of the boundary of the feasible region. Their differences lie mostly in their choice of geometric model for representing the boundary and in the actual computational representation. Finally, there are some differences in the technicalities of how the back-projection operator and feasibility tests are supported.

As Table 1 shows, the SDF model offers more generality than its alternatives, while supporting constant time operations.

Projection operators fall in two categories: constant time operation or iterative search schemes. One major feature of our SDF model is that the projection is a constant time operation. The only other model that can offer this is the box-limit model. On the other hand, the memory usage can be high. In our work, each joint needs $I \times J \times K$ cells of a map, storing a regular sampling of the distance-field, where I, J, K denote the resolution of

the map along the three axes. Adaptive distance-maps could be used in place of a regular sampling. However, our results show that in most practical cases, coarse maps can be used and so, the memory usage can be kept at an acceptable level.

The predominant trend in previous work in this research area is to consider only local joint-constraints models, the exception being [18] where the global dependency of joint parameters is considered in the case of forward kinematics. In this respect, our work is no different; our model is also a local model. However, we do consider the full dependency between joint parameters within a single joint. Neither box-limits, sinus-cones nor reach-cones offer this.

One final aspect of joint-constraints, on which we will elaborate below, is the ease with which one can set up the model. In machine learning, this is termed model selection and could be approached as a nonlinear regression problem [21]. Some models can employ sampled feasible poses for setting up the joint-constraints, the SDF model is one such model. Quaternion boundary fields share similarities with our approach in this regard, although their coordinate basis (imaginary part of a quaternion) is nontrivial to work with. To our knowledge, no prior work addresses model selection of sinus-cones or reach-cones.

The presented SDF model offers full modeling generality of local joint-constraints with constant time operations and easy model selection. This makes the presented model a novel method for obtaining more accurate joint-constraints, highly suitable for motion simulation of articulated figures.

3 The signed distance-field model

Even though the model presented here is a local model, it could be extended to cover inter-joint dependencies. Our base assumption is that the boundary of the feasible parameter space of a joint forms a single connected component. This implies that feasible joint motion is a connected subset. Thus, the test for feasibility is reduced to a simple inside/outside test. Thus, in case of an infeasible configuration, the point is projected unto the closest feasible point.

We use Z-Y-Z Euler angles as basis, where the orientation is specified by the angles $\mathbf{p} = [\phi\ \psi\ \theta]^T$. This allows us to work in a 3D space rather than 4D or 9D as would be the case for quaternions and homogeneous coordinates. By modeling the motion range geometrically, we have a broader basis of well-known geometrical representations to choose from, e.g., polygonal meshes, tetrahedral meshes, etc.

As performance is of great importance, the geometric representation must support two constant time operations: Verification of a feasible joint pose and projection of infeasible joint poses back onto the boundary of the feasible region. Distance-fields are known to offer both these qualities, but at the cost of extra memory usage. A distance-field is an implicit representation of the geometry, defined by the function $\Phi : R^3 \mapsto R$ where $|\nabla\Phi(\mathbf{p})| = 1$ everywhere and $\Phi(\mathbf{p}) = 0$ for all $\mathbf{p}$ corresponding to the geometry.

3.1 Building the signed distance-map

In the following, we describe how we compute the discrete signed distance-map from the continuous signed distance-field. In principle, any signed distance-field algorithm could be used [22]. For this paper, we used a simple brute-force approach. Because the signed distance-field rigging is done as a preprocessing step, the cost of generating the signed distance-map is of minor importance. In fact, we did not even bother storing the preprocessing for our tests. Our experience shows that a fairly coarse resolution is sufficient, as the

set of motions we study only requires a low number of temporal samples. Running this process off-line means that generating the distance-map is not a bottleneck, as might have been suspected. Table 2 in Sect. 4.1 lists runtime statistics for generating the signed distance-maps of a jumping motion.

