Example-Based Methods for Estimating 3D Human Pose from Silhouette Image using Approximate Chamfer Distance and Kernel Subspace

Hui Cao

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Recovering 3D human body pose from a single image remains one of the fundamental problems in computer vision, with potential applications in surveillance, video editing/annotation, human computer interface, and entertainment. The depth ambiguities of 3D-2D projection, part occlusion, clothes variation, and high degrees of freedom of the body pose make this 3D reconstruction problem particularly hard to cope with.

Existing approaches to this problem can be categorized into three groups: *Model-based approaches* fit a 3D human model to an image by minimizing a cost function [31, 50, 20, 29, 38]; *Learning-based approaches* directly infer poses from image features depending on a learned parametric model [2, 55]; *Example-based approaches* store a set of training examples whose 3D poses are known and search examples resembling the input image [5, 46, 60]. However, these approaches become incapable when it is necessary to handle a wide range of 3D human poses and high efficiency is also required. The model-based approaches are time-consuming and sensitive to the initialization of pose. Learning-based approaches are fast but it currently can only deal with a limited set of typical human poses. Example-based approaches may be a good choice for dealing with wide range of 3D human poses, but the issue of high time and memory complexities must be addressed.

In this dissertation, we aim to estimate 3D human pose from a (silhouette) image by combining the strengths of example-based and learning-based approaches. We propose two (time and memory) efficient example-based methods that can effectively obtain candidate poses from a large database. Next, we propose a kernel subspace method to re-rank the obtained candidate poses so that candidates which are exactly (or close to) the real pose of the input image can be assigned higher ranks.

In the beginning chapter we presented an overview of pose estimation problem, the

difficulties involved, our contributions and finally the outline of this thesis.

1.1 Challenging Issues

The appearance of human body in images varies in complicated ways with respect to both subjects and motions. Some of the main challenges in designing a system for estimating human pose from a single image are discussed below.

Visual ambiguities. Background clutter leads to the human body detection extremely hard. The depth ambiguity leads to depth information loss in projecting the body pose onto the image. Self-occlusion makes some parts invisible. These problems must be handled by incorporating prior knowledge.

Appearance variance. Due to differences in clothing and body shape, two different subjects in the same pose seldom appear the same in images. Thus, there is no simple mapping between 3D body pose and image appearance. Furthermore, shading, illumination changes, and deformable clothing change the appearance of the same subject over time. For these reasons, it is hard to use a simple model for characterizing complicated human appearance.

Pose variance. Even a very simple 3D representation of human body has at least 30 *degrees of freedom* (DOF). The enormous volume of the search space makes computing quite difficult.

1.2 Goal: Pose-from-Silhouette

In general, it is difficult to simultaneously handle all the problems listed in the previous section by current techniques. Therefore, some reasonable ways to address the issues in the list are to constrain the problem domain in certain ways, for example, constrained motion or suboptimal estimation.

This thesis focuses on estimating 3D body pose from silhouette. We do not make use of rich visual information such as clothes pattern, skin color or face pattern, to segment and label body parts such as the head, torso, thighs, calves and arms. The background clutter problem is avoided [1], while other possible problems — including pose and appearance variances, self-occlusion and depth ambiguity — will be handled.

As a summary, our goal is to estimate human pose in a generic setting: silhouette, arbitrary pose, and arbitrary viewpoint, with assumptions as:

[1] Background subtraction (e.g., from video) or depth information (e.g., from stereo) is required.

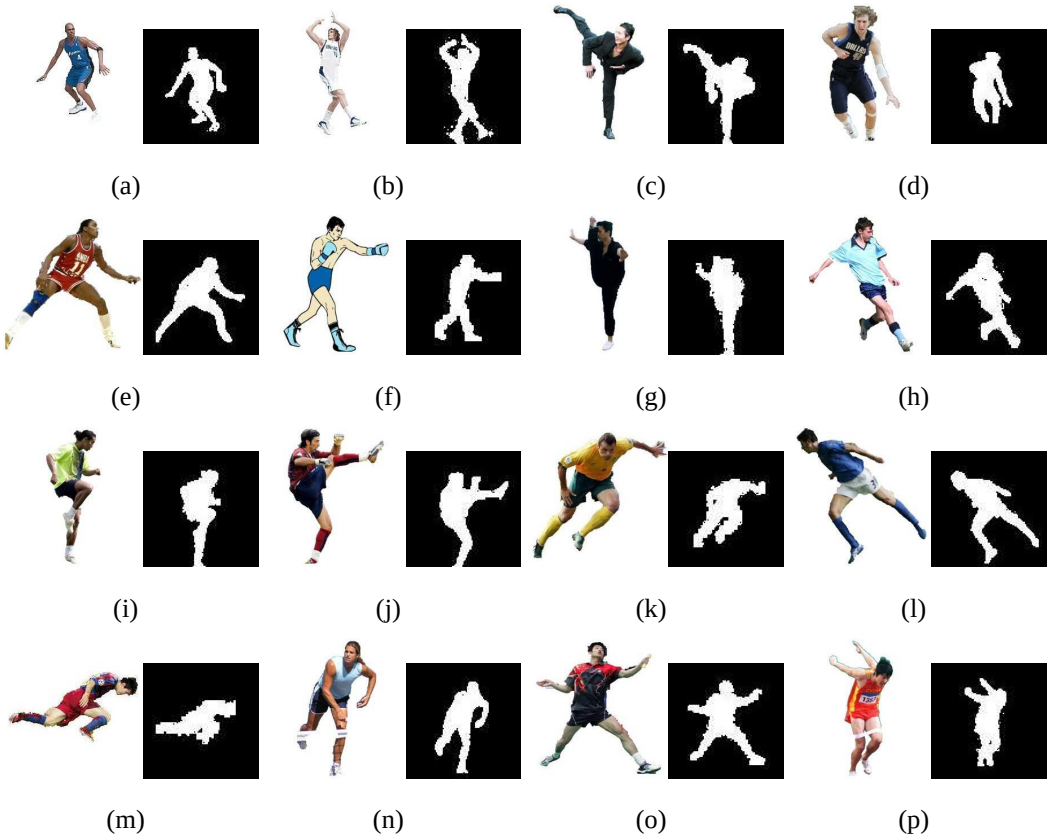


Figure 1.1: Examples of real images involving a variety of sports actions. In each pair of images, the first is the original image and the second is the silhouette used as input for pose estimation.

1. The torso of the target is approximately parallel to the imaging plane;
2. There is no serious external occlusion;
3. The whole body is visible.

Some image examples to be used are shown in Fig. 1.1.

1.3 Contributions

Pose from silhouette is essentially an ill-pose problem due to the high-dimensional state space combined with the unknown factors like parts occlusion, appearance variation and varying view position. In most cases, it is hard (or impossible) to provide a single optimal solution to this problem. The ultimate goal of this dissertation is to provide a few promising

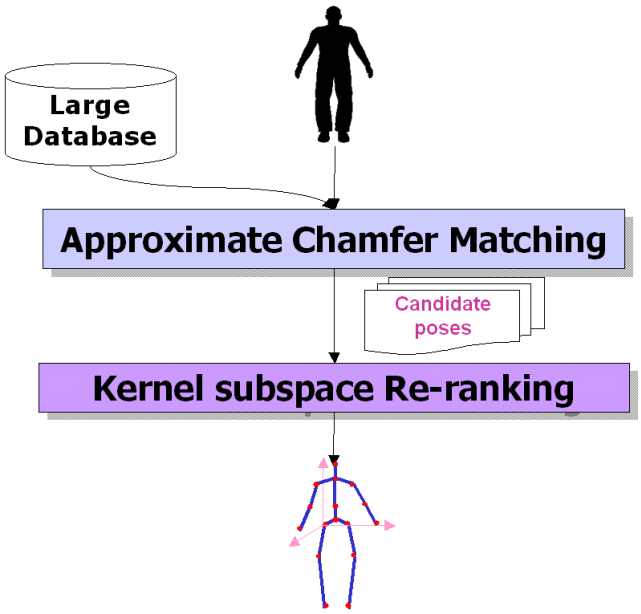


Figure 1.2: Processing flow of the pose estimation approach

estimates which are, hopefully, exactly or close to the ground-truth. We achieve this goal by two-stage processes (see Fig. 1.2). At the first stage, we obtain some candidate poses by searching over a large database of examples. At the second stage, these candidate poses are re-ranked by kernel subspace projection. A few top candidates are selected as the pose estimates.

Our contributions include: (a) constructing a large database; (b) efficient and effective shape matching methods; (c) re-ranking using kernel subspaces that capture the relationship of silhouettes and corresponding poses.

1.3.1 Constructing a Large Database

If the input silhouette well matches some examples in the database which is composed of a large number of human silhouettes annotated with the corresponding 3D body poses, the poses are probably close to the input's pose. Intuitively, this idea is capable of dealing with any complex poses if a large variety of examples being preserved in the database. To create a large database, we first created a medium-size source dataset by means of 3D human character rendering software and various human motion capture data. We then created a larger database containing two millions of examples by means of valid half-body combinations.

1.3.2 Approximate Chamfer Matching

It is usually infeasible for most of good image matching algorithms to deal with a large database. The chamfer distance [8] has proved to be an efficient and effective tool for shape comparison in many computer vision works. However, applying the chamfer distance to the large-scale matching task is still costly. To address this computational issue, we propose two approximate methods to accelerate computing chamfer distance.

The first method, which we refer to as eigen-chamfer matching, utilizes eigenspace approximation to distance transform in computing chamfer distance so that the computation of chamfer distance shifts to a low-dimensional subspace. This new method is able to efficiently complete a linear scan to a two million examples database, while the achieved estimation performance is yet competitive to the exact chamfer distance.

The second method, which is referred to as joint-chamfer matching, utilizes a part-to-whole strategy for searching pose candidates. The half-body candidate poses are first retrieved by means of partial chamfer matching, from which valid half-body combinations are picked out subject to the pre-computed combination constraints. The further evaluation on selected combinations is also efficient as it involves a small number of simple arithmetic (addition and minimization) operations on known half-body distances. This method is computationally extremely efficient and the current implementation can work near real-time.

1.3.3 Kernel Subspace Re-Ranking

The image similarity is not optimal and probably inconsistent to the desired pose similarity. In other word, sometimes when query is different to database images due to body size, clothes, *etc*, irrelevant poses may be overestimated while relevant poses which are close to real pose may be underestimated. Thus, it is necessary to re-rank the candidate poses in combination with other complemental knowledge.

