Learned Anatomical Joint Constraint Models with Rigid Map Networks

Glenn Jenkins, George Roger, Sean Macken, Michael Dacey and Tim Bashford

School of Applied Computing
University of Wales Trinity St. David Swansea, South Wales
Email: glenn.l.jenkins@uwtsd.ac.uk

Abstract – Accurate individual anatomical joint models are becoming increasingly important for applications in animation, ergonomics and biomechanics. A number of recent approaches have exploited unit quaternions to eliminate singularities when modelling orientations between limbs at a joint. This has resulted in the development of unit quaternion-based joint constraint validation and correction methods. A number of machine learning approaches have been applied to this problem and recent work has demonstrated the use of neural networks trained via competitive learning to model regular conical constraints on the orientation of the limb. In this paper we investigate a derivative of the Self Organising Map, the Rigid Map, for this application.

Keywords – Rigid Map; Unit Quaternion; Joint Constraint; Neural Networks

I. INTRODUCTION

Joint systems are essential components of anatomical models, ensuring anatomically correct movement during simulation. Current techniques are limited by their underlying representation of rotation or their abstraction of the joint function [1]. Accurate anatomical models of individuals are required in a number of fields including animation [2], biomechanics [3] and ergonomics [4]. In current applications, however, increasing accuracy incurs additional complexity often resulting in increased computational cost [5].

The use of dynamics-based models in biomechanics and rehabilitation research has increased, moving to bimodal approaches which combine biomechanical modelling with experimental data (i.e. measured external forces) [6]. Dynamics solutions can be used to produce realistic joint behaviour based on input, contact and constraint forces [7]. However non-invasive, in vivo measurement of model parameters (such as joint torques and constraint forces) are problematic and the verification of estimated values is time consuming and error-prone [6].

Inverse-Kinematics (IK) based approaches allow the precise placement of end effectors as constraints [8]. IK solvers attempt to resolve a system of constraints, but problems arise due to the existence of multiple potentially suboptimal solutions [8]. Well known applications of IK to model human limbs include interactive character control [9], path planning [10], motion retargeting [11], [12] and Motion Capture clean up [13]. IK is also being used to augment dynamics-based modules and provide additional analysis [14].

IK solvers can be classified as: analytical, often resorting to reduced coordinate formalisms; numerical, using iterative approaches to solve a system of constraints, or a combination of these [11], [15]. An important aspect of this is how joint constraints are represented. This work builds on previous research exploring the use of machine learning to model joint constraints; specifically using unsupervised neural networks to model unit quaternion based phenomenological [16] joints (whose behaviour can be modelled without reference to the underlying joint anatomy).

Rigid Maps [17] (similar to Kohonen’s Self Organising Map (SOM)), are used to implicitly model the boundary between valid and invalid orientations by modelling a group of valid unit quaternion orientations. The SOM produces a topography preserving projection of the prototypes from the n-dimensional input space onto an m-dimensional output space [18]. The Rigid Map [17] uses a fixed output space of uniformly distributed unit quaternions. Competitive learning is employed to train a Rigid Map to represent a group of valid unit quaternion orientations. In response to an input orientation, the output is the weight of the output node which best matches the input, this can be used to provide a target for correction.

In this paper constraints on the rotation of the limb (or swing [19]) with regular (circular) bounded constrained regions are considered. Irregular boundaries and rotation around the limb (or twist [19]) are the subject of future work. The paper is an extended version of our paper presented at UKSim 2016 [20] with new results relating to the distribution of output nodes.

II. RELATED WORK

A. Joint Constraint Approaches

Joint constraints can be expressed using Euler angles: this box-limit model is popular in animation tools and file formats [1]. The resulting models are often coarse, failing to capture inter-dimensional dependencies [21] and can
Quaternions are of unit length, to $i^2 = j^2 = k^2 = -1$. Unit quaternions represent $4\pi$ rotations, polar opposites (q and $-q$) represent the same orientation [22] i.e. they are antipodal. Multiplying complex numbers results in rotation in the complex plane, giving rise to the complex identity $i^2 = -1$. This is extended in a subset of quaternion space, where all quaternions are of unit length, to $i^2 = j^2 = k^2 = -1$. Unit quaternions occupy a three dimensional surface (a hypersphere) in four dimensional space and can be used to represent rotations. This representation is redundant as the unit quaternions represent $4\pi$ rotations, polar opposites (q and $-q$) represent the same orientation [22] i.e. they are antipodal.