Although we use a simple brute-force method in constructing the distance-map, we will—for completeness of presentation—give a full detail description. Initially, we create a regular grid of a user specified resolution. The grid is located in space such that the minimum and maximum corner points of the grid is within the angle interval bounds of the Euler angles:

$$\begin{bmatrix} \phi \\ \psi \\ \theta \end{bmatrix} \in \begin{bmatrix} 0, 2\pi \\ 0, 2\pi \\ 0, 2\pi \end{bmatrix}. \tag{1}$$

Once the grid has been created all distance values stored at the grid nodes are initially set to ∞.

$$\Phi(\mathbf{p}) = \infty \quad \forall \mathbf{p} \in \mathcal{N}, \tag{2}$$

where $\mathcal{N}$ is the set of all grid nodal positions in the map.

1. Next, we sample the joint motion from some exemplar based motion, $\{q^i\}_{i=1}^{i=N}$, for instance some motion capture data.
2. For each time sample q^i we extract the Euler angles of the present joint

$$\mathbf{p}^i = \begin{bmatrix} \phi^i \\ \psi^i \\ \theta^i \end{bmatrix} \leftarrow q^i. \tag{3}$$

   The Euler angles are collected in a chronologically ordered list $\{\mathbf{p}^i\}_{i=1}^{i=N}$.
3. For each node of the grid, we compute the distance to the closest sample point. Running through the entire list of samples we check the distance to the current sample. If the newly computed distance is less than the distance value currently stored in the corner node, then we replace the stored value with the new value. This gives the following selection operation:

$$\Phi(\mathbf{p}) \leftarrow \min_{i=1..N} \left\{ \Phi(\mathbf{p}), |\mathbf{p}^i - \mathbf{p}| \right\} \quad \forall \mathbf{p} \in \mathcal{N}. \tag{4}$$

The temporal sampling of the motion might be too coarse, thereby creating multiple disjoint regions in the final signed distance-map. There are at least two ways of dealing with the sampling problem. One may adopt a naive approach: Detect if the problem occurs and then redo the entire motion sampling at a finer resolution. This could be done by detecting how many connected components we have in the final map. However, a more intelligent approach is to use an adaptive sampling: Given a grid resolution, we can adaptively modify the motion capture sampling rate to ensure that the distance between any two consecutive motion samples is never larger than half the maximum grid spacing. We use the adaptive motion sampling strategy in our implementation.

3.2 Adaptive sampling of motion

We use a chronologically ordered list of our samples to interpolate between neighboring samples. Coarsely sampled regions are subsampled using spherical linear interpolation

$n = 0$
$t = 0$
$p^n \leftarrow q^n \leftarrow$ **current sample at time** t
for $n = 1$ **to end**
  $q^{n+1} \leftarrow$ **next sample at time** $t + \Delta t$
  while *distance*$(q^n, q^n + 1) >$ **grid spacing**$/2$
    $\Delta t \leftarrow \Delta t/2$
    $q^{n+1} \leftarrow$ **next sample at time** $t + \Delta t$
  end
    if (*distance to small*) **increase** Δt
  $n \leftarrow n + 1$
end

Fig. 2 Adaptive motion sample algorithm. Observe that the Δt variable can be seen as a kind of trust region radius

(slerp [23]) in angle space. We can subsample the region between two neighboring samples as densely as necessary, to ensure that the distance between the new samples are never greater than half the grid resolution. The resulting quaternions are then converted into Euler angles and added to the list between the existing samples.

Figure 2 summarizes the adaptive sampling algorithm. If disjoint motion is encountered, some interpolation scheme must be established to connect the manifolds of the unconnected motions. For testing purposes, we simply interpolate between the last frame of one animation with the first frame of the second animation. For the test sets used, this simple approach works well. It should be noted that this would not generally be sufficient. It should be ensured that the motion samples are not disjoint, meaning the motions should share at least one common point.

