Interpolated SOM Neural Networks for Anatomical Joint Constraint Modelling

Glenn Jenkins *, Ana Calderon

Cardiff School of Technologies
Cardiff Metropolitan University, Cardiff, South Wales.
Email: gljenkins@cardiffmet.ac.uk

Carolyne Skye, Larry Boltovskoi

UCLL University of Applied Sciences
Leuven
Belgium

***Abstract* - Unit quaternions offer a singularity free representation when modelling orientations between limbs at a joint. The development of accurate joint constraint models for such joints is a non-trivial task and several approaches have been suggested including a number which leverage machine learning which aim to create joint models based training data from an individual or group. Previous work has demonstrated the use of an extended Rigid Map Neural Network with a continuous output to model conical constraints (with a regular boundary). In this paper we employ a similar approach deploying an extended Self Organising Map (SOM) with a continuous output.**

Keywords - Self Organizing Map, Continuous, Unit Quaternion, Joint Constraint

I. INTRODUCTION

There are a number of fields in which anatomical joint models are required, constraints are either on the maximum range of angular motion (for example [1]) or a subset of comfortable poses ignoring the extremes (for example [2]). These joint models are phenomenological [3] modelling joint function but not the underlying biological structures. Anatomical joint models are required in biomechanical simulations to model patients [9] and in ergonomics to model average individuals [10]. In animation they are used ensure the realistic motion of articulated humanoid figures [4]. Constrained anatomical models (called priors) are fitted to recovered image [5-7] or motion capture data [8]. There is also an increasing interest in the use of models in the development of control systems for personal assistance robots [11, 12].

Unit Quaternion are well suited to the parametrization of human motion [13] and several approaches for defining constraints have been suggested. Learned anatomical joint constraint models have recently gained popularity, early approaches used synthetic datasets for two class supervised learning of both binary [14] (valid or invalid) and corrective (zero or correction) [15] constraints. More recent approaches [5, 11] capitalise on a growth in the number of publicly available human motion capture datasets. Such datasets contain only positive (valid) examples and invalid samples must be synthesised, unsupervised learning using only the recorded data is therefore attractive.

Our previous work has shown that Kohonen's SOM can be trained to implicitly model valid orientations parametrised as unit quaternions [17]. The trained network response is the weight of the output node which best matches the input, this can be used to provide a target for correction. The SOM can be extended to provide a continuous rather than discrete output [18]; applied to the SOM like Rigid Map neural network this produced a more accurate constraint model [16].

This paper combines our earlier work using the SOM [17] with methods to produce a continuous rather than discrete output we previously used to augment Rigid Map Network [16]. The constraints modelled are again simplified to a regular conical constraint on the rotation of the limb (or swing [19]) rotation around the limb (or twist [19]) is fixed.

II. RELATED WORK

The selection of rotational representation is a key factor in modelling anatomical joint constraints and unit quaternions are gaining popularity in kinematic anatomical models [13]. Unit quaternion algebra provides a singularity free rotational representation [20] while offering simple averaging and interpolation [13]. Unit quaternions are hyper-complex numbers, which occupy the surface of an S^3 hypersphere in four dimensional space ($\mathbf{R}^4$). They provide a redundant mapping of $SO(3)$ representing 4π rotations, antipodes (q and $-q$) represent the same orientation [20].

A number of approaches have been suggested to identify valid and invalid joint configurations expressed as unit quaternion. Lee [21] decomposed a single unit quaternion into a pair; each representing rotation in a single plane, conical, axial and revolute joints are defined. An extension of this approach modelled the relation between the conical and axial components based on recorded samples [22]. Support Vector Machines [14] and more recently fully connected feedforward neural networks [5] trained using supervised learning have also been used to implicitly model constraints classifying orientations as either valid or invalid.

More practical unit quaternion joint constraint models provide both validation and suggest a method to correct invalid orientations. Removing the $q = -q$ ambiguity in

unit quaternion space allows one of the four components to be dropped, this can later be re-calculated based on the three remaining components and the length (as the sign is know). Spherical primitives [1] and voxels [23] can be used to model the resulting three dimensional point cloud. In both cases a target for correction is identified by interpolating by dichotomy nearest primitive or voxel until the orientation is valid. Johnson [24] projected one hemisphere unit quaternion hypersphere into a three dimensional tangent space. A maximum deviation from the mean of recorded valid orientations provides the constraint while movement towards the mean (projected back into unit quaternion space,) is used for correction [24].