Approaches to modelling joint constraints represented using unit quaternions include the decomposition of a unit quaternion into conical and axial components (also unit quaternions,) constrained independently [29]. An extension of this approach modelled the relation between the conical and axial constraint components using sampled data [30]. In another approach based on sampled data a set of recorded unit quaternions representing valid orientations are projected as a point cloud in three-dimensions. This removes ambiguity in quaternion space ($q$ and $-q$ representing the same orientation) and allows valid orientations to be represented as a volume whose boundary represents the rotational limits. Spherical primitives were used to create an implicit surface representing the joint limits, however, this initial approach was limited by sparse data in difficult to sample areas [33]. In later work [31], the point cloud was voxelized with the density of each voxel calculated and the voxels subdivided until their density fell below a threshold. An iterative approach was employed in both cases to resolve invalid joint configurations, by rotating toward the nearest primitive (sphere or voxel,) until the orientation was valid.

Statistical approaches have also been employed and again the dimensionality of sampled unit quaternion was reduced. One hemisphere of the unit quaternion hypersphere was projected onto a three-dimensional tangent space allowing pose and joint constraints to be implemented in terms of a maximum deviation from the mean of the projected points [34]. Iterative correction towards the mean, to within the constraint and reverse projection can be used to correct an invalid orientation [34].

B. Neural Network based Joint Constraint Approaches

Artificial Neural Networks (ANNs) are inspired by the structure and mechanics of biological neural networks. They are composed of neurons linked to form complex networks like their biological counterparts, but differ significantly in their complexity and inter-node communications. A diverse range of network architectures and training methods exist from auto-associative memories such as the Hopfield network to unsupervised networks such as Kohonen SOM [35]. ANNs have been employed to model anatomical joint constraints represented using unit quaternions. Here unlike other approaches [29], [30], [33] unit quaternions can be used as input without conversion or projection. Supervised learning approaches have used Generalised Multi-layer Perceptron (or GMLP) Neural Networks with an evolved topology to implicitly model a joint constraint boundary [36]. These supervised training methods require examples of both valid and invalid input patterns along with suitable output pattern, a correction for an invalid orientation and a zero correction for valid input. Error in the neural network approximation results in some correction of valid orientations along with over and under correction of invalid orientations. The former may be partly overcome using a classifier to identify orientations requiring correction (for example a Support Vector Machine (SVM) [37]).

Supervised learning approaches (e.g. GMLP and SVM) are difficult to apply to recorded data (for example motion capture data [31]) as they require both valid and invalid patterns for training. To overcome this issue, ANNs trained using unsupervised techniques such as competitive learning have been proposed.

The SOM is a popular ANN trained using competitive learning [18]. The network is composed of two layers, an input and an output layer each containing nodes. Nodes in the output layer are arranged in a topology (for example a grid), and each input node connected to every output node by a weighted connection. Before training, the weights are randomly assigned, then for each time step patterns (as vectors) are presented at the input nodes. The output nodes compete and the winning output node is that with the shortest Euclidean distance between its weight vector and the input vector [18]. The winning node and its topological neighbours are updated moving the weight vectors of the winning node and its neighbours towards the input, according to a learning rate which decreases along with the size of the neighbourhood as training continues [18]. This reduces impact and magnitude of the training at each time step, with the network becoming less trained.
volatile [18]. Training ends when the network converges i.e. becomes stable [18], in that there is no change in the winner for any pattern or some other stopping condition is reached [38].

SOMs have been trained using a competitive learning approach to implicitly model joint constraints using only valid orientations expressed as unit quaternions [39], [40]. The output nodes of the network are trained via competitive learning to represent the training data while preserving the topography of the input space. The network responds to a given input orientations with the lowest orientation in its model of the input data. This can be used directly for correction [39], or with an iterative approach [40] like that employed by Herda et al [33].