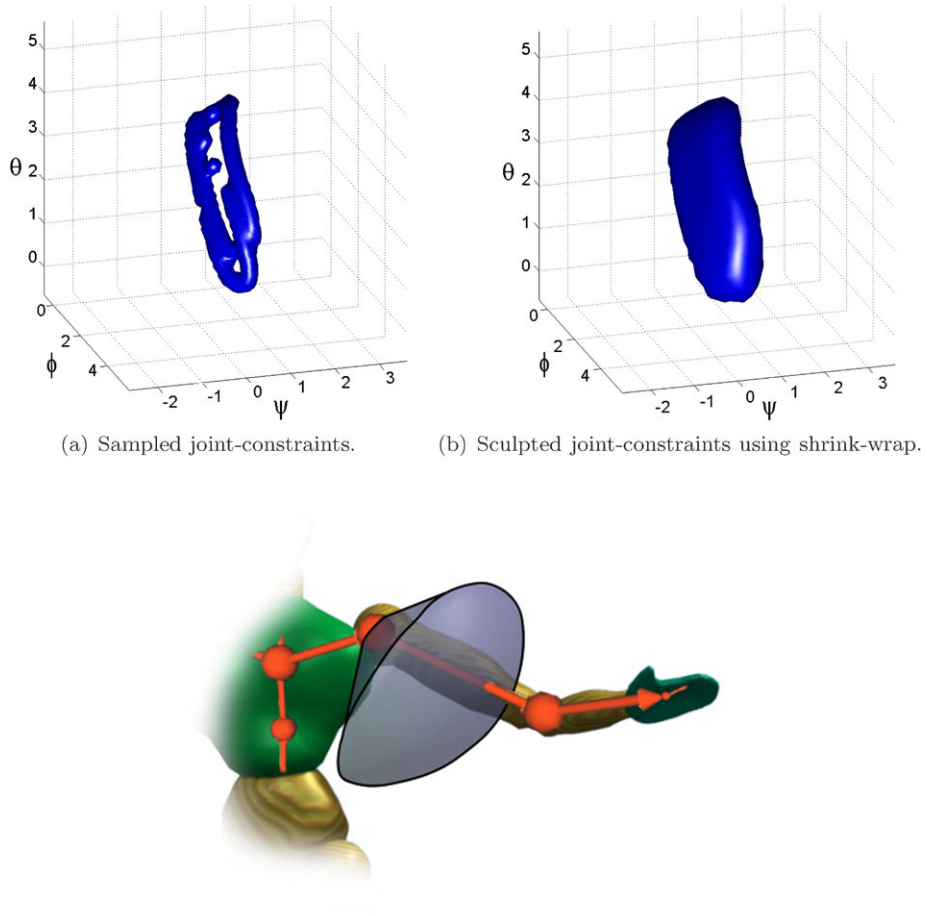
3.3 The signed distance property

We know that all motion samples are feasible poses and we can assume with some certainty that not all feasible poses are represented. Further, we know that some poses are in the interior of the region and some are on the boundary. In some cases, we want to treat all samples as if they were lying on the boundary. This would be the case when we have a very restricted motion from which we have built our signed distance-field, and we want to use these to make it possible to mimic the specific motion from which the samples have been taken. Thus, one may wish to add an additional pass to the distance-map generation algorithm. We want to implicitly add the interior void to our representation. The above part of the algorithm only represents the samples as the feasible position and has no notion of what the interior is. To add such a notion, one may convert the distance-map into a signed distance-map. The idea is to place a positive sign on some outside border cell of the grid. Then one simply performs a region filling operation where all neighboring cells not crossing the zero level-set is given a plus sign. In the end, all unassigned cells must be interior and are given a minus sign. One flaw in this approach is that the region must be a closed manifold, otherwise it will be hollowed out and there will be no interior region assigned. This can be helped by either ensuring that the motion is sufficiently dense sampled or by using an alternative method for determining boundary cells. Ensuring the density of the samples has shown to be difficult [15], and for this application the extra work is not needed. The single motion sampling represents a very limited area. In this setting, the above mentioned subsampling is sufficient

to ensure the signed property of the map. For sculpted joint-constraints on the other hand, the signed distance-map property must be handled as we will show in the next section.