A similar statistical approach used a four dimensional Von Mises-Fisher distribution to model sampled unit quaternion over a hypersphere in $\mathbf{R}^4$ [25]. The parameters for its probability density function are obtained per-join from a motion capture database [25]. A key advantage of this approach is that there is no need to pre-process the unit quaternion prior to constraint.

Machine learning is attractive for the development of unit quaternion join constraint systems, with the potential to learn either from individual or selected demographic data. Feedforward neural networks trained using supervised learning have been used to create implicit models of individual joint constraint boundaries, learning a corrective response to a presented unit quaternion orientation [26]. They have also been used to model a hyper-surface representing the valid orientations of multiple limbs, the gradient resulting differentiable function can be used to enforce constraint [11]. A key limitation of these methods is the need to include both valid and invalid training samples, making them difficult to apply directly to data from human subjects (for example motion capture data).

One approach to overcoming this limitation generating training data. Synthetic datasets have been generated using other unit quaternion joint constraint approaches for example [26]. The publication of large motion capture datasets have allowed the synthesis of balance training sets for supervised training. Jiang and Liu [11] used a dataset and validation method proposed by Akler and Black [2]. Murphy *et al* [5] populated an occupancy matrix representing swing and associated twist based on the Human3.6m dataset [27].

Unsupervised training methods such as competitive learning require only valid samples and offer an alternative to generated training data, potentially being able to learn directly from recorded samples. Simplified joint constraints parametrized as unit quaternions have been implicitly modelled using SOMs [17] trained via competitive learning. The network is trained to model valid orientations and responds to an input orientation as a unit quaternion with the weight of the nearest output node (also a unit quaternion). Used for correction [17] this results over correction of invalid orientations and undesirable correction of valid input. A threshold for correction combined with iterative correction towards network output [28] similar to the technique used by Herda *et al* [1] can be used to improve the results. In an extension to this work unit quaternion distance metrics were employed [28] with no significant improvement in the results.

A close relative of the SOM the Rigid Map Network [29] has also been applied to joint constraint modelling [30, 31]. Here output nodes represent a position in a known orientation space and the winning node is dictated using this position rather than the weight. Regular polytopes [30] and other methods [31] were employed to uniformly distribute output nodes in unit quaternion space. This was extended produce a continuous output with some success [16].

SOMs are a two layer feed-forward network trained using competitive learning [32]. Each layer is composed of nodes, the output layer is arranged according to a topology (such as a grid), and each node in the input layer has a weighted connection to each output node. During training weights are initialised to small random values, for each epoch training patterns (as vectors) are presented at the input nodes. The winning output node has the closest weight to the input node measured by Euclidean distance [32]. The winning node and its topological neighbours have their weights adjusted towards the input. The fractional learning rate controls this movement and decreases along with the neighbourhood radius for each epoch [32]. As the impact of the training is reduced over time the network becomes more stable [32]. When there are no changes in the winning node for any pattern the network has converged i.e. becomes stable [32] where this does not occur other computational limits may be used [33].

In the Batch SOM [34-36] training is modified such that the weights are updated after each epoch, rather than each pattern. This overcomes issues with pattern order [36] and allows the use of data-partitioned parallel methods [37]. The following equations are based on Lawrence *et al* [37], Equation 1 shows the weights at the end of an epoch $\mathbf{w}_k(t_f)$ for a SOM with k hidden nodes:

$$\mathbf{w}_k(t_f) = \frac{\sum_{t'=t_o}^{t'=t_f} \bar{h}_{ck}(t')\mathbf{x}(t')}{\sum_{t'=t_o}^{t'=t_f} \bar{h}_{ck}(t')} \tag{1}$$

here t_0 and t_f denote the start and end of the current epoch. The winning node is computed by calculating the distance between the input $\mathbf{x}(t)$ and each of the k weights $\mathbf{w}_k$, shown in Equation 2 below:

$$\bar{d}_k(t) = ||\mathbf{x}(t) - \mathbf{w}_k(t_0)||^2 \tag{2}$$

here $\mathbf{w}_k(t_0)$ are the weight notes from the end of the previous epoch. The winning node is computed by identifying the node closest to the input in the weight space, shown in Equation 3 below:

$$d_c(t) \equiv \min_k \bar{d}_k(t) \tag{3}$$

A Gaussian neighborhood function over a 2D lattice of nodes is used to control the extent to which a weight $\mathbf{w}_k$ is moved towards an input where the winning node has the weight $\mathbf{w}_c$. As shown in Equation 4 below:

$$\bar{h}_{ck}(t) = exp(-||\mathbf{r}_k - \mathbf{r}_c||^2/\phi\sigma(t)^2 \tag{4}$$

The vectors $\mathbf{r}_k$ and $\mathbf{r}_c$ represent two nodes in the topology. The width of the neighborhood function $\sigma(t)$ is decreased during training, using a suitable function. The implementation used in this work developed by Gorman [38] includes an additional parameter to provide additional control of the the width of the neighborhood function ϕ.

Goppert and Rosenstie [18] explored the use of interpolation techniques extending the SOM to produce a continuous output. In the usage phase the weight of the winner $\mathbf{w}_{c0}$ along with n runner-up nodes $w_{c1}, ..., w_{cn}$ are combined. They noted that distance based methods that combined the n closest nodes using an inverse weighted sum [18] or an exponential function [39] were limited by the assumption that input and output neighborhoods were similar. Geometrical and topological interpolation have been proposed to overcome these issues [18].

We hypothesize that a SOM with interpolated output can model regular conical joint constraint expressed as a unit quaternion with a greater degree of accuracy than the SOM. The remainder of this paper is structured as follows: Section 3 outlines the methodology used with reference to the techniques employed. Experimental results are reported in Section 4 with these being discussed in Section 5. Finally Section 6 draws provides a conclusion to this work and suggests areas for future investigation.

III. METHODOLOGY

This paper describes an extension to our earlier work [28] modelling valid orientations of a virtual limb parametrized using unit quaternions using a SOM neural network. A similar methodology is employed and repeated here for completeness. A SOM with a grid topology was trained to identify the closest valid orientation both for valid and invalid input, we extend this here to consider a group of closest nodes. The input layer represents the current virtual limb orientation as a unit quaternion. The weighted connections of the winning output node (or interpolated winning nodes) represent the nearest valid orientation as a target for correction. The performance of the augmented SOM in the context of anatomical joint constraint modelling was explored using regular constrained regions of varying sizes.

TABLE I. TRAINING CONFIGURATION

Parameter	*Description*	*Setting*
Input nodes	Number of input nodes	2^2
Output nodes	Number of output nodes	25^2
Training patterns	Number of training patterns.	1000
Initial neighborhood radius σ	Initial value σ_0	Output Nodes / 2
Neighborhood radius	Radius of the neighborhood at time t	$\sigma_t = \sigma_t - t\left(\frac{\sigma_0 - 1}{t_f - 1}\right)$
Constant of the Standard Deviation Φ	Additional control over the width of the neighborhood function.	0.5
Maximum training epochs	Maximum number of time steps.	1000
Distance metrics	Euclidean Distance	$E = (\|(q^a - q^b)\|)^2$
Batch Size	Size of dataset batches.	512
Interpolation Nodes	Number of interpolation nodes, where constant.	2

The Batch SOM training process begins with the weights of the interconnections between input and output nodes being set to small random values. The output nodes are arranged in a 2D grid topology. During training the input dataset is partitioned into batches, which are processed parallel. Each input pattern is presented in sequence, for each the winning node is identified as that whose weight is closest to the input; the distance metric used is provided in TABLE I. An update to the weight of the winning node is calculated moving this towards the input pattern. The weights of nodes in the topological neighborhood are similarly updated relative to their proximity to the winning node. The neighborhood size decrease linearly with time. Once all batches have been processed the the cumulative updates for all weights are applied. The constants used during training are provided in TABLE I each SOM was trained until a maximum number of training epochs was reached.