We propose two kernel subspace ranking methods: KPCA-ranking and KCCA-ranking. Kernel Principal Component Analysis (KPCA) [59] and Kernel Canonical Correlation Analysis (KCCA) [35] are used to learn nonlinear subspaces characterizing the underlying structure of image-pose pairs. Depending on the kernel subspace, candidate poses are ranked based on ranking criteria: subspace projection loss for KPCA-ranking and correlation score for KCCA-ranking. The kernel subspace ranking is complemental to image similarly ranking so that the combination of them perform better than either does alone.

1.4 Outline of the Thesis

This section provides an overview of the work presented in the thesis. The remainder of the thesis is arranged as follows.

In Chapter 2, we briefly review previous works on human body pose estimation from a single image. Previous works are classified into three groups: model-based, learning-based and examples-based approaches. The model-based approaches are further divided into top-down and bottom-up according to the strategy of exploring the human pose space.

Chapter 3 describes the human pose database and its construction process. we first create a medium-size source dataset of $14,964$ pose examples from the collected human motion capture sequences. Then, we exploit the half-body combination strategy to enlarge the source dataset into a 130 times larger database consisting of 2 million pose examples. The combination constraint between any two half-body poses is calculated based on their body orientation and pose proximities, thus ensuring the allowable combinations valid poses. In addition to explicitly preserving the large database, we also create a compact half-body database comprised of only half-body examples and the combination constraints.

Chapter 4 describes the first stage of our pose estimation approach. For a given input image, we obtain a set of candidate poses from the database. We propose two approximate chamfer matching algorithms to do this work. The eigen-chamfer matching method [11] uses the eigenspace approximation to speedup the chamfer distance. The joint-chamfer matching method [12] first searches half-body candidates by partial shape matching. Further evaluation are carried out on the valid half-body combinations subject to the combination constraints. Additionally, a refined matching step via adding silhouette cue and image normalization method are introduced. Finally, we present and discuss, for both synthesis and real images cases, the estimation results obtained by the proposed methods.

Chapter 5 describes the second stage of our pose estimation approach. We use the Kernel PCA and Kernel CCA to learn kernel subspaces [13, 14] which can well characterize the nonlinear manifold of image and pose. The ranks by the kernel subspace are combined with the original (image similarly) ranks so that the more likely candidate poses for a given input image can be assigned higher ranks. We present and discuss synthesis experiment and compare the experimental results before and after re-ranking.

The last chapter of the main dissertation is Chapter 6 which summarizes this work and provides speculation on future research directions.

Chapter 2

Related Work

The problem of human body analysis has a 20-year history in computer vision [25], yet remains one of the fundamental unsolved problems. Most works in this area focus on tracking and recognition of human motion from image sequence. Interested readers should refer to [3, 15, 22, 43, 42] for detailed surveys on this topic. In this chapter, we focus on reviewing a new emerging topic of estimation (or inference) of human pose from a single image, which our study belongs to.

Existing works are classified into three groups: model-based, learning-based and examples-based approaches. In the following section, we briefly explain these approaches.

2.1 Model-based Approaches

The model-based approach assumes an explicit parametric model of the human body, and the best human pose is determined based on how well it fits the observed image. The model-based approaches can be further categorized into top-down and bottom-up methods, according to the strategy of exploring the human pose space.

Top-down methods directly explore the high-dimensional human pose space, aiming to find the optimal [64] or several pose hypotheses [38], which minimizes a complex cost function that measures the (dis-)similarity between the view, corresponding to the predicted pose, and the actual observation. The human body models, which are roughly represented by the link-joint models comprised of 2D/3D geometric primitives (*e.g.*,cylinders), are fitted to the image to measure the similarity. With suitable initialization or effective searching means such methods can produce accurate results. Some early work [37] and extensions [68, 7, 49] consider the case where corresponding points between the body model and im-

age are known. In these works, certain simplifications are made, for examples, the scaled orthographic camera projection [68] or restricted poses in the image [7, 49], are assumed and geometric constraints are used to estimate human pose. In other cases, usually certain ”clever” searching or sampling means are required in order to explore the huge pose space efficiently. The strategy of partition sampling are proposed in [39, 31]. Partition sampling strategy divides the state space into M sub-spaces, corresponding to body parts, and applies sampling and evaluation one time in each of sub-spaces.

Bottom-up methods [29, 44, 54, 65, 41] do not use a global body model to fit the observed image, but instead fit the observed image with a set of parts models, which are represented by a single rectangle or feature point, and the geometric constraint between parts. A candidate list of body parts is first detected. These candidates are then pruned and assembled into the best full-body pose with the guidance of global geometric constraints. Felzenszwalb and Huttenlocher [20] consider the problem of fitting a pictorial structure to a background subtraction mask. the global optimal match of the structure could be found efficiently via Viterbi recurrence over a standard discretization of the configuration space. Ioffe and Forsyth [29] used importance sampling to incrementally update a set of candidate assemblies and optimized the combination based on 2D kinematic constraints. Mori *et al.*[45, 52] used an advanced image segmentation approach to build limb and torso detectors, whose outputs were optimally assembled together into the most probable body configurations, by incorporating arbitrary pairwise constraints between body parts, such as scale compatibility, relative position, symmetry of clothing and smooth contour connections between parts.

In general, top-down methods are computationally expensive, and may be easily trapped in local minima; while bottom-up methods are not capable of deducing 3D poses accurately. Recent attempts have been made to combine the top-down and bottom-up search strategies. Lee and Cohen [38] attempted to fit a volumetric 3D model to static 2D images and employed Data-Driven MCMC to find the MAP solution. Various information sources such as face detection, color segmentation, curve fitting, blob and ridge detection are used to form better proposals to facilitate the MCMC search. The work in *et al.*[62, 21] combined weak responses from bottom-up limb detectors based on a statistical model of image likelihoods with a full articulated body model using nonparametric belief propagation.

2.2 Learning-based Approaches

Learning-based methods are appealing because a variety of advanced modern learning techniques can be applied and inferring pose is fast enough for real-time applications. Most methods falling in this category share a common architecture, which extracts features from the image and represents them as vectors, from which predict the possible pose depending on a regression function from the image features space to the human pose space. Image features that have been used include Hu moments of silhouette images [55], concatenated coordinates of sampled boundary points [33], multi-scale edge direction histograms [16], distribution of shape contexts evaluated at sampled boundary points [2], and Harr-like features selected by AdaBoost [71, 51]. Regression functions are learned from a database of training examples (image-pose pairs), whose main goal is to globally or locally represent the relationship between image and pose, and should be able to provide efficient generalization to new images. Example regression functions include Local Weighted Regression [60], BoostMap [6], Mixture of Gaussian Model [56], Bayesian Mixture of Experts [63] and Relevance Vector Machine [2, 69].

Rosales and Sclaroff [55] learned mappings between image silhouettes and 2D body joint configurations. They use 3D motion capture data to generate training image silhouettes and corresponding 2D joint configurations. Joint configurations are clustered in 2D by fitting a Gaussian mixture using the EM algorithm. An inverse mapping is subsequently learned between image silhouette moments and 2D joint configurations, for each joint cluster. New image silhouettes are mapped to joint configurations and the most probable configuration is selected. Agarwal and Triggs [2] proposed to use Relevant Vector Machine (RVM) to infer human pose from the silhouette shape descriptor. Silhouettes are first preprocessed to vector descriptors using the shape context [9] and vector quantized. More recently, an extension of a mixture of RVM was proposed by Thayananthan *et al.*[69].

There is also a large amount of work done on non-linear manifold embedding in low dimensional space. Elgammal and Lee [19] inferred human poses by interpolating representative points in the low-dimensional image manifold for special activity, which is learned using Locally Linear Embedding (LLE) [36]. Tangkuampien and Suter [67] learned pose manifold of human motion via Kernel Principal Component Analysis (KPCA). Similarly, the image (silhouette) manifold was also learnt, Unseen silhouettes are projected through the two manifolds using Locally Linear Embedding (LLE) [36] reconstruction. The output pose is generated by approximating the pre-image (inverse mapping) of the LLE re-

constructed vector from the pose manifold. In the case where corresponding joint points between the body model and image are known, Grochow *et al.*[23, 61] learn the model on different input data, leading to different styles of Inverse Kinematic (IK). The model is represented as a probability distribution over the space of all possible poses with a novel Scaled Gaussian Process Latent Variable Model [36]. This means that the IK system can generate any pose, but prefers poses that are most similar to the space of poses in the training data.

One major weakness of learning-based approaches is that the ability to accurately represent the space of realizable poses depends almost exclusively on the amount and representativeness of the training data. In addition, most existing approaches use features extracted from silhouette images, and can not be easily applied to cluttered situation.

2.3 Examples-based Approaches

Examples-based approaches are built solely on a large database of pose and image feature pairs during training and no prior underlying structure of the pose space is incorporated. Given a query image, the database returns one or many candidate poses with the closest matching feature.

The main issue of examples-based approaches is how to perform a computationally expensive query quickly and accurately. Shakhnarovich *et al.*[60] developed an efficient search algorithm called parameter sensitive hashing (PSH) that uses a set of hash functions to quickly index approximate nearest neighbors of an input image from database. The PSH algorithm is interesting for the hash functions are selected to reflect the similarity in pose space, so that retrieved neighbors are likely to be the similar poses of input image. The solution is further refined using Locally Weighted Regression (LWR). Athitsos *et al.*[6] proposed a distance-approximating embedding algorithm called BoostMap for 3D hand pose retrieval from a large database. They used Adaboost to learn this embedding that maps the chamfer distance into Euclidean space, significantly reducing the computational cost of chamfer distance. Later, they extended the BoostMap algorithm to a cascade version [5] that only applies slower and more accurate approximations to the hardest cases. In the work of Mori and Malik [46], they stored a number of exemplar 2D views of the human body in a variety of different configurations and viewpoints. On each of these stored views, the locations of the body joints are manually marked and labeled. The input image is then matched to each stored view, using the technique of shape context matching in conjunction with a kinematic chain-based deformation model. Assuming that there is a stored view

sufficiently similar in configuration and pose, the locations of the body joints are then transferred from the exemplar view to the test shape. Given the 2D joint locations, the 3D body configuration and pose are then estimated using the algorithm in [68].

Examples-based approaches often have problems when working in high dimensional spaces as it is difficult to create or incorporate enough examples to densely cover the space. This is particularly true for human pose estimation which involves many articular degrees of freedom. In addition, most existing approaches can not be easily applied to cluttered situation.

Chapter 3

Human Pose Database

In this chapter we describe the detail of constructing human pose database. A sufficient database is critical to the performance of pose estimation in our work. We first create a medium-size source dataset of $14,964$ pose examples from a set of human motion capture sequences collected. We then exploit the strategy of half-body combination to enlarge the source dataset into a 130 times bigger database — including nearly 2 million pose examples. The combination constraint defined in terms of body orientation and pose proximities is used to ensure the allowable combinations valid poses. We created two kinds of database. The half-body database preserving only upper- and lower-body data will be used by joint-chamfer method; and the full-body database preserving two million full-body data will be used by eigen-chamfer method. Each element of database consists of 3D body pose and three image descriptors: silhouette, contour and distance transform.