3.4 Sculpted signed distance-field joint-constraints

It may be difficult to obtain enough motion samples to automatically generate signed distance-fields. Also, one may wish for the option to have specific control over the motion and directly model the behavior. Using a geometric representation, such as distance-fields, allows for an easy and direct approach for modeling signed distance-fields, for instance by using sculpting tools [24]. As a proof-of-concept to support our claim, we used a simple setup exporting the surface of a signed distance-field as a polygonal mesh and then used the open source 3D modeling program Blender [12] to tightly fit a two-manifold to the data, using an enclosing sphere and the shrink-wrap modifier. An example of a feasible region obtained in this way is shown in Fig. 3(b). An advantage of this approach is that a noncon-



(a) Sampled joint-constraints.

(b) Sculpted joint-constraints using shrink-wrap.

(c) Sculpted joint-constraints shown in Cartesian space.

Fig. 3 An example of refining the sampled joint-constraints of the left shoulder using an enclosing sphere and a simple shrink-wrap modifier from Blender [12]

vex manifold can be easily obtained and is easily modifiable, either by directly manipulating the mesh or by manipulation through some intuitive interface such as modifying the feasible region as a polygon on a sphere. Figure 6 shows the result of applying the sculpted signed distance-fields.

The shrink wrap method used in this example is a way of filling out the empty parts of the feasible region. For completeness, we will give a brief description of the way this modifier works. For a full description, we refer to Blender documentation [12].

1. Choose an enclosing two-manifold object, e.g., a sphere.
2. Project all points of the enclosing object inward onto the surface of the target object.
3. Generate a new signed distance-field from this object.

Note that the original shape of the two-manifold object has a significant impact on the quality of the resulting joint-constraints. We have chosen this approach as an easy-to-use and intuitive approach, to show the possibilities of signed distance-field joint-constraints. The method does not guarantee physical correctness of the constraints. Indeed it is possible to make joint constraints which are not conforming with physical reality using this method. Other more physically founded and elaborate techniques for constructing a surface from a point cloud, such as the one described in [15], may give more correct results. Either technique could be used together with our method.

3.5 Applying signed distance-field joint-constraints for inverse kinematics

We use an inverse kinematic (IK) modelling approach similar to [1], where the IK problem is formulated as an optimization problem which is solved using an iterative line-search method. The feasibility of a given IK iterate, $\mathbf{p}$, is determined by testing the corresponding Euler angle distance value in the signed distance-fields:

$$\Phi(\mathbf{p}) \leq \varepsilon \quad \Rightarrow \quad \mathbf{p} \quad \text{is feasible.} \tag{5}$$

The parameter ε is a threshold value to counter round-off errors. The impact of this parameter is analyzed in our results section. The signed distance-field lookup is performed in constant time by indexing the surrounding grid points and performing a trilinear interpolation at the lookup position. Infeasible poses are projected back onto the feasible region, by moving in the opposite direction of the distance-map gradient. The gradient is computed as a central finite difference approximation:

$$\nabla\Phi_{i,j,k} = \begin{bmatrix} \frac{\Phi_{i+1,j,k} - \Phi_{i-1,j,k}}{2\Delta i} \\ \frac{\Phi_{i,j+1,k} - \Phi_{i,j-1,k}}{2\Delta j} \\ \frac{\Phi_{i,j,k+1} - \Phi_{i,j,k-1}}{2\Delta k} \end{bmatrix}, \tag{6}$$

where $\Delta i, \Delta j, \Delta k$ denote the grid spacing along each coordinate axis. The gradients at the grid nodes surrounding $\mathbf{p}$ are interpolated using a trilinear interpolation on each component of the gradient. The projection of the infeasible $\mathbf{p}$ is then

$$\mathbf{p} \leftarrow \mathbf{p} - \Phi(\mathbf{p})\nabla\Phi(\mathbf{p}). \tag{7}$$

The central difference approximation of the gradient may cause numerical dissipation in the computation of $\Phi(\mathbf{p})$ and $\nabla\Phi(\mathbf{p})$. To alleviate this, the projection (7) can be applied twice; this does not make the procedure iterative. It is merely an implementation safeguard against numerical dissipation. Due to the distance-map properties, no more than two iterations are needed; this gives a constant time operation.