A SOM using real values and Euclidean distance metric was proposed and initial results modelling the rotation of the limb (or swing) for regular boundaries were positive [39]. However extending the model to include the rotation around the limb (twist) causes deterioration. ANNs have been extended into the domain of complex numbers (termed Complex Valued Neural Networks or CVNN) with a steady development of approaches since the 1970s [41]. SOMs with unit quaternion weights and using distance metrics in unit quaternion spaces demonstrate similar performance to real-valued SOMs using a Euclidean distance metric [40], for the limited joint models explored.

C. The Rigid Map Neural Network

The Rigid Map Network is a modified SOM proposed by Winkler et al [17] for pose estimation problems. In their approach, each output node represents a position \((p_i)\) in a known orientation space such that self-organisation is no longer required. The output node topology is therefore fixed and the nodes are uniformly distributed over the orientation space. In this case the \(S^3\) hypersphere, using regular polytopes.

The learning algorithm is modified so that the winning node \((\mathbf{w})\) is based on the proximity between the input pattern \((\mathbf{w})\) and the position \((p_i)\) of the output node rather than its weight determined by the inner product as shown in (1).

\[
\arccos(p_{wi} \cdot \mathbf{w}) \leq \arccos(p_{ii} \cdot \mathbf{w}) \quad \forall i
\]

The updating of weights, however, remains unchanged with the winning node and its neighbours being updated (shown in 2); when fired the network responds with the weight, which is the shortest Euclidean distance from the input [42].

\[
\Delta \mathbf{w}_i = \nu(p_{wi}, p_{i}, t) \cdot \lambda(t) \cdot (\mathbf{w} - \mathbf{w}_i)
\]

Where

\[
\nu(p, q, t) = \nu_0 \cdot e^{-\frac{d(p,q)^2}{\sigma^2(t)}}
\]

Both the learning rate \(\lambda\) and the radius of the neighbourhood \(\sigma\) decay exponentially with time [42], as shown in 4 and 5 respectively. Winkler [42] justifies the use of the Gaussian function for the neighbourhood based on Erwin et al [43] who suggested that while the neighbourhood function chosen is not critical; functions which were convex around the winner, with notably different values at distant neurons gave the best results. In this work the constant \(\varphi\) is adjusted to vary the breadth of the Gaussian function as shown in 3.

\[
\lambda(t) = \lambda(0) \cdot e^{-k\lambda \cdot t}
\]

\[
\sigma(t) = \sigma(0) \cdot e^{-k\sigma \cdot t}
\]

Existing Rigid Map approaches make use of regular polytopes to uniformly distributed the output nodes in unit quaternion space (the surface of an \(S^3\) hypersphere) [17], [20], [42]. The regular polytope employed are the polytetrahedron which consists of 120 vertices making up 600 tetrahedral cells and its reciprocal the polydodecahedron consists of 600 vertices making up 120 dodecahedral cells [44]. Combining these results in the vertices of the polydodecahedron being placed at the centre of the polytetrahedron [42]. An inability to explore larger output layers is a key limitation of using regular polytopes for this purpose and subdivision of the regular polytopes [17] has been suggested to overcome this.

The generation of uniform samples over the rotation group \(SO(3)\) has applications in a number of fields including computer graphics and robotics and numerous approaches exist minimising various criteria [45]. One of the simplest approaches is the OSPHERE algorithm proposed by Fishman [46] which exploits the spherical symmetry of the multidimensional Gaussian density function [45]. Given a hypersphere \(S^d\), \(d + 1\) coordinates \((x_0, \ldots, x_d + 1)\) are drawn from a normal distribution (with zero mean and unit variance,) and the normalised vector \((x_i / ||x||)\) is uniformly distributed over the hypersphere \(S^d\).

It is hypothesised that the Rigid Map Network will produce superior results to the earlier SOM approach as the orientation space is known and self-organisation can be abandoned. This exploratory paper considers the capabilities of the Rigid Map in modelling the orientation of the limb with a regular rotational boundary and no constraint on the rotation around the limb. Future work will explore more complex constraints including irregular boundaries and rotation around the limb.

