Similarity K-d tree method for Sparse Point Pattern Matching with Underlying Non-rigidity

Submitted to Pattern Recognition, revised version, 24 March 2005

Baihua Li * (corresponding author)

Dept. of Computing and Mathematics

Manchester Metropolitan University

Manchester, M1 5GD, UK

b.li@mmu.ac.uk

Tel: +44 (0)1612473598

Fax: +44 (0)161 247 1483

Qinggang Meng and Horst Holstein

Dept. of Computer Science

University of Wales, Aberystwyth

SY23 3DB, Wales, UK

qqm@aber.ac.uk, hoh@aber.ac.uk

Abstract

We propose a method for matching non-affinely related sparse model and data point-sets of identical cardinality, similar spatial distribution and orientation. To establish a one-to-one match, we introduce a new similarity K-dimensional tree. We construct the tree for the model set using spatial sparsity priority order. A corresponding tree for the data set is then constructed, following the sparsity information embedded in the model tree. A matching sequence between the two point sets is generated by traversing the identically structured trees. Experiments on synthetic and real data confirm that this method is applicable to robust spatial matching of sparse point-sets under moderate non-rigid distortion and arbitrary scaling, thus contributing to non-rigid point-pattern matching.

Keywords: *K*-dimensional tree; non-rigid point pattern matching; non-rigid pose estimation; robust point pattern correspondence; motion capture.

1 Introduction

Point pattern matching (PPM) and related topics have been extensively studied within a rich literature encompassing both theoretical and practical issues in areas such as computer vision, pattern recognition, computational geometry, astronautics, computational biology and computational chemistry [1, 2, 3, 4, 5, 6, 7].

We propose another application relating to PPM, arising from marker-based optical motion capture (MoCap) systems, widely used in clinical gait analysis, animation and

computer games, emphasising human motion representation [8, 9]. For MoCap, passive reflective markers acting as feature points are attached on a human subject. The subject's movement can be accurately recorded in "real-time", represented by a sequence of 3D feature-point data. Reconstruction of the subject's movement requires an additional step of *model-based* identification of the captured point data. The state of the art for model generation makes use of manual identification of feature points in one frame captured from a design pose of the subject, such as shown in Fig. 1. Typically, the same marker protocol and design pose are used on many different subjects, yet manual labelling is still needed for every new subject model generated. This applies even when the same subject is used in different MoCap trials, if markers have become detached or displaced. Model generation is labour intensive, highly non-productive and consequently costly in commercial situations.

In this study, we relegate the model generation issue in MoCap systems in general to the robust one-to-one PPM with underlying non-rigidity. Formally, we consider two point sets that are extracted from two subjects with underlying non-rigidity and non-uniform scaling, one being the model set with known point identities and the other representing the observed data. There exists neither a global nor local affine transformation between the point-sets. Assume we have an identified point-set $\mathcal{P} = \{p_i \in (\Re^3,\ \textit{label}_i),\ i = 1, ... M\}$ of one subject in a design pose, called the *model point-set*. We require to match this model point-set to the corresponding observed point-set $\mathcal{Q} = \{q_j \in \Re^3,\ j = 1, ... M\}$ of another subject in a similar pose. Sets $\mathcal{P}$ and $\mathcal{Q}$ have overall spatial distribution similarity and identical cardinality. However, the data set $\mathcal{Q}$ is generally corrupted by distribution errors, due to underlying non-rigid poses and position displacement of feature-point attachment. Scaling is also not uniform. There are neither global nor local affine transformations

between the model and data sets.

Point pattern matching (PPM) is a fundamental, commonly encountered, yet still open problem. Feature-based methods for object recognition, motion analysis and image registration often rely on point pattern analysis to establish a correspondence within two related point-sets. However, work relevant to PPM is largely restricted to rigid objects in Euclidean motions, or perspective matches under affine transformations, or piecewise approaches for non-rigid correspondence [2, 3, 4, 10, 5]. Additionally, uniform scaling is taken into account. In these cases, geometric hashing [11], a "world-view vector" [12], or graph/tree-based representation [13, 14], being based on geometric invariance, might be used to seek an exact one-to-one correspondence in sparse cases under rigid/affine transformations. Constraint satisfaction, in a manner of least-squares optimisation [15], Hausdorff-distances [16, 17, 18] or the well-known heuristic Iterative Closest Point (ICP) algorithm [19, 20], is used to find approximate correspondences and motion estimation for dense point sets. However, in the absence of rigidity, while yet demanding the exact correspondence between sparse point sets required for the PPM problem considered here, the above methods are not applicable.

To deliver an exact correspondence for sparse distribution with underlying non-rigidity, distribution discrepancy and non-uniform scaling, we appeal to a novel spatial index approach that is robust for matching such data. Benefiting from the well-studied multidimensional binary search tree (abbreviated *K*-d tree) techniques, we propose a new *similarity K*-d tree to address the PPM problem presented above.

In the following section, we give a brief review of related *K*-d tree techniques. In Section 3, we give details for the proposed *similarity K*-d tree developed for robust PPM. Experimental results using real MoCap data are given in Section 4. The algorithm analysis

using synthetic data and conclusions are stated in sections 5 and 6.