3.1 3D Body Pose and Motion Capture Data

The human body model used in this thesis is comprised of 18 key joints including: head, neck, chest, collars(LR), shoulders(LR), elbows(LR), hands(LR), hip, thighs(LR), knees(LR) and foots(LR). Because we intend not to encode the global orientation of human body, we represent the human body pose in terms of 3D coordinate rather than 3D angle. In addition, we fix the hip joint to be located at the origin point and so hip's 3D coordinate is constantly $[0, 0, 0]^T$. All of other joints are translated accordingly.

The human body motion capture data were obtained from a public website [53]. These motion capture data are stored in the BVH format [1]. There are several captured sequences

[1] See appendix A for detailed information on BVH format.

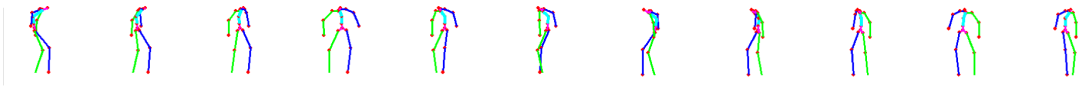


Figure 3.1: A set of body poses (original + 11 rotations) for a particular frame (representing with a stick model). The leftmost pose is the one corresponding to original frame and other poses are ones rotated around the vertical axis.

that depict a variety of different activities: ballet, break dance, walking, kicking, waving, jumping, playing basketball. The total number of frames collected is approximately 13,000 (at 30 frames/second), $1,247$ key frames of which were uniformly picked out as the source motion data to be used for generating pose database. Following the specification of BVH format, we transformed each frame into the body pose format: a 54 dimension vector concatenated by 18 joints' 3D coordinates. In order to obtain for each frame a complete set of poses observed from different viewangles, we calculated 11 new body poses at every 30 degree of rotation around vertical axis. As a result, we collected a full dataset of 1.5×10^4 body poses. Figure 3.1 shows a set of 12 poses derived for a particular frame.

3.2 Half-Body Representation and Combination Constraint

In comparison with the large variation of human body poses, the source dataset created in the previous section is too small to promise good performance of pose estimation. We intend to create a large database of up to millions of body poses by means of half-body pose combination on the source dataset. The human body is divided into two parts: upper-body and lower-body. The upper-body consists of 12 parts: head, neck, chest, collar(LR), shoulders(LR), elbows(LR), hands(LR), hip; and the lower-body involves the remaining 6 parts: thighs(LR), knees(LR) and foots(LR).

Unconstrained half-body combinations from the source dataset will yield approximate 225 million body poses. However, as we know, most of body poses generated by such unconstrained combination are *unrealistic*. Here, the notation of 'unrealistic' means a pose uncommonly observed in the real life, even if it is physically possible pose satisfying the local kinematic constraints. The combination constraint must be defined to prevent such unrealistic half-body combinations. There could be many ways to design such combination constraint. We choose to express the combination constraint simply as a large 2D lookup table C, of which each entry c_{ij} is filled in with a binary value indicating whether the

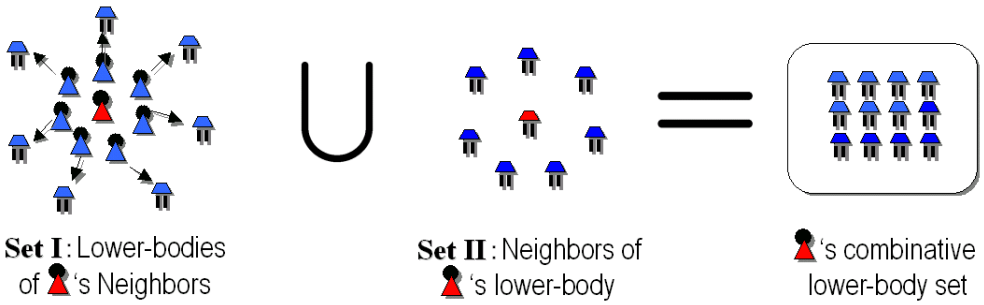


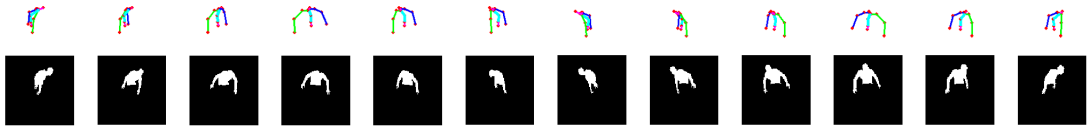
Figure 3.2: The illustration of finding combinative lower-body poses for a particular upper-body pose

upper-body pose i and lower-body pose j are able to form a valid combination.

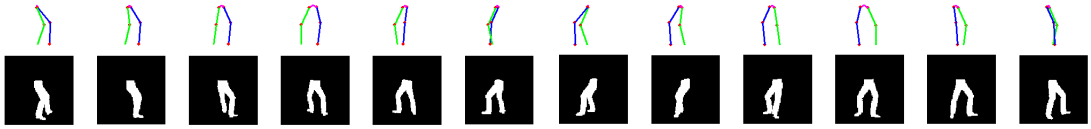
For each upper-body pose p_i^u, we find its combinative lower-body poses relying on two conditions: (1) body orientation[2]proximity and, (2) pose proximity. The first condition ensures that valid half-body combination must have consistent body orientations, and here the maximum allowable deviation between the upper- and lower-body orientations is ± 30 degrees. The second condition intends to confine combinative lower-body poses to be: (1) ones whose upper-bodies are neighbors of p_i^u; or (2) the neighbors of the p_i^u's lower-body pose p_i^l. Figure 3.2 illustrates the idea of searching two neighbor sets. An example $p_j^u(p_j^l)$ is the neighbor of $p_j^u(p_j^l)$ once the Euclidean distance between $p_j^u(p_j^l)$ and $p_j^u(p_i^l)$ is less than a pre-computed threshold distance — the average distance of 95% pairs of successive poses in the source dataset. Because the source dataset is made out of motion capture sequences, successive poses are usually close to each other. The average distance calculated from successive poses is a reasonable threshold to determine whether two poses are similar or not. We further strengthen this value by only adopting 95% smaller distances so as to reduce the effect of outliers(*e.g.*,, sudden motion change). Pseudo code of the algorithm for computing combination constraint is given in Appendix B. Applying this algorithm to the source dataset yielded a combination constraint allowing two million combinations which are less than 1% of unconstraint combinations.

One might ask why we don't use more detailed parts representation, which can yield many more body poses, than the use of half-body representation. The motivation of using half-body representation is that, in the case of using half-body representation we only need to define one constraint between half-body poses because half-body pose itself is a

[2]Since the information of body orientations are unknown for dataset poses, we decide them manually.



(a) upper-body examples



(b) lower-body examples

Figure 3.3: Examples of silhouettes and associated half-body poses representing with a stick model

realistic subpose; whereas in the case of using parts representation we need to define many constraints between related parts so as to ensure parts combination a realistic body pose. It is hard to come up with a reliable and efficient solution for finding constraint of parts combination.

3.3 Two Million Poses Database

In this section, we introduce the detail of constructing a large database based on the aforementioned half-body pose dataset together with the calculated combination constraint. Two kinds of database are created which will be used by different methods of pose estimation described in the next section. The half-body database preserves only the upper-body and lower-body data which intends to implicitly represent the large capability of full-body poses with the combination constraint. On the other side, the full-body database preserves all of two million full-body data through exact combinations. The notation of ”data” denotes the 3D body pose plus three image descriptors: silhouette, contour and distance transform.

We start with the introduction to construction of half-body database. A famous animation and 3D character design package called POSER [18] was used to render half-body poses with an artificial body model. We obtained a full dataset of $14,964$ views, at which the hip of human body model is constantly centered in the image. Because we are only interested in the shape of the projected model, We do not include texture or color information in the rendering results. Figure 3.3 shows examples of rendered silhouettes accompanying with corresponding body poses viewed in 3D space. From each obtained silhouette, We

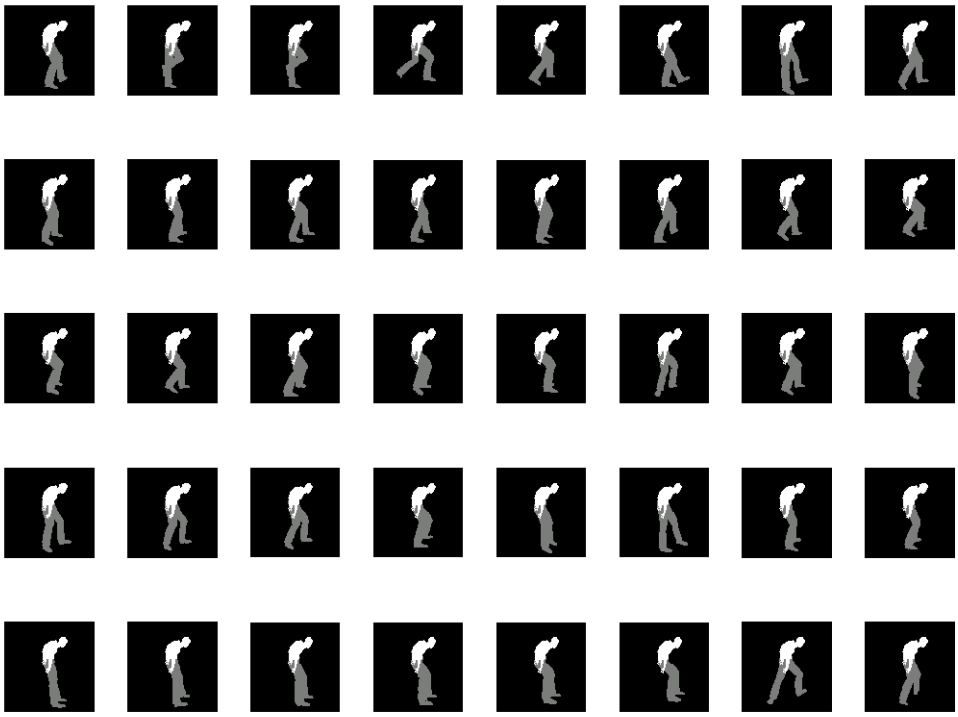


Figure 3.4: Silhouettes of valid combinations for a particular upper-body example

then compute the contour and distance transform (DT) that will be used as input. All of these data — including silhouettes, contours, DTs, and 3D body poses — together with the combination constraint make up of the half-body database.