Table 2 Timings of brute-force approach for distance-map generation for 30 joints in a jumping motion sample. Observe that even for detailed maps, the processing time is acceptable as this is a preprocessing step

Grid resolution	# Motion samples	Storage requirement	Computing time (secs.)
$8 \times 8 \times 8$	200	4 kB	0.21
$16 \times 16 \times 16$	200	32.7 kB	1.34
$32 \times 32 \times 32$	200	262 kB	8.30

4 Results and discussion

We have chosen to verify and validate the signed distance-fields in the context of IK. We use the box-limit model as a base of reference, mostly due to its widespread use in interactive application and because it is the model currently used in human motion tracking. We have excluded the alternative methods from the tests, since none of them live up to both the time and modeling demands of our application. We focus on performance and accuracy.

4.1 Building the signed distance-maps

In our tests, a brute-force approach was used for building the distance-maps from motion samples. Table 2 shows the timings for generating the signed distance-maps for the joint-constraints of a single jumping motion.

4.2 Parameterizing the signed distance-field joint-constraints

The signed distance-field model is dependent on the user specified grid resolution $I \times J \times K$ and the ε parameter. The parameters are orthogonal in the sense that grid resolution mostly influences the signed distance-field generation, while the feasibility threshold parameter is a run-time only parameter. We have tested different grid resolutions and it turned out that for our single motion sampled signed distance-fields there seemed to be a threshold around a resolution of $16 \times 16 \times 16$. For finer resolutions, the animations ran smoothly and with acceptable quality. For coarser resolutions, the generated motions were jagged and tended to get stuck.

For sculpted signed distance-fields, which tend to be more connected (see Fig. 3(b)), the resolution could be set much lower. Resolutions of $8 \times 8 \times 8$ or lower were acceptable in this case.

For the single motion case, the ε parameter needs to be large enough to ensure the existence of a solution for all poses, yet small enough to counter loose constraints. We observed that for values above 0.2 radians, the constraints are too loosely fitted, and for values below 0.2 radians the motion is jittery and the IK solver tends to stall.

4.3 Constant time performance

We have measured the computational time in an application, where the end-effectors of an IK solver are driven by the corresponding end-effector positions in a motion capture example. The measurements are performed on the IK solution. Figure 4 shows a plot of our measurements.

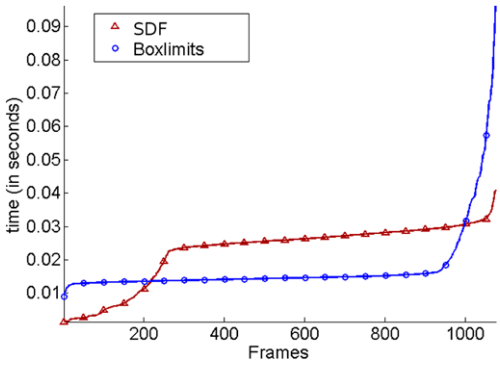
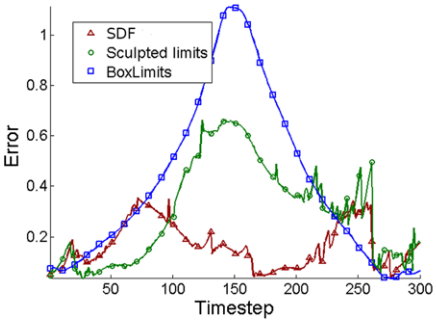
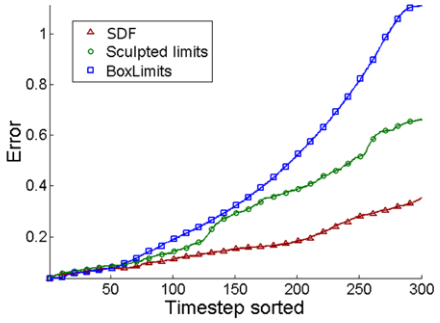


Fig. 4 Computation times of box-limits versus signed distance-fields when used in inverse kinematics. The measurements are ordered by value to make comparison easier. Notice that the box-limits are generally faster but cannot guarantee the same upper bound on computation time as the signed distance-fields



(a) Plot of error compared to motion capture.