2 Brief review of *K*-d trees

Many data structures for the tasks of matching and searching multidimensional databases are instances of the general class of *binary space partition* (BSP) trees, such as the *K*-d trees. The *K*-d tree was first introduced by Bentley [21], extended to an *adaptive K*-d tree in [22, 23], and modified with many variants to facilitate implementation of efficient storage and search [24, 25]. K-d trees have been used to solve a number of "geometric" problems in statistics and data analysis, and provide efficient and versatile methods for accessing large databases, for searching nearest- or farthest- neighbours and indexing structured data [13, 20].

Flavours of *K*-d trees differ mainly in the partition strategies for selecting cutting hyperplanes, and in the meaning of interior nodes and leaves. Typically, classical *K*-d trees [21] recursively use axi-orthogonal hyperplanes for *median partition*, to divide more-or-less in half the data set associated with an interior node. Splitting proceeds hierarchically until a desired number of points remains in the leaves. The hyperplanes are chosen in sequence to be perpendicular in turn to each of the K axes in a cyclic order. Such *K*-d trees have excellent depth property, being well balanced under median partitioning.

Adaptive *K*-d trees [22, 23] are constructed in a similar way to classical *K*-d trees with, however, hyperplane directions chosen in a non-fixed order. Taking data tendency into account, the *adaptive K*-d tree hyperplane at each node is chosen to be perpendicular to an axis with the largest data extension. Data is split on that axis into a balanced number of points on each side of the splitting plane by *median partition*, or split through a *mean-*

position of the data set, to achieve a good aspect ratio. For tree construction by *median partition*, the choice of orthogonal axis is recorded at each interior node; while for *mean-position* partition, orthogonal-axis and absolute coordinates of the hyperplane are both stored.

The basic *K*-d tree structures [13, 25] and many of their variants have been devised for general or specific practical applications. However, most partition strategies employed in K-d trees pay little consideration to data distribution. Planes may very possibly split a dense data portion at an early stage of partition, or pass through some points, making the tree shapes very sensitive to point position and producing completely different trees for two similar point-sets with distribution errors. Moreover, tree representation through absolute coordinates cannot lead to consistent tree construction for a corresponding scaled point-set. While a hybrid K-d tree may prove a fruitful line in future research, we concentrate in this paper on developing a new variant $-$ the *similarity* K-d tree $-$ to address robust PPM for sparsely distributed non-rigid data.

3 The new similarity K-d tree

We propose a similarity *K*-d tree with partitioning based on adaptive spatial low-density priority order. Like most *K*-d trees, we use axi-orthogonal hyperplanes to recursively partition a point-set into subsets, but the splitting hyperplane is located at the *bisection of the largest projected interval*, along the orthogonal coordinate $\mathcal{H}_\nu$ forming the *hyperplane-axis*. For a K-dimensional point-set $\mathcal{P}_\nu = \{\mathbf{p}_i \in \Omega^K\}$ of cardinality $|\mathcal{P}_\nu|$ at an interior node ν, the hyperplane-axis $\mathcal{H}_\nu \in K$ of the splitting plane is determined from Eq. (1) by the largest ψ-projected interval $\Delta_{\nu,\psi}$ of the data coordinates,

$$\mathcal{H}_\nu := \max_{\psi \in K} \Delta_{\nu,\psi} \ , \tag{1}$$

where

$\Delta_{\nu,\psi} = \max_{\mathbf{p}_{k_i} \in \mathcal{P}_\nu} \left((\mathbf{p}_{k_{i+1}} - \mathbf{p}_{k_i})_\psi \mid (\mathbf{p}_{k_{i+1}})_\psi \geq (\mathbf{p}_{k_i})_\psi \right)$ is the maximum coordinate interval in the direction ψ and $i = 1 \ldots |\mathcal{P}_\nu|$ is an ordering index. If there is a tie among maximal candidate intervals, Δ_{ν,ψ_1} and Δ_{ν,ψ_2}, in the sense that

$$\frac{|\Delta_{\nu,\psi_1} - \Delta_{\nu,\psi_2}|}{(\Delta_{\nu,\psi_1} + \Delta_{\nu,\psi_2})/2} < 0.1 \ , \tag{2}$$

the choice is biased towards median partitioning for tree balance, and reducing the maximal data extent in direction ψ: $\max_{\mathbf{p}_i, \mathbf{p}_j \in \mathcal{P}_\nu} |(\mathbf{p}_i - \mathbf{p}_j)_\psi|$, to maintain a favourable aspect ratio [23].

In this study, we emphasise tree consistency and robustness not only to distribution error, but also to non-uniform scaling. For this goal, we store at each interior node during tree construction the orthogonal hyperplane-axis identifier $\mathcal{H}_\nu$ (e.g. x, y or z in 3D) and the number of points $|\mathcal{P}_l|$ split to the left subtree. Using the number of left-child points rather than an absolute partitioning coordinate not only improves the tolerance to distribution errors and scaling, but also guarantees a feasible consistent interpretation (refer to Section 4.2) for a corresponding scaled point-set.