The full-body database is built based on the half-body database. Subject to the combination constraint, we generate all possible full-body silhouettes by combining upper- and lower-body silhouettes concerned. Figure 3.4 shows the valid combined silhouettes for a particular upper-body example. After all possible silhouettes have been generated, we compute the contours and distance transforms for them. The full-body database preserves these full-body contours and distance transforms. To make database efficient, silhouettes and body poses are preserved in half-body format, and they are combined on demand during the estimation process. In addition, the full-body database does not preserve the combination constraint which is necessary for half-body database, but it needs to preserve an inverse index table from which we can quickly know half-body sources for certain full-body pose example.

Table 3.1 summarizes the space complexities by two databases. Silhouettes and contours are not preserved in image form, instead, we only preserve the XY-coordinate infor-

Table 3.1: Summary of space complexities by half- and full-body databases

DB Type	**3D Body Poses**	**Silhouettes**	**DTs**	**Contours**	**Combination Constraint**	**Inv. Index Table**
Half-Body	39M	49M	500M	10M	214M	—
Full-Body	39M	49M	18.6G	1.3G	—	8M

mation of foreground points — the points falling in the foreground area in silhouette case and the contour points in contour case — plus the number of foreground points. In full-body database, the 3D body poses and silhouettes (of 2 million full-body examples) are not made explicitly but use the half-body counterparts in the half-body database. During the pose estimation stage, the 3D body poses as well as silhouettes are created on demand with the help of inverse index table which indexes a full-body pose example to its corresponding upper-body and lower-body examples. During the pose estimation stage, the half-body database is used directly; while the full-body database which has a unaffordable memory requirement, is used in a low-dimensional representation. The details of using these two database in pose estimation are described in the next chapter.

Chapter 4

Obtaining Candidate Poses using Approximate Chamfer Distance

In this chapter, we describe the first step of our pose estimation approach — for a query image , obtaining a set of plausible candidate poses from the large database (introduced in chapter 3). Basically, it is a large-scale shape matching problem. To efficiently match a large number of shape examples, we propose two approximate chamfer matching methods that significantly reduce the computational cost compared with the normal chamfer matching method. One is an eigen-chamfer matching method which uses the eigenspace approximation to distance transform in computing chamfer distance. The other is a joint-chamfer matching method which does not match (full-body) examples, instead finds plausible half-body candidate poses at first and then evaluates the valid combinations of these candidate poses efficiently by a small number of simple arithmetic operations.

4.1 Introduction

The main difficulty of the silhouette-to-pose task is the one-to-many mapping: a given silhouette can often arise from several poses. Little work is available on resolving this issue. The only work that intends to addresses this issue is found in [56], which uses *specialized maps* to learn the mapping in the form of several sub-functions. The fundamental idea is to generate a finite number of hypotheses through the inverse functions. Extrapolating this idea, examples-based method can generate much more hypotheses. This is exactly what our framework advocates. For a query image, we obtain a set of plausible candidate poses by searching the large database.

There are a number of shape matching methods. Some of them explicitly establish point correspondences between two shapes and subsequently a transformation that aligns the two shapes. These two steps can be iterated, and this is the principle of methods such as iterated closest points (ICP)[10] or shape context matching [9]. However, these methods are computationally too expensive to finish millions of matches in reasonable time. In this work, we employ the chamfer matching method. Chamfer matching is an efficient and effective way for matching shapes, as it does not explicitly make point correspondences between two shapes and it is tolerant to small shape variations because the chamfer distance function varies smoothly when the point locations change by small amounts.

In the remainder of this chapter, The concept of chamfer distance is introduced. Subsequently, the proposed eigen-chamfer and joint-chamfer matching methods are in turn described, following by an optional refined matching step by adding silhouette cue. Afterward, an image normalization method is presented. Finally, we present and discuss the pose estimation results for both synthesis and real image dataset.

4.2 Chamfer Distance

To formalize the idea of chamfer distance, we denote the set $T = \{t_i\}_{i=1}^{N_T}$ whose elements are the coordinates of contour points of a contour example in the database, and the set $Q = \{q_i\}_{i=1}^{N_Q}$ for the query contour. N_T and N_Q denote the number of points in T and Q, respectively. The chamfer distance from T to Q is the mean distance of all points in T to their closest points in Q,

$$d_{cham}^{(T,Q)} = \frac{1}{N_T} \sum_{t \in T} \min_{q \in Q} ||t - q||, \tag{4.1}$$

where $|| \cdot ||$ can be any norm such as the Euclidean or Cityblock. The chamfer distance can be efficiently computed by first calculating the distance transform of the contour image using two-pass algorithm introduced in [8]. In the distance transform DT_Q, the value of each pixel, $DT_Q(p)$, indicates the distance from pixel p to the closest pixel in the contour Q:

$$DT_Q(p) = \min_{q \in Q} ||p - q||. \tag{4.2}$$

More frequently, the following truncated distance transform

$$DT_Q(p) = \min(\min_{q \in Q} ||p - q||, \tau) \tag{4.3}$$

is used. The truncated distance transform has a maximum bound value τ (set by user) and can thus avoid excessive large distance. The truncated distance transform is shown to be robust and have superior performance in practice.

By replacing the *min* operation in (4.1) with the truncated distance transform (4.3), the chamfer distance now becomes a simple look-up operation

$$d_{cham}^{(T,Q)} = \frac{1}{N_T} \sum_{t \in T} DT_Q(t). \tag{4.4}$$

Let $\mathbf{c}_T$ denote the vectorized contour image to which T corresponds. If we use $\mathbf{c}_T$ replace T, (4.4) can be rewritten as an inner-product representation:

$$d_{cham}^{(T,Q)} = \frac{1}{N_T} \mathbf{c}_T{}' \mathbf{dt}_Q, \tag{4.5}$$

where $\mathbf{dt}_Q$ is the vectorization of distance transform DT_Q.

The chamfer distance function is not a metric, as it is not a symmetric function ($d_{cham}^{(T,Q)} \neq d_{cham}^{(Q,T)}$). In order to measure the shape similarity more accurately, it is preferable to making it symmetric, for example by defining the bidirectional chamfer distance as the sum of two chamfer distances from T to Q and from Q to T,

$$D_{cham}^{(Q,T)} = d_{cham}^{(T,Q)} + d_{cham}^{(Q,T)}. \tag{4.6}$$

4.2.1 Precomputing Distance Transforms

Matching a moderate number of shape examples with the bidirectional chamfer distance is computationally cheap compared to other complicated methods. However, if the task is going to perform millions of matches, the accumulated cost is considerably high.

A simple way to reduce computational cost is precomputing distance transforms for contour examples available in the database. That is the reason why we preserved all distance transforms in the database at the stage of constructing database. However, loading all precomputed distance transforms is unfeasible as it requires too much memory, and performing 2 millions matches is still costly [1]. The high time and memory complexities make the way of precomputing distance transforms unsuitable for large-scale shape matching problem in our work. Other better ways must be considered.

[1] We conducted a preliminary experiment in which $10,000$ matches are performed using chamfer distance and also the bidirectional one. The result shows that 2 millions matches may take about 6 seconds and 11 seconds for chamfer distance and the bidirectional one, respectively.

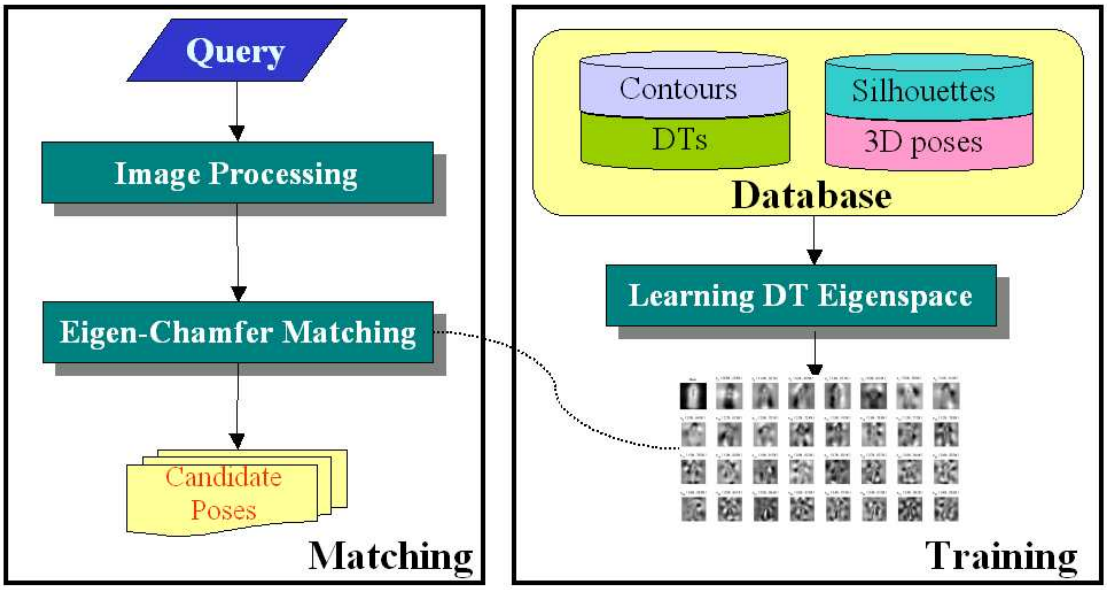


Figure 4.1: Flowchart of eigen-Chamfer matching method.

4.3 Eigen-Chamfer Matching Method

In this section, we present the eigen-chamfer matching method that can match a large number of shape examples efficiently, in both time and memory. At the core of eigen-chamfer matching method is the use of eigenspace approximation to chamfer distance. The eigen-chamfer matching method is partly inspired from the pioneer work of Murase *et al.*[48, 47]. However, our method should be more robust in practice, since the eigenspace approximation here is based on the chamfer distance while the method in *et al.*[48, 47] is essentially an eigenspace approximation to the Euclidean distance.

Since human body shapes are in fact restricted, the corresponding distance transforms should also be restricted. It suggests us to use low-dimensional subspace representation to approximate these distance transforms. By using the eigenspace approximation to distance transform, the resulting eigen-chamfer distance shifts the chamfer distance computation from the image space to a lower-dimensional subspace. Thus, the eigen-chamfer matching method gains significant advantages in both time and memory efficiencies. Figure 4.1 shows the flowchart of the eigen-chamfer method and the involved technique details are described in the remainder of this section.