(b) Accumulated error over time.

Fig. 5 Plots showing the difference between the right elbow's global position using the different joint-constraints. The more restrictive joint-constraints cancel out the effect of the redundancy of the joint, thus getting a solution closer to the motion capture reference. 1 unit on the axis corresponds roughly to 20 cm in real world measurements

Our experiments show that the signed distance-fields are slower than the box-limits. This was to be expected. We can also see that the worst case performance of the signed distance-fields is much better than worst case for the box-limits. In the worst case, the signed distance-fields still achieve approximately 20 fps, while the box-limits only reach approximately 10 fps.

4.4 Increased modelling accuracy

Using the more constrained signed distance-field model, we expect a gain in accuracy as this should reduce the redundancy of the simulation. The test uses an elbow joint; this joint is a child joint of the shoulder which has a wide rang of motion. We measured the deviation of the joint, using both sampled, sculpted, and box-limits, shown in Fig. 5(a). As expected, the results show that the box-limits do not constrain the position very much. The box-limits result in an error which, transformed into real world measures, corresponds to an error of 20 cm. The simulated motion is shown in Fig. 6.

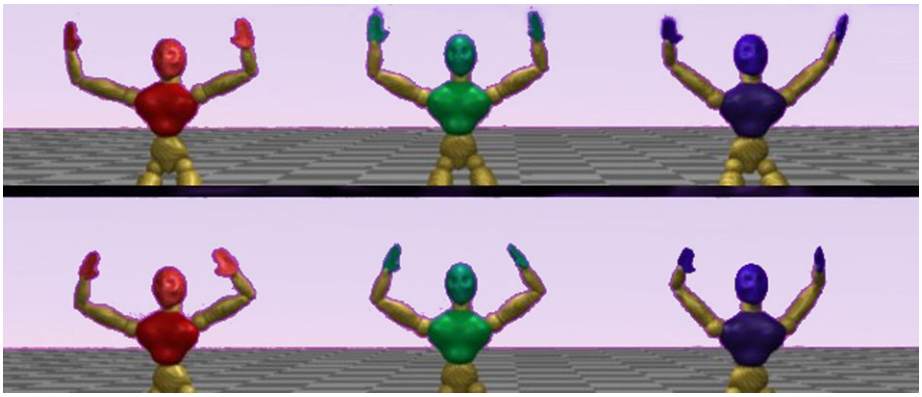


Fig. 6 Comparison study of animation quality when using box-limits versus signed distance-fields. The figure on the *left* shows signed distance-fields, the figure on the *right* shows box-limits. The figure in the *center* is the motion capture reference

4.5 Limitations

The main limitation of the signed distance-fields model is the memory consumption. The expressiveness of the joint constraints is linked to the resolution of the signed distance-map. This again is linked to the amount of memory used by the underlying data structure. For most practical purposes, this is not a problem but for application with very high demands for precision it may slow down the preprocessing step.

Another limitation is the fact that the method for building the signed distance-field relies on a large number of sampled motion points. In this work, we have not focused on the challenge of reconstructing joint constraints from inadequate information, as this is a common condition for all methods and not specifically linked to our method.

5 Conclusion

We have presented a novel joint-constraint model using signed distance-fields. The model supports general non-convex joint-constraints of 3 degrees of freedom and works well with sampled motion data. The memory usage is cubic in the resolution of the mesh. However, in most cases it is possible to get by with a comparably low resolution, in which case the memory usage is acceptable. The assumption of locality has shown to be insufficient, e.g., the orientation of the hip joint indeed has an effect on the joint-constraints of the knee. Therefore, an interesting venue for further research would be to extend our model to handle more than 3 degrees of freedom. The fitting of our model, although loose on account of the threshold, is still the tightest fitting model, compared with the other presented models as seen in Fig. 1.