The remainder of this paper is structured as follows: Section 3 provides a description of our methodology with reference to the techniques employed. Section 4 reports the results of the experiments undertaken with these discussed in Section 5. Finally Section 6 draws conclusions from this work and highlights areas for future investigation.
III. METHODOLOGY

This paper describes the application of the Rigid Map network to the modelling of valid orientations of a virtual limb parametrised using unit quaternions.

A. Rigid Map Network Configuration

The network consists of four input nodes connected to a number of output nodes by a weighted connection. The output nodes are placed into a topology each having a position on the unit quaternion hypersphere. These were initially arranged using a selection of regular polytopes in 4D-space, in this case the polydodecahedron and polytetrahedron (as used by Winkler et al [17], [42]). In this paper we extend this to an arbitrary number of uniformly distributed points in $S^3$ generated using the OSPHERE algorithm [46] described above.

The number of output nodes and number of patterns where fixed as were indicated in TABLE I. Rigid Maps were trained to identify the closest valid orientation for both valid and invalid input where the input layer represents the current limb orientation and the weighted connections of the winning output node represent the nearest valid orientation. An initial evaluation of their performance in the context of anatomical models was undertaken using regular constrained swing regions of varying sizes.

Experiments were undertaken with output layers containing 120, 600, and 720 nodes laid out as polytetrahedron, polydodecahedron and a combination of both respectively. This was extended to include with output layers containing between 120 and 10000 nodes distributed uniformly over the $S^3$ hypersphere.

B. The Rigid Map Training Phase

The Rigid Map training process (defined by Winkler et al [17], [42]) begins with the weights of the interconnections between input and output nodes being set to random orientations. Then for each time step the input set (comprising a number of patterns) is presented to the network. For each pattern a squared distance ($D$) between the input pattern and the position of the output node is calculated using a given metric (detailed in TABLE I). The weights of the node with the shortest distance (the winning node) were adjusted towards the input vector according to the learning rate. As both the input and weight are expressed as unit quaternions spherical linear interpolation (slerp) [47] is used. Output nodes that were within the neighbourhood (topological regions according to position) of the winning node also have their weights updated. The learning rate and neighbourhood decrease exponentially with time.

The constants used are provided in TABLE I along with the ranges used for experimentation. Each Rigid Map was trained until it converged (entered a stable state in which the winning node for each pattern does not change) or a maximum number of training epochs (shown in TABLE I,) was reached.

C. The Rigid Map Usage Phase

The testing or usage phase begins with an input pattern being presented to the trained Rigid Map. The Rigid Map responds with the weight of the winning node using a second distance metric ($E$) between the input and the weight (the connections between the input nodes and a given output node).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input nodes</td>
<td>Number of input nodes</td>
<td>4</td>
</tr>
<tr>
<td>Output nodes</td>
<td>Number of output nodes</td>
<td>720</td>
</tr>
<tr>
<td>Output node distribution</td>
<td>Distribution of output nodes on $S^3$ where constant</td>
<td>Combined vertices of polytetrahedron and polydodecahedron</td>
</tr>
<tr>
<td>Training patterns</td>
<td>Number of training patterns where constant</td>
<td>1000</td>
</tr>
<tr>
<td>Initial learning rate ($\alpha$)</td>
<td>Initial value $\alpha(0)$ where constant. This decays exponentially at each for time step $t$ with constant $k$ (see equation (4)).</td>
<td>1.2</td>
</tr>
<tr>
<td>Learning rate decay constant ($k_l$)</td>
<td>Constant used to control rate of learning rate decay. ($k$ in 4).</td>
<td>0.3</td>
</tr>
<tr>
<td>Initial neighbourhood radius ($\sigma$)</td>
<td>Initial value $(\sigma(0))$, this decays exponentially at each for time step $t$.</td>
<td>2π Radians</td>
</tr>
<tr>
<td>Neighbourhood rate decay constant ($k_n$)</td>
<td>Constant used to control rate of learning rate decay. ($k$ in equation (5)).</td>
<td>1.4</td>
</tr>
<tr>
<td>Neighbourhood constant ($\psi$)</td>
<td>Constant used control the neighbourhood function, ($\phi$ in equation (3)).</td>
<td>0.026</td>
</tr>
<tr>
<td>Maximum training epochs</td>
<td>Maximum number of time steps.</td>
<td>1000</td>
</tr>
<tr>
<td>Distance metrics</td>
<td>Euclidean Distance / Direction Cosine</td>
<td>$E = (|a - c|^2)^{1/2}$ / $D = \arccos(q_0 \cdot q_1)^2$</td>
</tr>
</tbody>
</table>