The usage phase begins with an input pattern being presented to the trained SOM. The network responds with the weight of the winning node. At this stage various interpolation techniques were applied. The simplest approach took the average of the weights of the closest N interpolation nodes $\mathbf{w}_{ci}$.

$$\mathbf{w}_o = \frac{\sum_{i=1}^{N} \mathbf{w}_{ci}}{N} \tag{5}$$

here $\mathbf{w}_o$ is the output. The second approach used an inverse weighted average of the weights of the N closest nodes:

$$\mathbf{w}_o = \sum_{i=1}^{N} \alpha(\mathbf{x}, \mathbf{w}_{ci}) - \mathbf{w}_{ci} \tag{6}$$

here $\mathbf{x}$ is the input, α is the inverse weighting and $w_c i$ is the current interpolation node.

$$\alpha(\mathbf{x}, \mathbf{c}_i) = \frac{(\sum_{j=1}^{N} \mathbf{x} - \mathbf{w}_{cj}) - \mathbf{w}_{ci}}{\sum_{j=1}^{N} \mathbf{x} - \mathbf{w}_{cj}} \tag{7}$$

Finally geometric interpolation [18], for brevity equations 8 to 10 show only a single interpolation node $\mathbf{w}_{ci}$ however, this approach can be repeated for additional interpolation nodes.

$$\mathbf{w}_o = \mathbf{w}_c + \hat{\mathbf{d}}_{ci}(\|\mathbf{d}_{xc}\|(\hat{\mathbf{d}_{xc}} \cdot \hat{\mathbf{d}_{ci}})) \tag{8}$$

where:

$$\mathbf{d}_{xc} = \mathbf{x} - \mathbf{w}_c \tag{9}$$

$$\mathbf{d}_{ci} = \mathbf{w}_c - \mathbf{w}_{ci} \tag{10}$$

Here $\mathbf{w}_c$ is the closest (winning) node and $\mathbf{w}_{ci}$ the interpolation node. The output is then used as the correction target, invalid orientations are moved towards the boundary while valid orientations remain unchanged. In previous approaches valid orientations were often corrected and invalid orientations poorly corrected due to the discrete nature of the output. The use of interpolation aims to reduce both.

Output layers containing between 2^2 and 26^2 nodes were used in experimentation, with 25^2 nodes used when constant. Training datasets varied between 500 and 6000 patterns with a test set of 6000 patterns and batches (for the Batch SOM training,) between 8 and 1024 patterns. A constant range of 90° was used where the range was not varied. TABLE I provides default values for the other parameters used these were identified though experimentation. To ensure the consistency of the results each experiment was repeated five times. The training dataset contained only valid orientations (see Figure 1, to retain anatomical validity only orientations antipodal orientations were not considered valid, this has been shown to improve training [40]. For validation a dataset comprising 'ideal' corrected orientations was generated using an existing quaternion constraint approach (see [31] for details).

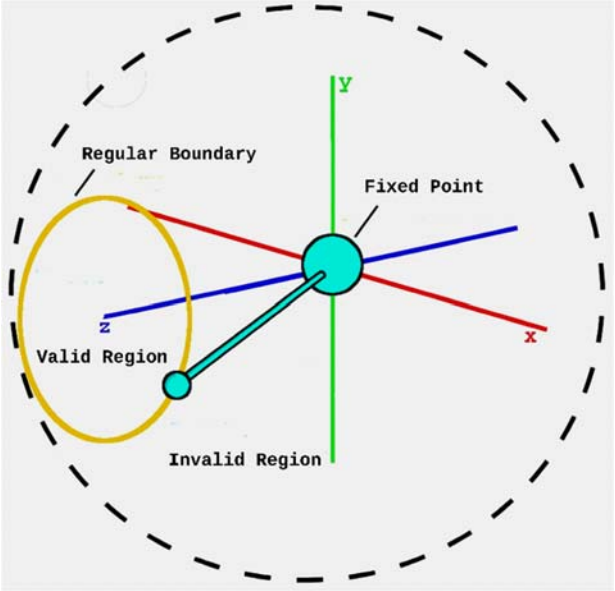

Figure 1. Dataset generation model, the valid region is inside the regular boundary the invalid region outside reproduced from [28] used with permission.