An intuitional illustration of the similarity *K*-d tree using the 2D example data is given in Fig. 2. We observe that the hyperplanes are so chosen that points with the nearest normal distances from the cutting planes are maximally distant from those planes. The planes, in effect, locate the sparsest point distributions in the x and y directions. Splitting prioritised by sparse intervals provides a good heuristic for reducing distribution error ambiguities. We summarise the similarity *K*-d tree construction algorithm in Fig. 3.

4 Using the similarity K-d tree for robust PPM

We demonstrate the ability of the proposed similarity K-d tree to solve the sparse PPM problem in MoCap systems as described in the introduction, for a difficult situation of matching captured feature points from human subjects with underlying non-rigidity, as illustrated in Fig. 1. All 3D feature point data were obtained from a commercial marker-based MoCap system - the Vicon 512, composed of 7 cameras with infrared illumination. The measurement accuracy of marker position in 3D is to a level of a few millimetres in a control volume spanning metres in linear extent.

4.1 Point-set alignment

In our motion capture system, the world coordinate system has its origin on the ground, the xy-plane is parallel to the ground, and the z-axis is vertical. In the case of motion capture, the two point-sets $\mathcal{P} = \left\{p_i \in \Re^3\right\}$ and $\mathcal{Q} = \left\{q_j \in \Re^3\right\}$ with identical cardinality M may be obtained in different coordinate systems of distinct location and orientation. They need to be aligned to a consistent coordinate system by centring and rotating. Firstly, the vector centroids $\mathbf{c}_\mathcal{P}$ and $\mathbf{c}_\mathcal{Q}$ are calculated as in Eq. (3).

$$\mathbf{c}_\mathcal{P} = \frac{\sum_{i=1}^{M} \mathbf{p}_i}{M} \quad , \quad \mathbf{c}_\mathcal{Q} = \frac{\sum_{j=1}^{M} \mathbf{q}_j}{M} \tag{3}$$

Secondly, the orientation vectors $\mathbf{o}_\mathcal{P}$ and $\mathbf{o}_\mathcal{Q}$ in the co-centered systems are determined from the weighted sum in the form of second distribution moments,

$$\mathbf{o}_{\mathcal{P}} = \frac{1}{M} \sum_{i} (\mathbf{p}_i - \mathbf{c}_{\mathcal{P}}) \; ||\mathbf{p}_i - \mathbf{c}_{\mathcal{P}}||_2$$

$$\mathbf{o}_{\mathcal{Q}} = \frac{1}{M} \sum_{j} (\mathbf{q}_j - \mathbf{c}_{\mathcal{Q}}) \; ||\mathbf{q}_j - \mathbf{c}_{\mathcal{Q}}||_2 \; . \tag{4}$$

Finally, for alignment, each point in $\mathcal{P}$ and $\mathcal{Q}$ is transformed with respect to the corresponding centroid $\mathbf{c}_{\mathcal{P}},\ \mathbf{c}_{\mathcal{Q}}$ and orientation vector $\mathbf{o}_{\mathcal{P}},\ \mathbf{o}_{\mathcal{Q}}$, through suitable translation and rotation.

4.2 Consistent interpretation of point-sets by similarity K-d trees

We apply the similarity K-d tree construction (Fig. 3) to the aligned model point-set $\mathcal{P}$, to obtain the similarity K-d tree $T_{\mathcal{P}}$ for the model. Left-right traversal outputs leaves in an ordered list of labelled model points. Construction information for the orthogonal-axis and number of points in the left subtree is stored at each interior node during the model tree building.

Having available the model tree $T_{\mathcal{P}}$, we proceed recursively to extract a consistent interpretation tree $T_{\mathcal{P}}(\mathcal{Q})$ for the aligned observed data $\mathcal{Q}$. We follow the structural information embedded in its model tree, as shown in Fig. 4. Left to right traversal of the leaves of tree $T_{\mathcal{P}}(\mathcal{Q})$ then yields an output sequence of the points in the data $\mathcal{Q}$. This point list, together with the output point list from the model tree construction, serve to define corresponding point-pair matches between model set $\mathcal{P}$ and data set $\mathcal{Q}$.

A practical implementation requires only the logical existence of the tree $T_{\mathcal{P}}(\mathcal{Q})$. It need not actually be constructed, as only the output list of left-right traversed leaves is required from the structure, as in Fig. 4.

4.3 Experiment results

Illustrative identification results, all with perfect correspondences, are shown for a number of human subjects of heights 1.2m to 1.8m in Fig. 5 with representative marker attachments. Identified points in the observed data sets are labelled consistently through correspondence with their left-displayed models (labels are omitted in Fig. 5(c) for space limitations). We also link identified points according to the model protocol for meaningful representation of the underlying non-rigid poses. Fig. 5(a) shows a routine lower-limb marker protocol used in clinical gait analysis, and Fig. 5(b) a typical marker set for human character animation. The example in Fig. 5(c) investigated a dense distribution with 51 markers.

The construction of the model K-d tree need only be performed once for a specific point distribution of an identified model set $\mathcal{P}$. Subsequent identifications under similar distributions require only the logical construction and traversal of the data tree for each new point set $\mathcal{Q}$. The current burden of manually identifying numerous subject data sets with the same marker protocol in intensive applications is thereby much reduced.