References

1. Zhao, J., Badler, N.I.: Inverse kinematics positioning using nonlinear programming for highly articulated figures. ACM Trans. Graph. **13**(4), 313–336 (1994)
2. CMU Graphics Lab Motion Capture Database, http://mocap.cs.cmu.edu/. Carnegie Mellon University (2009)

3. Herda, L., Urtasun, R., Hanson, A., Fua, P.: Automatic determination of shoulder joint limits using quaternion field boundaries. Int. J. Robot. Res. **22**(6), 419–434 (2003)
4. Hauberg, S., Lapuyade, J., Engell-Nørregård, M., Erleben, K., Steenstrup Pedersen, K.: Three Dimensional Monocular Human Motion Analysis in End-effector Space. Lecture Notes in Computer Science, pp. 235–248. Springer, Berlin (2009)
5. Engell-Nørregård, M., Niebe, S., Erleben, K.: Local joint-limits using distance field cones in Euler angle space
6. Kim, J., Yang, J., Abdel-Malek, K.: A novel formulation for determining joint constraint loads during optimal dynamic motion of redundant manipulators in DH representation. Multibody Syst. Dyn. **19**, 427–451 (2008). doi:10.1007/s11044-007-9100-4
7. Sugiyama, H., Yamashita, H.: Spatial joint constraints for the absolute nodal coordinate formulation using the non-generalized intermediate coordinates. Multibody Syst. Dyn. **26**, 15–36 (2011). doi:10.1007/s11044-010-9236-5
8. Maurel, W., Thalmann, D.: Human shoulder modeling including scapulo-thoracic constraint and joint sinus cones. Comput. Graph. **24**(2), 203–218 (2000)
9. Wilhelms, J., Van Gelder, A.: Fast and easy reach-cone joint limits. J. Graph. Tools **6**(2), 27–41 (2001)
10. Darwin 3d. Acclaim fileformat. http://www.darwin3d.com/gamedev/
11. Autodesk. Autodesk. http://usa.autodesk.com
12. Blender. The blender foundation. http://www.blender.org
13. Engell-Nørregård, M., Erleben, K.: A projected back-tracking line-search for constrained interactive inverse kinematics. Comput. Graph. **35**(2), 288–298 (2011)
14. Engell-Nørregård, M., Erleben, K.: Estimation of joint types and joint limits from motion capture data. In: WSCG 2009: 17-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, p. 17 (2009)
15. Herda, L., Urtasun, R., Fua, P., Hanson, A.J.: Automatic determination of shoulder joint limits using quaternion field boundaries. Int. J. Robot. Res. **22**(6), 419–438 (2003)
16. Erkan Engin, A., Turgut Tümer, S.: Three-dimensional kinematic modelling of the human shoulder complex—part i: Physical model and determination of joint sinus cones. J. Biomech. Eng. **111**(2), 107–112 (1989)
17. Korein, J.U.: A Geometric Investigation of Reach. MIT Press, Cambridge (1985)
18. Shao, W., Ng-Thow-Hing, V.: A general joint component framework for realistic articulation in human characters. In: I3D '03: Proceedings of the 2003 Symposium on Interactive 3D Graphics, pp. 11–18. ACM, New York (2003)
19. Lee, S.-H., Terzopoulos, D.: Spline joints for multibody dynamics. ACM Trans. Graph. **27**(3), 1–8 (2008)
20. Herda, L., Urtasun, R., Fua, P.: Hierarchical implicit surface joint limits for human body tracking. Comput. Vis. Image Underst. **99**(2), 189–209 (2005)
21. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer Series in Operations Research. Springer, New York (1999)
22. Bærentzen, A.: Robust generation of signed distance fields from triangle meshes. In: Fellner, D., Moller, T., Spencer, S. (eds.) Fourth International Workshop on Volume Graphics, pp. 167–239 (2005)
23. Dam, E., Koch, M., Lillholm, M.: Quaternions, interpolation and animation. Technical report, University of Copenhagen (1998)
24. Bærentzen, J.A., Christensen, N.J.: Interactive modelling of shapes using the level-set method. Int. J. Shap. Model. **8**(2), 79–97 (2002)