IV. RESULTS

Given suitable training data the SOM is able to provide a nearest valid quaternion to a given input. As the range of the constraint increases the performance decreases as shown in Figure 2(b). Increasing the number of output nodes predictably increases results until another aspect of the training becomes limiting (patterns, training epochs etc.) The number of training patterns seems to have less impact on the performance, as shown in Figure 2(c). During training the dataset is partitioned into batches; training on these is conducted in parallel. Increasing the batch size relative to that of the training set increases performance (as shown in Figure 2(d)) the increase slows as other factors limit the SOMs performance.

Figure 2(a) compares the interpolation approaches with an increasing number of interpolation, nodes in addition to the winner. The 'discrete' plot shown is a SOM with no interpolation and is for reference only. With a small number of notes all approaches improve on the discrete SOM, in each case geometric interpolation out performs the weighted average which in turn out performs the average. Both average and weighted average decrease in performance as the number of interpolation nodes increases. Geometric interpolation shows some improvement with up to two nodes but no further improvement beyond this. Interpolation (particularly geometric interpolation,) produces a greater improvement for larger ranges as shown in Figure 2(b).

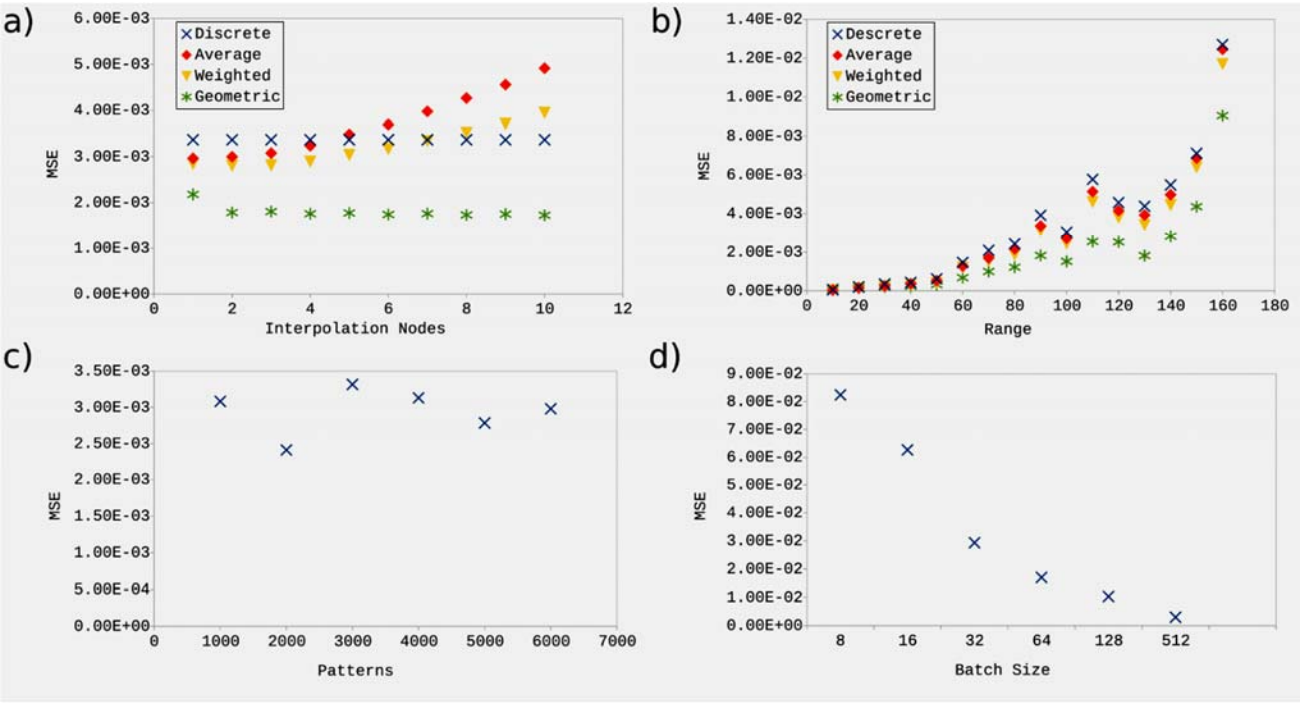

Figure 2. Graph (a) shows the effect of increasing the number of nodes used for interpolation, with the discrete (no interpolation,) for comparison. Graph (b) shows the range of the constraint for discrete and selected interpolation methods, with two interpolation nodes. Graph (c) shows the increase in the number of patterns, while (d) shows the impact of increasing batch size.