Factors that support correct identification are close pose similarity and low point density. Generally, tree generation for denser point-sets is sensitive to smaller pose differences. In this case, pose similarity is a stricter requirement than in the case of matching sparser point-sets. For instance, to identify a 23 or 33 feature-point pattern as in Fig. 5(a) and Fig. 5(b), we always achieved a 100% correct identification rate, despite obvious differences in flexion, separation and relative position of legs and feet, and in the case of Fig. 5(b), the opening out, bending and level of the arms. But an extremely dense and complex pattern, such as shown in Fig. 5(c), may easily result in wrong matches for unideal data poses, such as obtained from substantial lowering or raising of the arms

compared to the model.

5 Discussions

5.1 Evaluation and comparison of using K-d trees for PPM

To evaluate the robustness of the similarity K-d tree for PPM, we tested the algorithm on synthetic data in the 3D case and compared the identification performance with two typical K-d trees: the classical and adaptive, the latter in both median partition and mean-position partition versions. We assess the K-d trees with respect to the vital problem of distribution noise in PPM. We generated a set of M random points in a cube of edge $E = 1000$ units as a model set in 3D, for $M = 50, 75, 100$ and 200 points with *average coordinate interval* $\bar{l} = E/M$ of 20, 13.3, 5 and 2.5 respectively. To obtain corrupted observed data, we applied zero-mean Gaussian noise of standard deviation σ to each of the model point coordinates x, y and z, respectively. The parameter $N = 2\sigma/\bar{l}$ is used to obtain a normalised *dimensionless noise* level. Effectively, N indicates the ratio of the coordinate noise-spread $\pm\sigma$ to the average interval $\bar{l}$ in that coordinate. The purpose of expressing noise in dimensionless units is to provide a meaningful normalisation for overlaying identification rates obtained at different axial point densities $1/\bar{l}$. The results shown in Fig. 6 and Fig. 7 are based on the average fraction of correctly identified points in 500 randomly generated trials at each dimensionless noise level N.

Identification rates for 50 and 100 points obtained with similarity K-d trees, classical K-d trees and adaptive K-d trees versus the dimensionless noise level N are shown in Fig. 6. We observe that the similarity K-d tree has the best identification rate over other K-d trees. In particular at $N = 1$, when the noise spread is of the same order as the

average coordinate interval, the similarity tree alone suffers little identification failure. The method takes advantages of 1) a low-density prioritised partition strategy, 2) subset counting during model tree building, and 3) data tree reconstruction consistent with its model tree. All contribute to distribution noise tolerance. The classical K-d tree chooses hyperplanes in a fixed order and utilises median-partitioning, simulating "subset counting", thereby also guaranteeing an inherent consistency of tree building for both model and data sets. It can tolerate data distribution error to a moderate degree. Compared to the similarity K-d tree, the classical tree structure is more sensitive to the data distribution, as it may place a partition plane through a dense data portion without concern to property 1) above. By contrast, conflicting with consistency property 3), adaptive K-d trees consider independently the data tendency of sets $\mathcal{P}_\nu$ and $\mathcal{Q}_\nu$ during the partitioning process, causing ambiguity on the interpretation tree and lowered identification rates. Adaptive K-d trees with mean-partitioning, indicated by absolute coordinates, further degrade tolerance to distribution noise compared to median-partitioning advantaged by "subset counting".

Fig. 7 shows identification rates for various point numbers M randomly generated in a cube of edge E=1000 units, for similarity and classical K-d trees. At a given dimensionless noise level N, we observe a systematic dispersion in the identification curves, favouring higher identification rates at higher point densities. We ascribe this phenomenon to the manner of data partitioning at the tree nodes. Each partition of the node data derived from sorting reduces the number of points in the child partitions, yet still samples the full pre-partition extents in the other $K - 1$ coordinate axes. This leads to a reduced density in those coordinates. Such a divide and conquer strategy associates greater noise tolerance and reduced likelihood of identification error with each new tree level. Higher point numbers within the cube will generate deeper tree structures, hence favouring higher

identification rates at a given dimensionless noise level.

The classical K-d tree shows only a modest improvement of identification rate with increased point density. It benefits from the divide and conquer strategy only in a statistical sense, as median partitioning may occur in locally dense data regions throughout the algorithm. By contrast, the similarity tree maximises the advantage of increased sparseness by dynamically seeking the least dense region among all dimensions for placing the partition. Moreover, the method takes advantage of the increased statistical likelihood of finding wider departures from the mean in the distribution of coordinate intervals, as the number of points M within the cube increases. Our empirical investigation shows that the relative *spatial gain* $\bar{l}_{max}/\bar{l}$, denoting the average maximum coordinate interval divided by the average coordinate interval $\bar{l}$, rises with increasing M (refer also to [26, 27]). The similarity tree therefore achieves a further systematic identification gain, particularly at shallower tree levels. The observed increased dispersion (Fig. 7(a)) in the identification curves for similarity trees over those of the classical trees (Fig. 7(b)) verifies the expected identification advantage of similarity K-d trees over the classical K-d trees at increasing point densities for a given noise level N.

5.2 Complexity

An efficient implementation of the similarity K-d tree PPM algorithm reduces sorting at interior nodes to sorted sublist extraction. The complexity of such an approach is discussed here.