4.3.1 Inverse Distance Transform Subspace

Principal Component Analysis (PCA) [30] is used to extract the desired number of principal components from the training dataset containing M distance transform vectors, $\mathbf{dt}_1, \ldots, \mathbf{dt}_M$. The eigenvectors, $\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_N]'$, are obtained by solving the eigenvalue problem $\mathbf{\Lambda} = \mathbf{E}'\Sigma\mathbf{E}$, where $\mathbf{\Sigma}$ is the covariance matrix of distance transform:

$$\mathbf{\Sigma} = \frac{1}{M}\sum_{i=1}^{M}(\mathbf{dt}_i - \mathbf{m})'(\mathbf{dt}_i - \mathbf{m}), \tag{4.7}$$

$$\mathbf{m} = \frac{1}{M}\sum_{i=1}^{M}\mathbf{dt}_i. \tag{4.8}$$

$\mathbf{\Lambda}$ is the diagonal matrix with eigenvalues $\lambda_1 \geq \ldots \geq \lambda_N$ of $\mathbf{\Sigma}$ on its main diagonal, so that $\mathbf{e}_j$ is the eigenvector corresponding to the j-th largest eigenvalue.

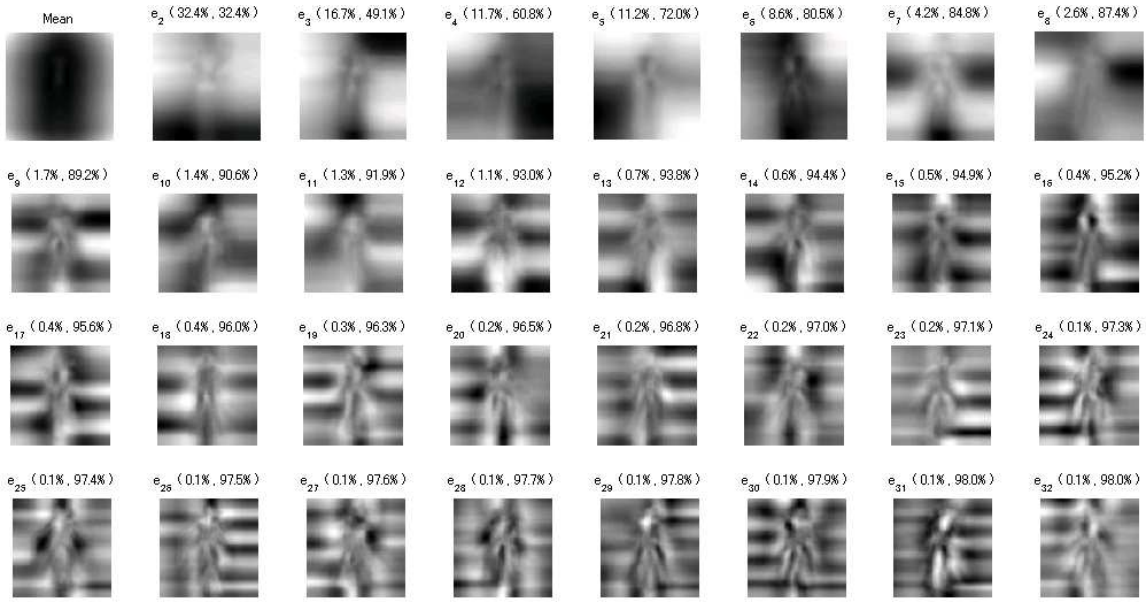
However, the distance transform subspace learned in this way is not appropriate for approximating the distance transform. In distance transform, the pixels far from the contour have large values and important pixels of the contour and its neighbors always have small values. Consequently, the major variance of training data is dominated by those unimportant pixels far from the contour, and hence the obtained subspace is incapable of characterizing the pixels around contour.

To learn a better distance transform subspace which is able to capturing the characteristics of the important pixels, we employ the inverse distance transform:

$$\mathbf{idt} = \tau\mathbf{1} - \mathbf{dt}, \tag{4.9}$$

where $\tau\mathbf{1}$ denotes the vector $[\tau, \ldots, \tau]'$ of the same dimensions as $\mathbf{dt}$. In the inverse distance transform, pixels around contour are assigned to big values while pixels far from contour to low value. Applying PCA to inverse distance transforms is able to extract eigenvectors which characterize the pixels around contour well, and meanwhile neglect the pixels far from contour.

Figure 4.2 illustrates the distance transform and inverse distance transform subspaces learned using the same training data set. For each subspace, the mean vector and top 31 eigenvectors are shown. The numbers in the parenthesis display the percentage of total variance captured by that eigenvector and by the eigenvectors up to now, respectively. It can be clearly observed from the figure that the first few eigenvectors in distance transform subspace are "meaningless" while the first few eigenvectors in inverse distance transform subspace indeed capture the significant information related to human shape.



(a)

(b)

Figure 4.2: Visualization of distance transform eigenspace (a) and inverse distance transform eigenspace (b). In each subfigure, the mean vector and top 31 eigenvectors are depicted (from top-left to bottom-right).

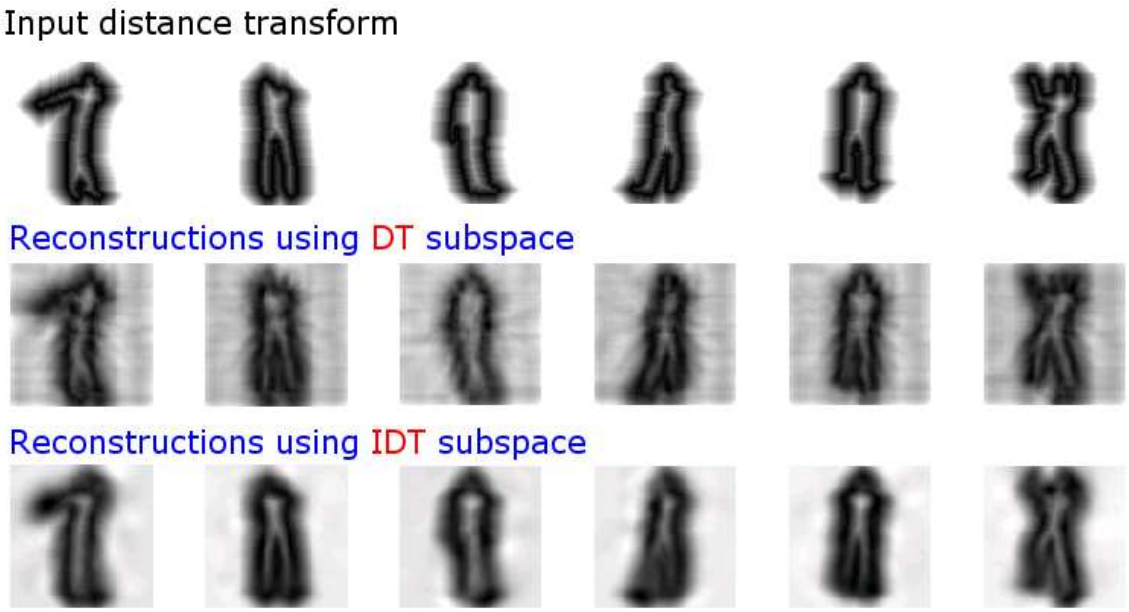


Figure 4.3: Comparison of reconstruction using distance transform subspace and inverse distance transform subspace.

Also, we performed reconstruction experiments by using distance transform subspace and inverse distance transform subspace. Test data are different from the training data. The inverse distance transform subspace uses 67 eigenvectors while the distance transform subspace uses 96 eigenvectors. Some reconstruction examples are shown in Fig. 4.3. The second row of the figure illustrates the reconstruction results using distance transform subspace. There are clearly large noises occurring in the reconstructions and the human body region is highly blurred. The third row gives the reconstruction results using inverse distance transform subspace. The reconstructions are considerably good — looking quite like the smoothed approximations of the input distance transforms.

4.3.2 Eigen-Chamfer Distance

The learned inverse distance transform subspace is used to approximate a new or existing distance transform. Through the linear combination of eigenvectors, a distance transform is expressed by

$$\mathbf{dt} \;=\; \tau\mathbf{1} - \mathbf{idt} \approx \tau\mathbf{1} - (\sum_{i=1}^{k} f_i \mathbf{e}_i + \mathbf{m}), \tag{4.10}$$

where $\mathbf{e}_i$, f_i are the i^{th} eigenvector and eigen coefficient respectively, and $\mathbf{m}$ is the mean vector.

By substituting the eigen approximation of distance transform (4.10) into the chamfer distance (4.5), we get the eigen-chamfer distance as follows:

$$\begin{aligned} d_{cham}^{(T,Q)} &\approx \frac{1}{N_T}\mathbf{c}_T{}' \left(\tau\mathbf{1} - (\sum_{i=1}^{k} f_{qi}\mathbf{e}_i + \mathbf{m})\right) \\ &= \tau - \left(\sum_{i=1}^{k} f_{qi}(\frac{1}{N_T}\mathbf{c}_T{}'\mathbf{e}_i) + \frac{1}{N_T}\mathbf{c}_T{}'\mathbf{m})\right) \\ &= \tau - \mathbf{f}_{eigen}^{Q}{}'\mathbf{d}_{basic}^{T}. \end{aligned} \tag{4.11}$$

In (4.11), the vector $\mathbf{f}_{eigen}^{Q} = [1, f_{q1}, \ldots, f_{qk}]'$ contains a constant value 1 and the eigen coefficients of $\mathbf{idt}_Q$; the vector $\mathbf{d}_{basic}^{T}$= $[\frac{1}{N_T}\mathbf{c}_T{}'\mathbf{m}, \frac{1}{N_T}\mathbf{c}_T{}'\mathbf{e}_1, \ldots, \frac{1}{N_T}\mathbf{c}_T{}'\mathbf{e}_k]'$ contains basic chamfer distances — the chamfer distances from T to mean the and eigenvectors. As a result, the eigenspace approximation to $d_{cham}^{(T,Q)}$ equals to the inner-product of $\mathbf{idt}_Q$'s eigen coefficients and $\mathbf{c}_T$'s basic chamfer distances.

In a similar way, we can derive the eigenspace approximation to $d_{cham}^{(Q,T)}$:

$$\begin{aligned} d_{cham}^{(Q,T)} &\approx \frac{1}{N_Q}\mathbf{c}_Q{}' \left(\tau\mathbf{1} - (\sum_{i=1}^{k} f_{ti}\mathbf{e}_i + \mathbf{m})\right) \\ &= \tau - \left(\sum_{i=1}^{k} f_{ti}(\frac{1}{N_Q}\mathbf{c}_Q{}'\mathbf{e}_i) + \frac{1}{N_Q}\mathbf{c}_Q{}'\mathbf{m})\right) \\ &= \tau - \mathbf{f}_{eigen}^{T}{}'\mathbf{d}_{basic}^{Q}, \end{aligned} \tag{4.12}$$

where $\mathbf{f}_{eigen}^{T}$=$[1, f_{t1}, \ldots, f_{tk}]'$ and $\mathbf{d}_{basic}^{Q} = [\frac{1}{N_Q}\mathbf{c}_Q{}'\mathbf{m}, \frac{1}{N_Q}\mathbf{c}_Q{}'\mathbf{e}_1, \ldots, \frac{1}{N_Q}\mathbf{c}_Q{}'\mathbf{e}_k]'$ are $\mathbf{idt}_T$'s eigen coefficients and $\mathbf{c}_Q$'s basic chamfer distances, respectively.