V. DISCUSSION

SOMs are capable of modelling valid orientations in unit quaternions space and can be used as a target for correction, the Mean Squared Error (MSE) when correcting given invalid and valid orientations is comparable to those from our earlier SOM based approach [17]. The results demonstrate that the SOM and extended SOM with interpolation are capable of providing a suitable target for correction. The reported Mean Squared Error (MSE) is comparable to our earlier work with SOM [17] and Rigid Map [16] networks. The discrete result is again improved by interpolating between the winning node and a small number of runner up (or interpolation) nodes as observed in Figure 2(a) and Figure 2(b).

Geometric interpolation produced superior results (shown in Figure 2(a) and Figure 2(b)) this suggests that both the contribution to the winning node and vector of interpolation are significant. The average and weighted average approaches showed decreasing performance as the number of interpolation nodes was increased. Here the nodes used in interpolation are becoming progressively further from the input and their influence has a negative impact as shown in Figure 2(a). Figure 2(b) shows that the interpolated approaches consistently out perform the discrete SOM with geometric interpolation providing significant improvements in all cases. Increasing the number of training patterns has little impact for the range tested (see Figure 2(c)), however, the Batch SOM appears more sensitive to the size of batches used in training. Figure 2(d) suggests an asymptotically decaying increase in performance as the batch size approaches half the dataset.

The discrete SOM struggled with correction of valid orientations and the overcorrection of invalid orientations as the constrained region grew and output node density decreased [31]. While increasing the density of the output layer reduces this [31], the ideal orientation for correction is often between the winner and runner up nodes either within the valid region or near the boundary. Interpolating between these can produce more accurate result without an increase in the output layer density, potentially reducing computational cost.

VI. CONCLUSION

When applied to the modelling of anatomical joint constraints in unit quaternion space the SOM with an interpolated output produces an improvement over the traditional SOM with discrete output. Both average and weighted average approaches are effective provided that the number of interpolation nodes is small, geometric interpolation can be used with any number of interpolation

nodes but no improvement is observed after the first two nodes. This is significant as there is a computational cost associated with the interpolation process, a small number of interpolation nodes would incur little overhead.

Future work will also explore the creation of motion capture based training and test sets, using a similar approach to Jiang and Liu [11] for either a individual joints or multiple inter-related joints. This will require moving beyond modelling the regular conical constraints on the orientation of the limb (swing,) and include rotation along the limb axis (twist).

REFERENCES

[1] L. Herda, R. Urtasun, P. Fua, and A. Hanson, "Automatic determination of shoulder joint limits using quaternion field boundaries," *Int J Rob Res*, vol. 22, no. 6, pp. 419–444, 2003.

[2] I. Akhter and M. J. Black, "Pose-conditioned joint angle limits for 3d human pose reconstruction," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 1446–1455.

[3] A. E. Engin and S. T. Tumer, "Improvised dynamic model of human knee joint and its response to loading," *J. Biomech. Eng.*, vol. 115, no. 2, pp. 137–142, 1993.

[4] L. Zhu, X. Hu, and L. Kavan, "Adaptable anatomical models for realistic bone motion reconstruction," in *Computer Graphics Forum*, vol. 34, no. 2. plus 0.5em minus 0.4emWiley Online Library, 2015, pp. 459–471.

[5] =2plus 43minus 4 P. Murthy, H. T. Butt, S. Hiremath, A. Khoshhal, and D. Stricker, "Learning 3d joint constraints from vision-based motion capture datasets," *IPSJ Transactions on Computer Vision and Applications*, vol. 11, no. 1, p. 5, 2019. [Online]. Available: https://doi.org/10.1186/s41074-019-0057-z =0pt

[6] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik, "End-to-end recovery of human shape and pose," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7122–7131.