To create the model K-d tree for the set $\mathcal{P}$ of M points, there is an initial requirement of full sorting of the M points along each of the K cartesian coordinate axes, with complexity $O(KM \log M)$. We additionally cross-link the sorted K coordinate lists of the

data, so that given any one coordinate of any point, we can locate the other coordinates of that point in the lists. Cross linking has an $O(KM)$ overhead on storage and processing.

With cross-linked sorted coordinate lists available, a splitting plane at node ν will maintain the initial relative ordering in the split left and right subsets, partitioned at the biggest gap along the splitting plane axis $\mathcal{H}_\nu$. Cross linking will enable partitioning of parent lists into left/right child sublists, reorganised so as to be sorted in each of the coordinates. This allows the hierarchial tree building to continue.

Each complete tree level requires $O(KM)$ complexity to manage the list splitting. An average of $O(\log M)$ tree levels therefore entails a total tree building complexity of $O(KM \log M)$, including initial sorting. This complexity degenerates in a worst case scenario, where just one point splits off at each tree level, to $O(KM^2)$. Using similar arguments, establishing the correspondence for the data set $\mathcal{Q}$ leads to the same average and worst complexity as for model tree construction.

6 Conclusion

We propose a new similarity K-d tree method for sparse PPM with underlying non-rigidity and non-uniform scaling between the model and data sets. The similarity K-d tree emphasises the robustness to data distribution by adaptive partitioning in low-density priority order. Optimal splitting planes prioritised by maximal sparse intervals eschew denser portions. Moreover, data appearing dense in one hyperplane need not appear dense in another hyperplane, so the adaptive choice of hyperplanes maximises the possibility of finding sparse regions in the child partitions. The orientation of the splitting plane and the count of the points in the split subsets are recorded at each interior node, making the tree

construction invariant to scaling. Therefore, construction of the data tree, consistent with respect to the counts and optimal splitting planes of the model tree, yields data points in left-right leaf traversal that robustly correspond to the model points in left-right leaf traversal. Heuristically, the correspondence is least affected by distribution errors and scaling.

The similarity K-d tree has been shown to give superior PPM compared to other K-d trees in synthetic trials, and has been found effective for real-world PPM application in MoCap. Experimental results demonstrate its successful performance in a difficult situation: matching point-sets obtained from diverse human subjects. This method has benefited a number of commercial and medically oriented MoCap projects, automatically identifying various subjects' design pose data with respect to a predefined model. This research enriches the usage of trees for non-rigid PPM.

Acknowledgments

We are grateful to Alan Jones of the Institute of Mathematical and Physical Sciences, Univ. of Wales, Aberystwyth, and to Prof. S. Rao Jammalamadaka of the Dept. of Statistics and Applied Probability, Univ. of California, Santa Barbara, for valuable discussion on the distribution of spacings. All 3D data used in this paper were obtained by a marker-based optical Vicon Motion 512 System, installed at UWA.

References

[1] Special issue on nonrigid registration. *Computer Vision and Image Understanding*, 89 (2-3) (2003).

[2] H. Alt and L. J. Guibas. *Discrete Geometric Shapes: Matching, Interpolation, and Approximation, in Handbook of Computational Geometry*. Elsevier Science Publishers, North-Holland, Amsterdam, (1999) 121–153.

[3] G. S. Cox and G. de Jager. A survey of point pattern matching techniques and a new approach to point pattern recognition. In *Proc. South African Symposium on Communications and Signal Processing*, 1992, pp. 243–248.

[4] B. Li, Q. Meng, and H. Holstein. Point pattern matching and applications - A review. In *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Washington, D.C. USA, 2003.

[5] J. Maintz and M. Viergever. A survey of medical image registration. *IEEE Engineering in Medicine and Biology Magazine*, 2 (1) (1998) 1–36.

[6] D. M. Mount, N. S. Netanyahu, and J. Le Moigne. Efficient algorithms for robust feature matching. *Pattern Recognition*, 32 (1) (1999) 17–38.

[7] F. P. Preparata and M. I. Shamos. *Computational Geometry: An introduction*. New York, Springer-Verlag, 1988.

[8] M. Gleicher. Animation from observation: Motion capture and modtion editing. *Computer Graphics*, 33 (4) (1999) 51–55.

[9] J. Richards. The measurement of human motion: A comparison of commercially available systems. *Human Movement Science*, 18 (5) (1999) 589–602.

[10] B. Li, Q. Meng, and H. Holstein. Articulated pose identification with sparse point features. *IEEE Trans. on Syst., Man, Cybern., Part B Cybernetics*, 34 (3) (2004) 1412–1423.

[11] H. J. Wolfson and I. Rigoutsos. Geometric hashing: An overview. *IEEE Computational Science and Engineering*, 4 (4) (1997) 10–21.

[12] F. Murtagh. A new approach to point pattern matching. *Publications of the Astronomical Society of the Pacific*, 104 (1992) 301–307.

[13] V. Gaede and O. Günther. Multidimensional access methods. *ACM Computing Surveys*, 30 (2) (1998) 170–231.

[14] W. E. L. Grimson, T. Lozano-Perez, and D. Huttenlocher. *Object Recognition by Computer: The Role of Geometric Constraints*. MA: MIT Press, 1990.