Each experiment was repeated ten times to ensure the consistency of the results. The Rigid Map used in this work was based on that presented by Winkler et al [42], modified such that the output nodes occupy the whole hypersphere rather than a single hemisphere.

D. Training Data

The unit quaternions occupy the three dimensional surface of a hypersphere ($S^3$) in four dimensional space ($\mathbb{R}^4$) and represent a double covering of the group of rotations (SO(3)) [22] (hence movement between two poles represents a $360^\circ$ rotation). The training set consisted of patterns representing valid orientations, these
were generated on one hemisphere of the unit quaternion hypersphere. This is anatomically correct as while the antipode represents the same orientation a rotation though 360° should not be valid. There are also training implications as demonstrated by our earlier work [32] which showed valid regions on both sides of the hypersphere (with both q and its antipode q being valid) disrupted learning.

The size of the training set was varied between 500 and 6000 patterns.

![Figure 1. Model used for dataset generation. Valid region inside boundary invalid region outside.](image)

In experiments where the range was not varied, a constant range of 90° was used. Defaults for the other parameters are given in TABLE I and were identified though extensive experimentation. The training dataset contained only valid patterns, similar to those recorded from the movement of a human arm. A set of ‘ideal’ corrections (no correction for valid orientations and the nearest valid orientation for invalid ones) were generated using Lee’s [29] approach and provided a measurement of the Rigid Maps capabilities.

IV. RESULTS

The results show the effect of correcting the orientation to that suggested by the Rigid Map (the unit quaternion represented by the weight of the winning node) and indicate successful training of the Rigid Map. An increase in the range (angle between the virtual limb and the z-axis) of the constrained region results in a decrease in performance as shown in Figure 2(a). The resulting corrections however are inferior to those of our earlier SOM [39] as shown in Figure 2(a). This SOM used a similar number of output nodes (625) arranged in a two dimensional grid. It was trained using the same training data and a similar number of training epochs.

Experiments were also undertaken to investigate the effects of increasing the number of training epochs. This produced an increase in performance (as shown in Figure 2(b)), which attenuates rapidly as the number of epochs increases. The error appears independent of the number of training patterns (shown in Figure2(d)). The results show little difference between output nodes uniformly distributed using the OSPHERE algorithm and the polytope based distributions. In both cases the network error decreases as the number of output nodes is increased, (shown in Figure2(c)). In the case of the points distributed using OSPHERE the decay appears exponential.

V. DISCUSSION

The results demonstrate that Rigid Maps are capable of identifying the nearest unit quaternion representing a valid orientation of a virtual anatomical limb, providing an implicit model of a region occupied by valid orientations in unit quaternions space. The Mean Squared Error (MSE) on the test set (containing invalid and valid orientations) is reasonably low, but higher than those for the SOM (shown in Figure 2(a)) [39]. Research indicates that the dimensionality of the output layer of the SOM should match the information dimension of the input data [48] and that where the input manifold is of a higher dimension than the neural lattice there is a trade-off between continuity and resolution in the map created [49]. This suggests that while the grid based SOM currently has an advantages, the introduction of another dimension of variation (twist around the limb) may require a higher dimensional output space.