[7] =2plus 43minus 4 D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt, "Vnect: Real-time 3d human pose estimation with a single rgb camera," vol. 36, no. 4, 2017. [Online]. Available: http://gvv.mpi-inf.mpg.de/projects/VNect/ =0pt

[8] =2plus 43minus 4 L. Herda, R. Urtasun, and P. Fua, "Hierarchical implicit surface joint limits for human body tracking," *Computer Vision and Image Understanding*, vol. 99, no. 2, pp. 189–209, 2005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1077314205000159 =0pt

[9] R. Riemer, E. T. Hsiao-Wecksler, and X. Zhang, "Uncertainties in inverse dynamics solutions: a comprehensive analysis and an application to gait," *Gait & posture*, vol. 27, no. 4, pp. 578–588, 2008.

[10] =2plus 43minus 4 X. Wang, "A behavior-based inverse kinematics algorithm to predict arm prehension postures for computer-aided ergonomic evaluation," *J Biomech*, vol. 32, no. 5, pp. 453 – 460, 1999. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021929099000238 =0pt

[11] Y. Jiang and C. K. Liu, "Data-driven approach to simulating realistic human joint constraints," in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. abs/1709.08685, May 2018.

[12] Z. Erickson, V. Gangaram, A. Kapusta, C. K. Liu, and C. C. Kemp, "Assistive gym: A physics simulation framework for assistive robotics," 2019.

[13] =2plus 43minus 4 J. H. Challis, "Quaternions as a solution to determining the angular kinematics of human movement," *BMC Biomedical Engineering*, vol. 2, no. 1, p. 5, 2020. [Online]. Available: https://doi.org/10.1186/s42490-020-00039-z =0pt

[14] G. Jenkins, "Support vector machines for anatomical joint constraint," in *11th International UKSim Conference on Computer Modelling and Simulation*. plus 0.5em minus 0.4emCambridge, UK: IEEE Computer Society, 2009, pp. 186–190.

[15] G. Jenkins and P. Angel, "Evolved topology genralized multi-layer percerptron (gmlp) for joint constraint modelling," in *9th International Conference on Computer Modelling and Simulation*, K. Al-Begain, A. Orsoni, and D. Al-Dabass, Eds. plus 0.5em minus 0.4emOxford, UK: United Kingdom Simulation Society, 2006, pp. 1–6.

[16] G. L. Jenkins and A. P. N, "Interpolated rigid map neural networks for anatomical joint constraint modelling," *International Journal of Simulation: Systems, Science and Technology*, vol. 20, no. S1, pp. 13.1–13.6, Mar. 2019.

[17] G. L. Jenkins and M. E. Dacey, "Self organising maps for anatomical joint constraint," in *13th International UKSim Conference on Computer Modelling and Simulation*. plus 0.5em minus 0.4emCambridge, UK: IEEE Computer Society, 2011, pp. 42–47.

[18] J. Göppert and W. Rosenstiel, "Topology-preserving interpolation in self-organizing maps," in *NeuroNimes, EC2*, Ñımes, France, October 1993, pp. 425–434.

[19] =2plus 43minus 4 F. S. Grassia, "Practical parameterization of rotations using the exponential map," *J. Graph. Tools*, vol. 3, no. 3, pp. 29–48, Mar. 1998. [Online]. Available: http://dx.doi.org/10.1080/10867651.1998.10487493 =0pt

[20] A. Watt and M. Watt, *Parameterisation of Rotation*. plus 0.5em minus 0.4emNew York: ACM Press, 1992, pp. 356–368.

[21] J. Lee, "A hierarchical approach to motion analysis and synthesis for articulated figures," PhD, Korean Advanced Institute of Science and Technology, 2000.

[22] Q. Liu and C. Prakash, E, "The parameterization of joint rotation with the unit quaternion," in *7th Digital Image Computing: Techniques and Applications*, C. Sun, H. Talbot, S. Ourselin, and T. Adriannsen, Eds., Sydney, Australia, 2003, pp. 410–417.

[23] L. Herda, R. Urtasun, and P. Fua, "Hierarchical implicit surface joint limits for human body tracking," Computer Vision Lab, Ecole Polytechnique Federal de Lausanne (EPFL), Tech. Rep. CH-1015, 2004.