[15] D. W. Eggert, A. Lorusso, and R. B. Fisher. Estimating 3-D rigid body transformations: a comparison of four major algorithms. *Machine Vision and Application*, 9 (5/6) (1997) 272–290.

[16] L. P. Chew, M. T. Goodrich, D. P. Huttenlocher, K. Keriera, J. M. Kleinberg, and D. Kravets. Geometric pattern matching under euclidean motion. *Computational Geometry, Theory and Applications*, 7 (1/2) (1997) 113–124.

[17] M. Hagedoorn and R. Veltkamp. A general method for partial point set matching. In *Proc. 13th ACM Symp. on Computational Geometry*, 1997.

[18] D. Huttenlocher, D. Klanderman, and A. Rucklige. Comparing images using the Hausdorff distance. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15 (9) (1993) 850–863.

[19] P. J. Besl and N. D. McKay. A method of registration of 3-D shapes. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14 (2) 1992 239–255.

[20] Z. Zhang. Iterative point matching for registration of free-form curves. *Computer Vision*, 13 (2) (1994) 119–152.

[21] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18 (9) (1975) 509–517.

[22] J. L. Bentley. Multidimensional binary search trees used in database applications. *IEEE Transartions on Software Englneerlng*, 4 (5) (1979) 333–340.

[23] J. L. Bentley. K-d trees for semidynamic point sets. In *Proc. 6th Ann. ACM Symp. on Computational Geometry*, 1990, pp. 187–197.

[24] A. Henrich, H. W. Six, and P. Widmayer. The LSD tree: spatial access to multi-dimensional point and non-point objects. In *Proc. IEEE Int. Conf. on Very Large Databases*, Netherland, Aug. 1989, pp. 45–53.

[25] H. Samet. The design and analysis of spatial data structures. *Addison-Wesley, Reading, MA*, 1990.

[26] J. S. Rao. Spacings in statistical inference. In *Proc. Int. Conf. on Distribution Theory, Order Statistics and Inference*, Santander, Spain, June 2004.

[27] J. S. Rao and M. Sobel. Incomplete Dirichlet integrals with applications to ordered uniform spacings. *Journal of Multivariate Analysis*, 10 (1980) 603–610.

Biography

About the Author - BAIHUA LI received the BS and MS degrees in electronic engineering from Tianjin University, Tianjin, China in 1989 and 1994, respectively, and the PhD. degree in computer science from the University of Wales, Aberystwyth, UK in 2003. She is a Lecturer in the Department of Computing and Mathematics, Manchester Metropolitan University, UK. Her current research interests include computer vision, pattern recognition, human motion tracking and recognition, 3D modelling and virtual reality.

About the Author - QINGGANG MENG received the BS and MS degrees in electronic engineering from Tianjin University, Tianjin, China in 1989 and 1994, respectively, and the PhD. degree in computer science from the University of Wales, Aberystwyth, UK in 2003. He was a Lecturer from 1991 to 1997 in Tianjin University, China, and a researcher from 1997 to 1998 in City University of Hong Kong. Currently, he is a Research Associate in the Department of Computer Science, the University of Wales, Aberystwyth, UK. His research interests include biology/psychology inspired robot learning and control, developmental learning for embedded systems, and computer vision.

About the Author - HORST HOLSTEIN received the degree of BS in Mathematics from the University of Southampton, UK, in 1963, and obtained a PhD in the field of rheology from University of Wales, Aberystwyth, UK, in 1981. He is a Lecturer in the Department of Computer Science, University of Wales, Aberystwyth, UK. His research interests include geophysical gravi-magnetic modeling, motion tracking, computational bioengineering and fluid dynamics.

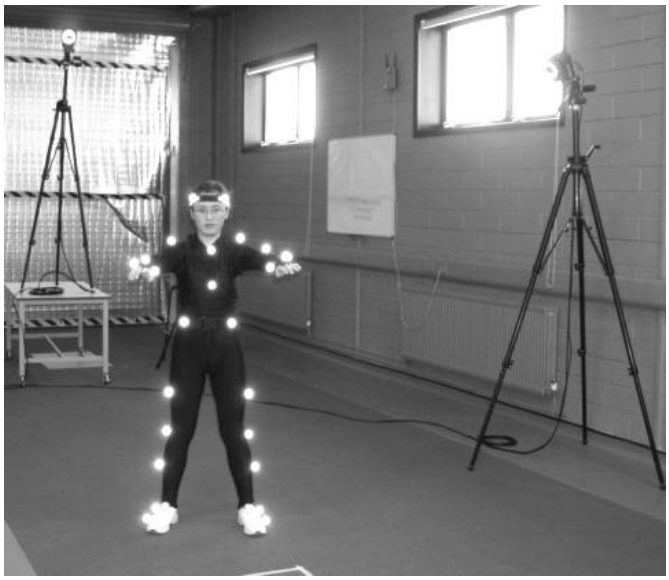
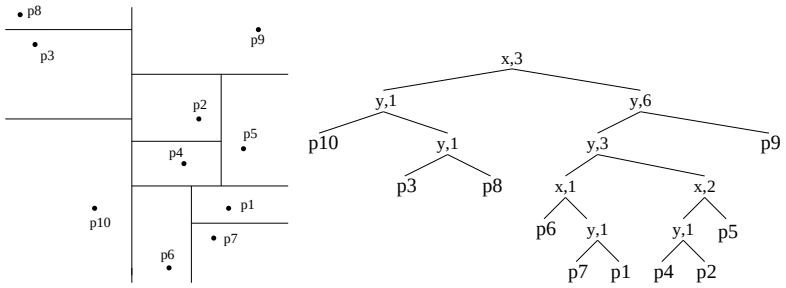


Figure 1: Capture of 3D human pose data in Vicon-512 system.