As in the earlier SOM-based approach [39] there were issues with over-correction of invalid input and the correction of valid input. Over-correction occurs where the limb is corrected to the orientation provided by the Rigid Map (or the SOM) which is inside (rather than on the boundary of) the valid region. The correction of valid input occurs when the input orientation is already valid and the Rigid Map responds with the closest orientation in its map of the valid region. A possible solution is the use of the correction method proposed in our earlier work [40]. This used the largest Euclidean distance between input and output of the SOM during training a threshold for correction to reduce correction of valid orientations. Along with an iterative dichotomic correction to an orientation within this threshold to reduce overcorrection. Augmentation of the Rigid Map to give a continuous output rather than a single value has been proposed by Winkler [42] (based on earlier adaptions of the SOM [50], [51]).
Their approach makes use of the topology-preserving nature of the mapping to provide a weighted average of selected output nodes. It could be employed to reduce correction of valid input orientations in both the Rigid Map and SOM. In addition, the testing process measures the MSE based on the distance from the boundary and provides an indication of performance. However, it provides no indication of the quality of the map produced. Alternatives which quantify the resolution and continuity have been proposed and may allow more detailed comparison of approaches [49].

The results provide an insight into the effects of problem, network and training attributes on performance. It is clear that the network is capable of learning constraints of varying sizes, although larger constraints appear to demonstrate a higher error. This suggests an increase in over-correction of valid points as output nodes are more dispersed over the valid region and an increase in over-correction of invalid points as fewer output nodes occupy spaces near the boundary. Improvements resulting from the increase in output nodes can be ascribed to an increase in the density of output nodes over the valid region, reducing correction errors. Winkler [42], recommends a uniform distribution of output nodes, however no further polytope exist [44]. Using the alternative methods suggested by Fishman [46], the density of output nodes can be further increased improving the results.

Previous results with the SOM [39] network showed improvements with an increase in the data set size but these are not echoed in the results for the Rigid Map, suggesting that the other factors limit further improvements in performance. Increasing the duration of training (the number of training epochs,) has some effect but this rapidly attenuates.

VI. CONCLUSION

Rigid Maps have been shown to be capable of implicitly modelling the boundary between valid and invalid orientations in unit quaternion space to a degree of accuracy. However, this requires that the output nodes are uniformly distributed in the output space [42]. This initial research shows them to demonstrate inferior performance to the traditional SOM with output nodes in a two dimensional grid. This we ascribe to the use of a fixed output node topology during training. Both approaches have similarities to non-machine learning based solutions [31], [34] with the advantage that no decomposition or reformatting of the unit quaternion orientation is required. Limited validation has been undertaken. The results above compare the network output to ‘ideal’ test data produced using the approach suggested by Lee [29], which is limited to regular boundaries. A comparison of the Rigid Map performance with that of our earlier SOM [39] against the same test data is also presented. The next stage in the validation of this model is to compare other approaches [27], [30], [31] in terms of accuracy, map quality and execution time. The ideal datasets in this case...
will be generated using the virtual limb sampling approach outlined in our earlier work [52].

Further improvements of the Rigid Map, including smoothing of the output to give a continuous output rather than a single value, have been proposed [42] (based on earlier augmentations the SOM [50], [51]). Research suggests that where the input manifold is of a higher dimension than the neural lattice, there is a trade-off between continuity and resolution [49]. The Rigid Map may produce more continuous maps and therefore provide a better solution when coupled with such techniques. Alternative approaches to generation of uniform incremental grids on SO(3) [45] may improve the distribution of output nodes in the Rigid Map and have implications for sampling when creating input.

In addition, there are a large number of networks based on the SOM, which aim to overcome its shortcomings (including the limits of its fixed output layer) [53] and a number of metrics to quantify the performance of mappings produced have been developed [49]. Future work will consider the use of such metrics to further evaluate the quality of the maps produced by the SOM and Rigid Map. Along with the improvement of the SOM and Rigid Map neural networks developed and alternatives such as the Growing Neural Gas neural network [54] are also considerations for our future work.

Current results are encouraging and suggest that Rigid Maps are able to implicitly model constraints on the rotation of the limb with regular boundaries in unit quaternion space. They may have potential for modelling similar constraints with irregular boundaries and rotation around the limb while providing advantages over current approaches. Beyond anatomical joint constraint there may be scope for the use of Rigid Maps in other context which require the constraint of orientations expressed as unit quaternion.

REFERENCES