[24] M. P. Johnson, "Exploiting quaternions to support expressive interactive character motion," PhD, Massachusetts Institute of Technology, 2003.

[25] E. Brau and H. Jiang, "A bayesian part-based approach to 3d human pose and camera estimation," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, Dec 2016, pp. 1762–1767.

[26] G. L. Jenkins and M. E. Dacey, "Evolved topology generalized multi-layer perceptron (gmlp) for anatomical joint constraint modelling," in *14th International UKSim Conference on Computer Modelling and Simulation*. 1em plus 0.5em minus 0.4emCambridge, UK: IEEE Computer Society, 2012, pp. 107–112.

[27] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1325–1339, 2014.

[28] G. L. Jenkins and M. E. Dacey, "A unit quaternion based som for anatomical joint constraint modelling," in *16th International UKSim Conference on Computer Modelling and Simulation*. plus 0.5em minus 0.4em Cambridge, UK: IEEE Computer Society, March 2014, pp. 89–95.

[29] =2plus 43minus 4 S. Winkler, P. Wunsch, and G. Hirzinger, "A feature map approach to pose estimation based on quaternions," in *Artificial Neural Networks â€" ICANN'97*, ser. Lecture Notes in Computer Science, W. Gerstner, A. Germond, M. Hasler, and J.-D. Nicoud, Eds. plus 0.5em minus 0.4em Springer Berlin Heidelberg, 1997, vol. 1327, pp. 949–954. [Online]. Available: http://dx.doi.org/10.1007/BFb0020275 =0pt

[30] G. L. Jenkins, G. Roger, M. E. Dacey, and T. Bashford, "A rigid map neural network for anatomical joint constraint modelling," in *18th International UKSim Conference on Computer Modelling and Simulation*, Cambridge, UK, April 2016, in Press.

[31] =2plus 43minus 4 G. Jenkins, G. Roger, S. Macken, M. Dacey, and T. Bashford, "Learned anatomical joint constraint models with rigid map networks," *International Journal of Simulation, Systems, Science and Technology*, vol. 17, no. 35, pp. 31.1–31.7, August 2016, http://ijssst.info/Vol-17/No-35/paper31.pdf. [Online]. Available: http://ijssst.info/Vol-17/No-35/paper31.pdf =0pt

[32] K. Mehrotra, C. K. Mohan, and S. Ranka, *Unsupervised Learning*. plus 0.5em minus 0.4emMassachusetts: MIT Press, 1997, pp. 155–216.

[33] T. J. Klassen and M. I. Heywood, "Towards the on-line recognition of arabic characters," in *International Joint Conference on Neural Networks*, Honolulu, HI, May 2002, pp. 1900–1905.

[34] S. Luttrell, "Derivation of a class of training algorithms," *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, vol. 1, pp. 229–32, 02 1990.

[35] T. Kohonen, "Things you haven't heard about the self-organizing map," in *IEEE International Conference on Neural Networks*, 1993, pp. 1147–1156 vol.3.

[36] F. Mulier and V. Cherkassky, " Self-organization as an iterative= kernel smoothing process." *Neural computation*, vol. 7, pp. 1165–77, Nov 1995.

[37] =2plus 43minus 4 R. D. Lawrence, G. S. Almasi, and H. E. Rushmeier, "A scalable parallel algorithm for self-organizing maps with applications to sparse data mining problems," *Data Mining and Knowledge Discovery*, vol. 3, no. 2, pp. 171–195, 1999. [Online]. Available: https://doi.org/10.1023/A:1009817804059 =0pt

[38] =2plus 43minus 4 G. C. (2018, Feb.) Tesnor flow som. Github Repository. [Online]. Available: https://github.com/cgorman/tensorflow-som =0pt

[39] Y. Cheneval, P. Demartines, and L. Tettoni, "Function approximation using an architecture based on Kohonen self-organizing maps," in *Journées Neurosciences et Sciences de l'Ingénieur*, Oléron, May 1992.

[40] G. Jenkins and P. Angel, "Joint constraint modelling using evolved topology generalized multi-layer perceptron (gmlp)," *International Journal of Simulation, Systems, Science and Technology*, vol. 9, no. 5, pp. 15–27, 2008.