(a) partition (b) tree structure

Figure 2: A similarity *K*-d tree with 2D example data: p1(6,-4), p2(4,2), p3(-7,7), p4(3,-1), p5(7,0), p6(2,-8), p7(5,-6), p8(-8,9), p9(8,8), p10(-3,-4).

Procedure BuildSimilarityKDTree($\mathcal{P}_\nu$):

1. **if** $\mathcal{P}_\nu$ contains only one point **then**
2. **Output** a leaf node storing this point.
3. **else:**
4. Sort the points $\mathcal{P}_\nu$ along each of the *K* coordinate axes;
5. Determine hyperplane-axis $\mathcal{H}_\nu$ by Eq. (1);
6. Split $\mathcal{P}_\nu$ into left/right subset $\mathcal{P}_l$ and $\mathcal{P}_r$ by locating the hyperplane at the mid-position of the biggest gap perpendicular to the orthogonal-axis $\mathcal{H}_\nu$;
7. Create node *V*=$(V_l, (\mathcal{H}_\nu, |\mathcal{P}_l|), V_r)$, where

   V_l : = BuildSimilarityKDTree($\mathcal{P}_l$);

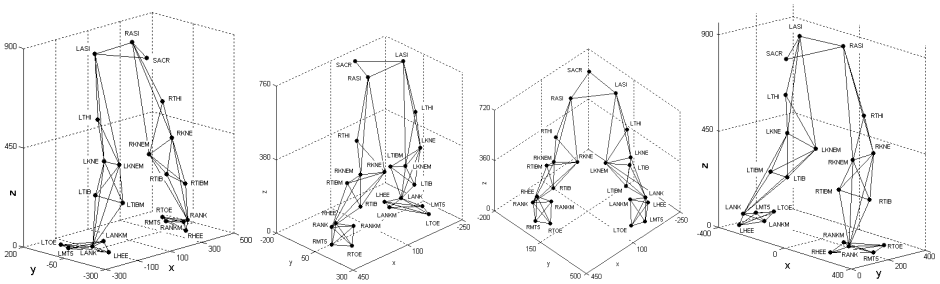
   V_r: = BuildSimilarityKDTree($\mathcal{P}_r$);
8. **return(*V*).**

Figure 3: BuildSimilarityKDTree ($\mathcal{P}_\nu$)

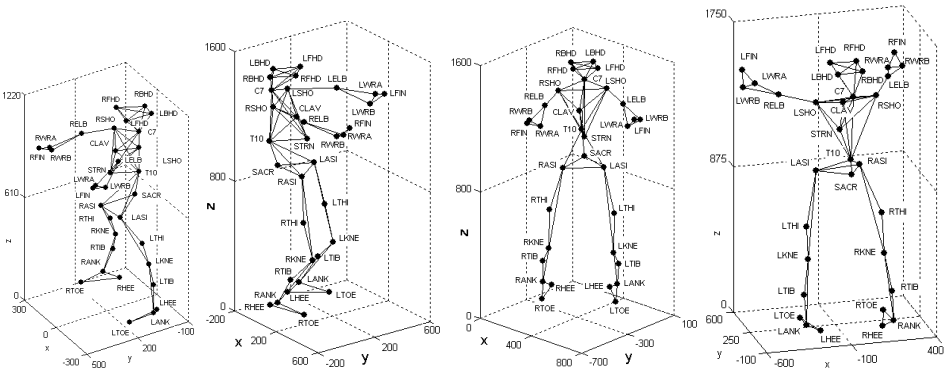
Procedure ConsistentSimilarityKDTree($\mathcal{Q}_\nu$, *V*):

1. **if** $\mathcal{Q}_\nu$ contains only one point **then**
2. **Output** this point.
3. **else:**
4. Let $(V_l, (\mathcal{H}_\nu, |\mathcal{P}_l|), V_r)$=$V$
5. Sort $\mathcal{Q}_\nu$ along the hyperplane-axis $\mathcal{H}_\nu$;
6. Split $\mathcal{Q}_\nu$ into left/right subsets $\mathcal{Q}_l$ and $\mathcal{Q}_r$ with $|\mathcal{Q}_l| = |\mathcal{P}_l|$;
7. ConsistentSimilarityKDTree($\mathcal{Q}_l$,V_l);
8. ConsistentSimilarityKDTree($\mathcal{Q}_r$,V_r);
9. **end.**