The efficiency achieved by eigen-chamfer distance is in terms of both time and memory. Basic chamfer distances $\mathbf{d}_{basic}^{Q}$ in $d_{cham}^{(Q,T)}$ and eigen coefficients $\mathbf{f}_{eigen}^{Q}$ in $d_{cham}^{(T,Q)}$, are computed only once before matching. The computations of basic chamfer distances $\mathbf{d}_{basic}^{T}$ in $d_{cham}^{(T,Q)}$ and eigen coefficients $\mathbf{f}_{eigen}^{T}$ in $d_{cham}^{(Q,T)}$, for all examples in database, can be finished at offline stage. Thus, the major computation of eigen-chamfer distance only involves the inner-product as many operations as the dimensions of subspace. The eigen-chamfer distance is memory efficient because it only needs the preservation of eigen coefficients and basic chamfer distances, both having the number of as the dimensions of subspace.

If we use a single subspace to characterize the whole distance transforms in the database, the eigen-chamfer distance would be ineffective and inefficient. A better solution is considered in our work. We first divide the large training set into serval subsets and afterward

apply PCA to each subset for a separate subspace. Consequently, only a small number of eigenvectors are needed to describe the subset. For each distance transform in the database, we choose the optimal subspace which minimizes the information loss after subspace projection. The ID of the optimal subspace and corresponding eigen coefficients are preserved.

4.4 Joint-Chamfer Matching Method

In this section, we propose an alternative way to approximate chamfer distance. This method, which we refer to as joint-chamfer, exploits the part-to-whole matching characteristic of the chamfer distance to quickly search relevant half-body candidates over the half-body database. Then, it finds out a few half-body combinations which resemble the input image best.

The joint-chamfer method is comprised of three steps as follows:

Half-body Candidate Retrieval Calculate the chamfer distance $d_{cham}^{(U,Q)}$ (to the input image) for each example U in the upper-body database. A certain number of top ranked examples with smaller distances are marked as upper-body candidate poses. In a similar way, a certain number of lower-body candidate poses are selected, too.

Selecting Valid Combinations From these half-body candidates, select valid half-body combinations which satisfy the combination constraints (Sec. 3.2).

Selecting Candidate Combinations For each valid combination, calculate approximately the bidirectional chamfer distance based on known half-body chamfer distances and distance transforms. Select a few top ranked combinations with smaller distances as pose candidates.

Figure 4.4 shows the flowchart of joint-chamfer method and the details of these steps are described in the following.

4.4.1 Finding Half-body Candidates via Partial Shape Matching

For a given input image Q, we use the chamfer distances, $d_{cham}^{(U,Q)}$ and $d_{cham}^{(L,Q)}$, to assign scores respectively to upper- and lower-body examples in the half-body database. The chamfer distance is an effective tool for finding relevant half-body candidates since it is capable of matching a partial contour to a whole contour without segmenting the image into parts.

The chamfer distances $d_{cham}^{(U,Q)}$ and $d_{cham}^{(L,Q)}$, are respectively the mean distances of all

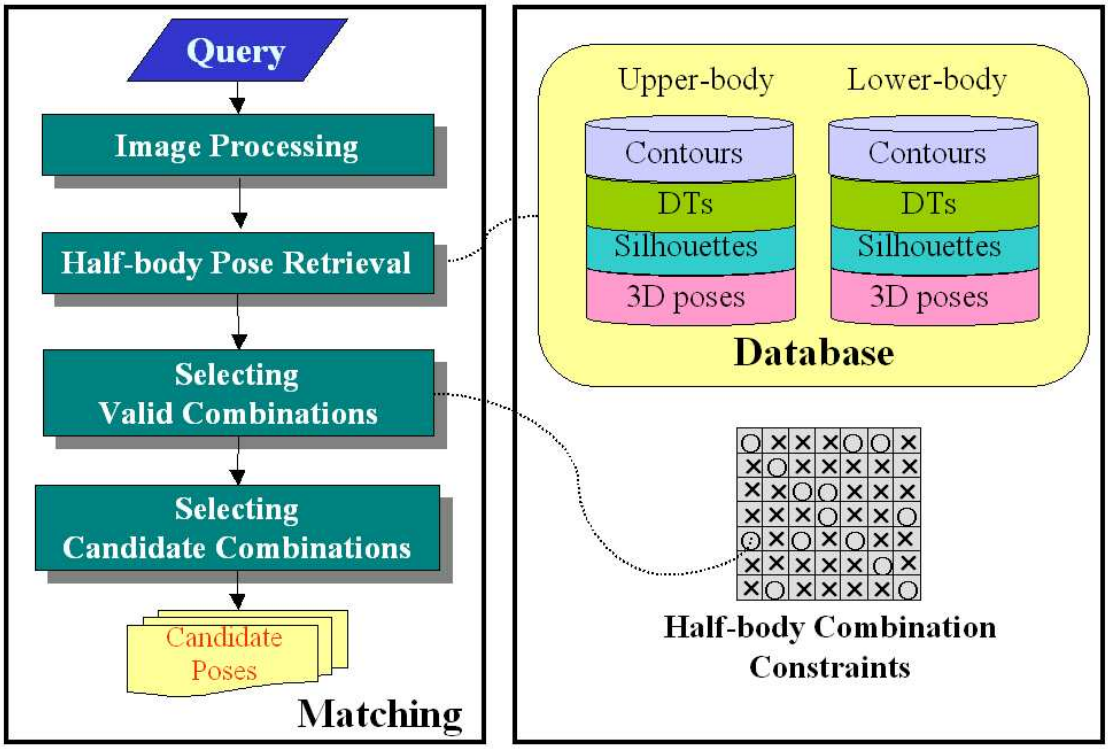


Figure 4.4: Flowchart of joint-Chamfer matching method.

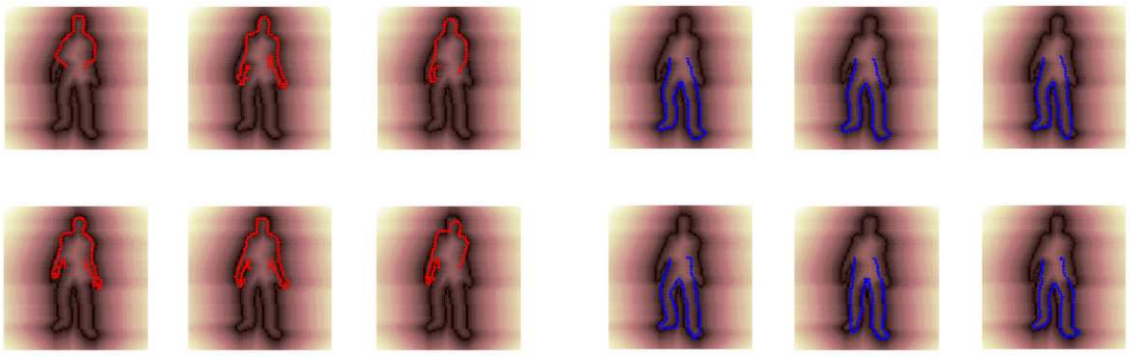
points in upper-body and lower-body contours to their closest points in the input contour,

$$d_{cham}^{(U,Q)} = \frac{1}{N_U} \sum_{u \in U} DT_Q(u), \tag{4.13}$$

$$d_{cham}^{(L,Q)} = \frac{1}{N_L} \sum_{l \in L} DT_Q(l), \tag{4.14}$$

where N_U and N_L denote the number of points in U and L, respectively. The less the chamfer distance is, the better the half-body contour matches the (part of) input contour. This ensures us to find the relevant half-body candidates. However, the candidate set is unavoidably mixed with irrelevant half-body examples. For example, when certain upper-body examples in the database happen to match the lower-body part of the input contour, these upper-body examples can be wrongly thought of as upper-body candidates. Fortunately, the subsequent matching processes are able to prune these irrelevant half-body candidates. Therefore, in the current step it is simply to obtain sufficient number of relevant half-body candidates, regardless of how many irrelevant ones are mixed in.

We compute chamfer distances for half-body examples in the database. The resulting chamfer distances are sorted in ascending order and a certain number of upper-body ex-



(a) (b)

Figure 4.5: Six retrieved (a) upper-body and (b) lower-body candidates overlapped on the distance transform image of an input contour

amples as well as lower-body examples are marked as half-body candidates. Currently, 800 upper-body candidates and 300 lower-body candidates are considered in our work [2]. Figure 4.5 shows half-body candidates retrieved for a given input image. In the figure, the background is the distance transform of the input images, the red/blue contours overlapped are, respectively, top 6 upper-/lower-body candidates with the smallest chamfer distances. Because only chamfer distance is used, short contours are likely to be more favored (see the first upper-body candidate for example).

4.4.2 Selecting Valid Combinations under Combination Constraint

From the retrieved half-body candidates, we select valid half-body combinations subject to the combination constraint. The selection process is very easy. For a possible combination of upper-body u and lower-body l, we check the corresponding entry ct_{ul} in the constraint table $\mathbf{CT}$, if ct_{ul} equals 1 then we consider this combination a valid one. In general, from all 800×300 unconstrained combinations, about $10,000 \sim 30,000$ valid combinations remain for further evaluation.

4.4.3 Selecting Candidate Combinations by Distance Combination

This step aims to choose a few candidate combinations from tens of thousands combinations. The bidirectional chamfer distance is used to evaluate each valid combination.

[2]The upper-body that has larger pose variation needs much more candidates to deal with.

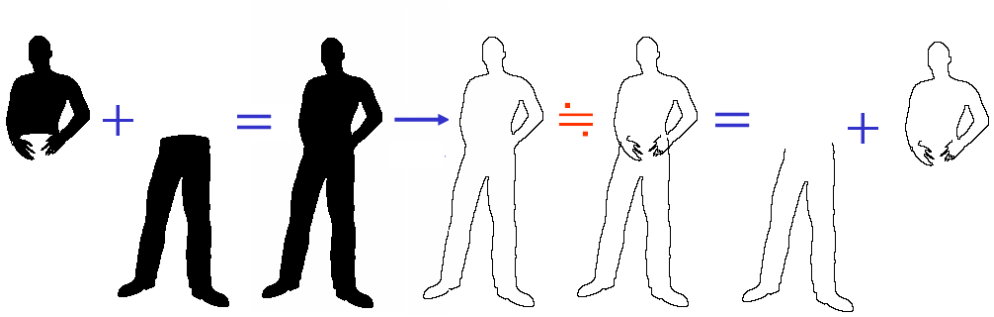


Figure 4.6: Contour approximation. The contour extracted after silhouette combination can be approximated by the union of half-body contours. The superfluous hands' contour is a small portion compared to whole contour.

Computing exactly the bidirectional chamfer distance for tens of thousands combinations is costly as it involves a sequence of time-consuming operations — silhouette generation→contour extraction→distance transform computation .

To avoid the time-consuming image operations, we resort to an approximate way which computes the bidirectional chamfer distance without performing image operations. Figure 4.6 shows an example of contour approximation. In the figure, the contour extracted after silhouette combination equals approximately the union of half-body contours. The contour approximation is reasonable because overlapping is generally not severe, and sometimes exact equation holds true when two half-body silhouettes are separate. Also, the contour approximation simplifies the calculation of distance transform. Thus the approximation of contour combination J as well as the approximation of distance transform DT_J are defined as,

$$J \simeq U \cup L, \tag{4.15}$$

$$DT_J(q) \simeq \min(DT_U(q), DT_L(q)), q \in Q. \tag{4.16}$$

The value of each pixel in DT_J equals the closest distance to contour J, thus equaling the minimum value between DT_U and DT_L.

Using the above approximations, the bidirectional chamfer distance can be computed efficiently. Substituting (4.15) and (4.16)into the bidirectional chamfer distance (4.6), gives

$$\begin{aligned} D(Q,J) &= d_{cham}^{(Q,J)} + d_{cham}^{(J,Q)} = \frac{1}{N_Q}\sum_{q\in Q} DT_J(q) + \frac{1}{N_J}\sum_{j\in J} DT_Q(j) \\ &\simeq \frac{1}{N_Q}\sum_{q\in Q}\min(DT_U(q),DT_L(q)) + \frac{1}{N_U+N_L}\Big(\sum_{u\in U} DT_Q(u) + \sum_{l\in L} DT_Q(l)\Big). \end{aligned} \tag{4.17}$$

(4.17) shows that the bidirectional chamfer distance can be efficiently computed by a small number of *min* and *addition* operations. The approximate $d_{cham}^{(J,Q)}$ can be obtained at near-zero cost because it involves two un-normalized half-body chamfer distances, $\sum_{u \in U} DT_Q(u)$ and $\sum_{l \in L} DT_Q(l)$, which have been already calculated at the half-body retrieval stage. The approximate $d_{cham}^{(Q,J)}$ involves a small number of min operations which equals the point number of the input contour.

We sort valid half-body combinations based on the approximate chamfer distances. A few top ranked combinations, that is the full-body pose, are selected as candidate poses.

4.5 Refined Matching via Silhouette Cue

As an optional step, we may introduce the silhouette distance to refine the matching results obtained by the eigen-chamfer or joint chamfer methods. The silhouette combination is computed by pixel-wise logical OR operation on two half-body silhouettes. The distance between the combined silhouette S^J and the input silhouette S^Q is defined in terms of the ratio of un-overlapping area defined as

$$D_{silhouette}^{(Q,J)} = 1 - \frac{AREA(S^J \bigwedge S^Q)}{AREA(S^Q \bigcup S^J)}, \tag{4.18}$$

where $AREA()$ is the operator for computing the area of ROI.

We use the unified distance $D_{cham}^{(Q,J)} + D_{silhouette}^{(Q,J)}$ to re-rank the candidates.

4.6 Normalization of Input Image

The chamfer distance function is not invariant against image transformations such as translation, rotation or scale. Therefore, each of these cases needs to be handled by searching over the parameter space. In order to match a large number of templates efficiently, hierarchical search methods have been already suggested. In this work, we employ a regression-based method to normalize the input image before matching so that the normalized input is likely to be aligned with the database images.

First, the foreground within the input image is scaled in accordance with database images. We assume that for all images in the database, their foreground areas can be approximately inferred from the invariant shape descriptor. In this work we formulate this assumption by a multivariate linear regression model and choose 10 Hu moments $\{h_i\}$

[27] as regression input:

$$\hat{A} = \sum_{i=1}^{10} w_i h_i, \tag{4.19}$$

where $\hat{A}$ is the inferred area and weight parameters w_i are learned from database images using the algorithm of stochastic ridge regression [17].

From the Hu moments derived from a query image, the scaling ratio is computed by the squared root of the inferred area $\hat{A}$ using the above model divided by the original foreground area A. Next, the scaled foreground is translated to the center of the image as database images. The overall formula for normalization is

$$\vec{\hat{p}} = \left[(\vec{p} - \vec{m}) * \sqrt{\frac{\hat{A}}{A}} + [50\ 50]^T \right]_0^{99}. \tag{4.20}$$

In (4.20), $\vec{p}$ and $\vec{m}$ respectively denote the original coordinates of the foreground point and the mean coordinate of the foreground in the query image. Operator $[\cdot]_0^{99}$ guarantees transformed coordinates within the database's image size: 100×100.

Sometimes this normalization method may lead to inaccurately scaled results. We thus propose a multiscales extension to determine scaling parameter more carefully by traversing over a few scaling ratios around the inferred one, that is, $\sqrt{\frac{\hat{A}}{A}} * (1 \pm x)$ where typical values of x can be 0.05, 0.1, and so on. The refined matching (Sec. 4.5) is repeated for the multiscales normalization of the query image, and among obtained multiple matching scores we can take either the minimum or the mean value.

4.7 Experiments

We have conducted a couple of experiments on both synthetic and real image datasets. In this section, the performance of our methods are demonstrated. The experiment on synthetic dataset provides a quantitative evaluation to our methods as ground-truth poses for synthetic data are accessible. The experiment on real image dataset demonstrates the generalization ability of our methods because the real image data involve a wide range of pose variation, and moreover, variations in clothing, body size and view angle further complicate the estimation task.

All methods evaluated in the experiments are: joint-chamfer match method and three kinds of eigen-chamfer match methods:

Table 4.1: The summary of experimental data

	Number	Data Source	Selection Method
Training dataset	3000	source dataset $(14,964)$	uniform
Testing dataset I	2000	database (2 million)	random
Testing dataset II	2000	dilations of testing dataset I	—
Testing dataset III	155	real images	google search

1. *eigen-chamfer F* using the forward eigen-chamfer distance $d_{cham}^{(Q,T)}$. Eight subspaces are used, each containing $63 \sim 67$ eigenvectors, whose accumulated variances are 95% of the total variance in the training set;

2. *eigen-chamfer R* using the reverse eigen-chamfer distance $d_{cham}^{(T,Q)}$. Only one of eight subspaces, containing 67 eigenvectors, whose accumulated variance is 95% of the total variance in the training set;

3. *eigen-chamfer BI* using the bidirectional eigen-chamfer distance $d_{cham}^{(Q,T)} + d_{cham}^{(T,Q)}$. Eight subspaces are used, each containing $35 \sim 37$ eigenvectors, whose accumulated variances are 90% of the total variance in the training set.

The experimental data are summarized in Table 4.1. For eigen-chamfer methods, training data are 3000 samples selected uniformly from the source dataset $(10,964)$. The testing dataset has three parts. Testing set I contains 2000 samples selected randomly from the large database. In order to investigate the performance when images are different from the database images, the testing dataset II contains the one-pixel dilations of the samples in testing dataset I. Testing dataset III contains 155 real images collected using Google's image search engine. These real images involve a variety of sports: basketball, dance, football, tennis, figure skating, *etc*.

4.7.1 Evaluation Criteria

The estimation performance is evaluated with respect to: precision, computational time and memory requirement. The precision of estimation is assessed in terms of: (1) HIT-1, (2) HIT-5 and (3) ROS (rate of success), when query's ground-truth is known; and by subjective assessment when query is a real image.

The definitions of HIT-1, HIT-5 and ROS are given as follows:

$$\text{HIT-1} = \frac{\sum_{i=1}^{N_{test}} (ground\ truth \in \{Rank_1, \ldots, Rank_K\})}{N_{test}}, \quad (4.21)$$

$$\text{HIT-5} = \frac{\sum_{i=1}^{N_{test}} (one\ of\ 5NNs \in \{Rank_1, \ldots, Rank_K\})}{N_{test}}, \quad (4.22)$$

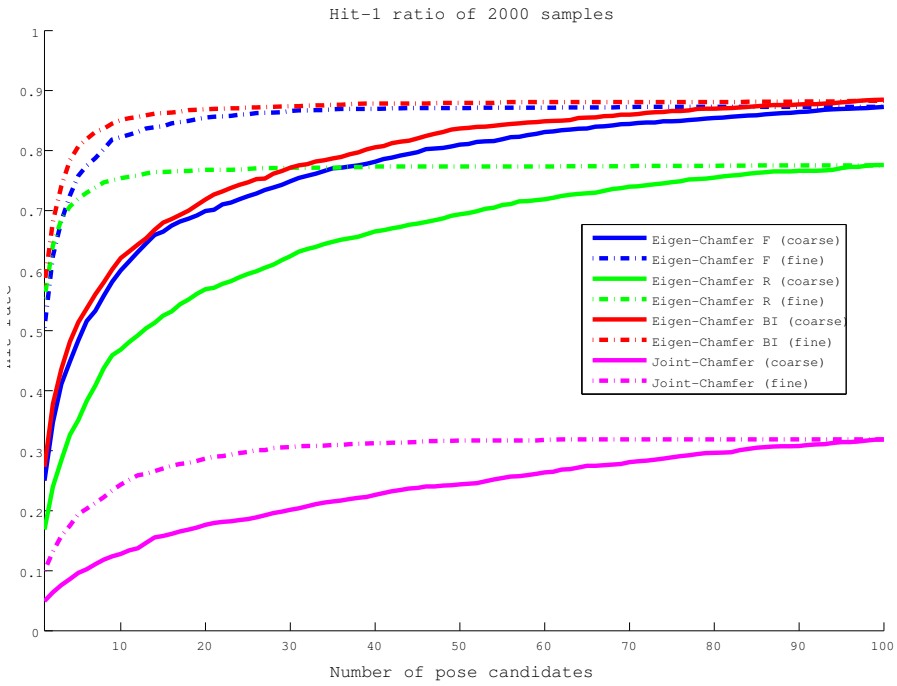
$$\text{ROS} = \frac{\sum_{i=1}^{N_{test}} \frac{1}{K} \sum_{k=1}^{K} (||Rank_k - ground\ truth|| < \epsilon)}{N_{test}}. \quad (4.23)$$

In the above definitions, if the condition inside $(\cdot)$ is satisfied, $(\cdot)$ returns 1, otherwise 0; N_{test} is the number of test data. HIT-1 calculates the ratio of the groundtruth's occurrence among candidates till the K^{th} rank. HIT-5 relaxes HIT-1's condition from "groundtruth's occurrence" to "occurrence of one of 5 neareast neighbors". In the definition of ROS (4.23), $||\cdot||$ is the Euclidean distance between the ground truth and candidate pose. The threshold ϵ is set using the average pose distance calculated from 95% successive example pairs in the source dataset. The average pose distance is an appropriate threshold to judge whether or not two poses are similar — since the source dataset is made from the motion capture sequence and thus most of successive examples are usually close to each other. Though, there is ambiguity in using Euclidean distance to measure the similarity for high-dimensional data like 3D pose. It is necessary to replace Euclidean distance with other optimal distance.