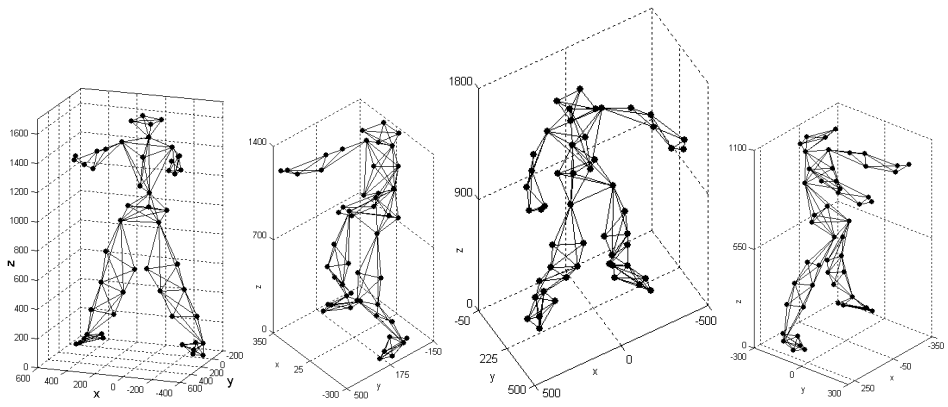
Figure 4: ConsistentSimilarityKDTree ($\mathcal{Q}_\nu$, V)



(a) lower-limb distribution with 23 feature points.

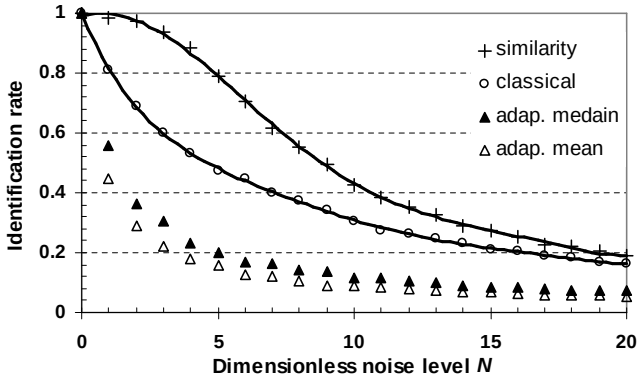


(b) full-body distribution with 33 feature points.

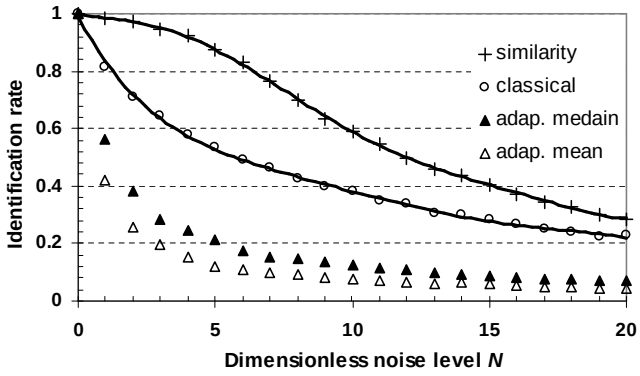


(c) full-body distribution with 51 feature points.

Figure 5: PPM for non-rigid human subjects: model sets (left) followed by three correctly identified observed data sets.

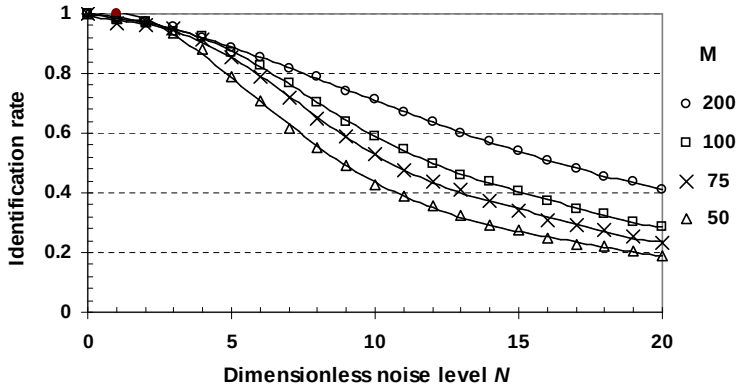


(a) 50 points

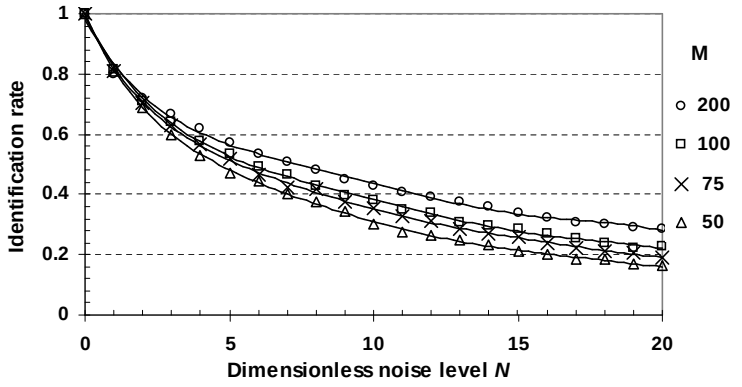


(b) 100 points

Figure 6: Identification rate obtained by similarity, classical and adaptive (median and mean) K-d trees.



(a) similarity K-d tree



(b) classical K-d tree

Figure 7: Identification rate for different point numbers M in a cube.

Contents

List of Figures