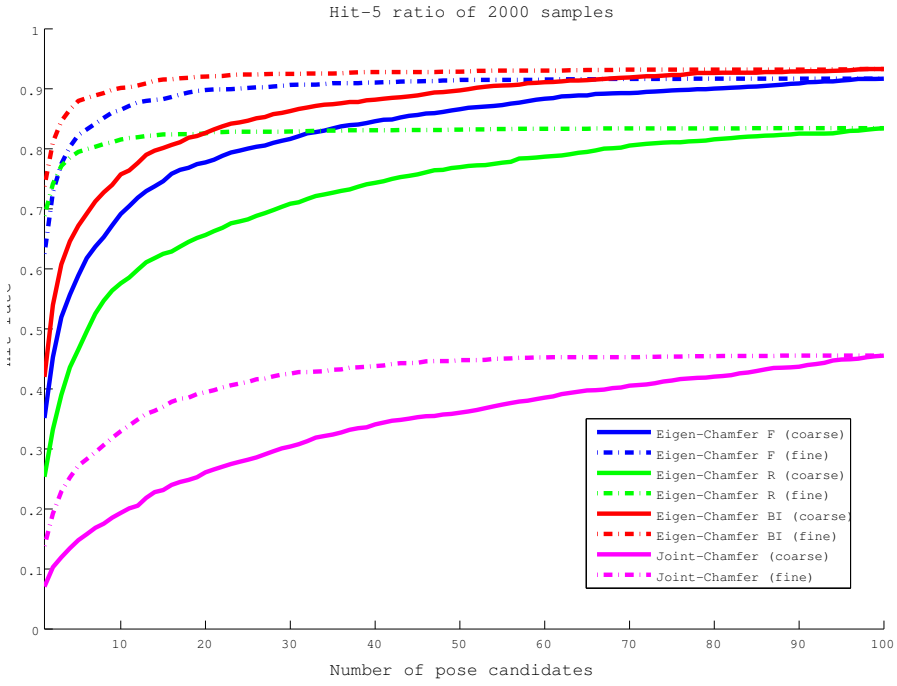
4.7.2 Experimental Results on Synthetic Dataset

We conducted experiments on test dataset I and test dataset II. Top 100 candidates ranked by each chamfer method are considered as the coarse estimation results. These candidates are re-ranked by taking account of silhouette cue (Sec. 4.5). The re-ranked candidates are considered as the fine estimation results. We evaluated HIT-1, HIT-5 and ROS from both coarse and fine estimation results.

Figures 4.7 and 4.8 show the resulting graphs of HIT-1 and HIT-5. In any case, the rates incurred by the eigen-chamfer BI (red) are best and the rates by the joint-chamfer (pink) are worst. For testing dataset I, the rates after adding silhouette cue always improve the rates by chamfer distance only. However, for testing dataset II, the rates after adding silhouette cue dropped down significantly. Thus, the silhouette cue would lead to negative effect when input images are different to database images. In the case of testing dataset I, the eigen-chamfer F (blue) outperformed the eigen-chamfer R (green). The reason might be that the eigen-chamfer F uses eight subspaces to approximate distance transform, then

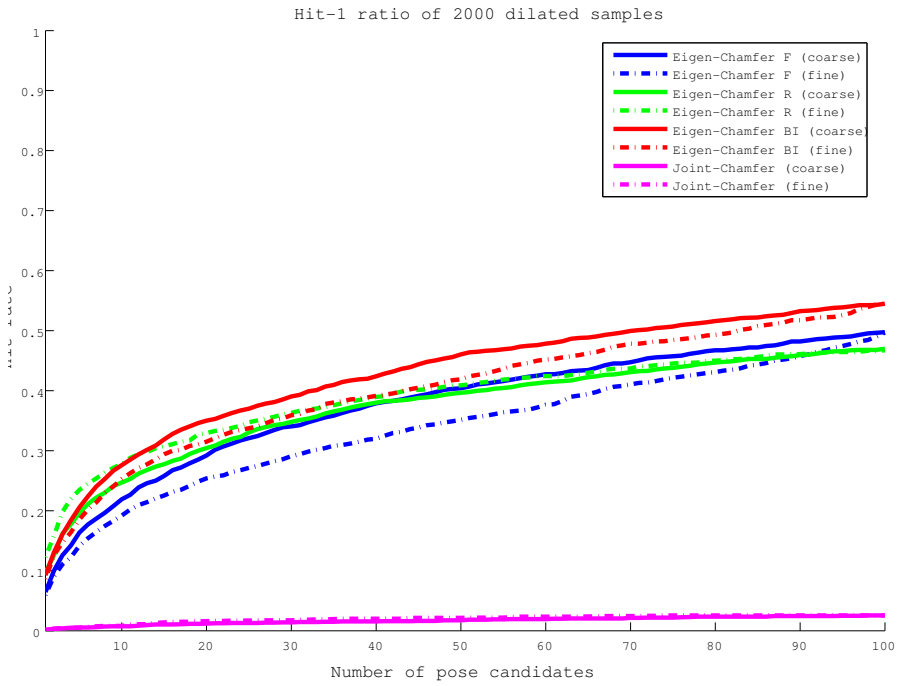


(a)

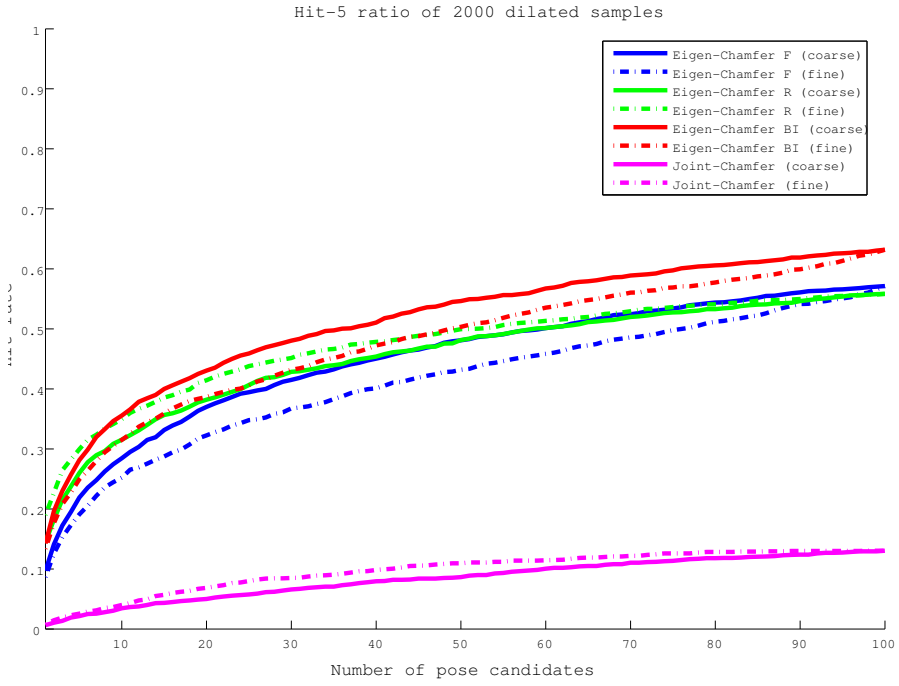


(b)

Figure 4.7: HIT-1 and HIT-5 rates for database samples.



(a)



(b)

Figure 4.8: HIT-1 and HIT-5 rates for dilated database samples.

Table 4.2: Rate of success for database images

	Eigen-Chamfer F		Eigen-Chamfer R		Eigen-Chamfer BI		Joint-Chamfer	
Rank	Coarse	Fine	Coarse	Fine	Coarse	Fine	Coarse	Fine
1	62%	76%	55%	81%	77%	**86%**	35%	45%
10	58%	67%	49%	67%	71%	**78%**	33%	41%
20	55%	63%	46%	60%	68%	**74%**	32%	38%
30	53%	60%	45%	56%	65%	**71%**	31%	37%
40	52%	57%	43%	52%	64%	**68%**	30%	35%
50	51%	55%	43%	50%	62%	**66%**	29%	34%
60	50%	54%	42%	47%	61%	**64%**	29%	33%
70	49%	52%	41%	45%	60%	**63%**	29%	31%
80	48%	50%	40%	43%	59%	**61%**	28%	30%
90	47%	49%	40%	41%	58%	**59%**	28%	29%
100	47%	47%	39%	39%	**58%**	**58%**	28%	28%

clearly the incurred approximation loss is less than that by the eigen-chamfer R which uses a single subspace to approximate distance transform. In the case of testing dataset II, the eigen-chamfer F (green) becomes worse than the eigen-chamfer R (blue), probably because in the eigen-chamfer R, the contour of dilated image biased the distance while in the eigen-chamfer R, the subspace approximation to distance transform is somehow robust for dilated image. It proves to some extent that the eigen-chamfer distance, in addition to improving efficiency, is more robust than normal chamfer distance in practice. We are interested in investigating this characteristics in the future work.

Tables 4.2 and 4.3 summarize the rate of success for testing dataset I and testing dataset II, respectively. The ROS are shown from rank-1 to rank-100 (at interval of 10). The resulting trend of ROS is similar to HIT-1 and HIT-5 rates. Although eigen-chamfer BI uses more fewer eigenvectors, the bi-directional power enabled it to perform better than either eigen-chamfer F or eigen-chamfer R. In the case of testing dataset II, the ROS of coarse evaluation are better than fine evaluation is due to the addition of silhouette cue leads to negative effect.

Table 4.3: Rate of success for dilated database images

	Eigen-Chamfer F		Eigen-Chamfer R		Eigen-Chamfer BI		Joint-Chamfer	
Rank	Coarse	Fine	Coarse	Fine	Coarse	Fine	Coarse	Fine
1	35%	33%	51%	**57%**	55%	51%	18%	20%
10	35%	34%	45%	50%	**52%**	49%	17%	20%
20	34%	33%	43%	47%	**51%**	47%	17%	19%
30	34%	32%	41%	45%	**49%**	47%	17%	19%
40	33%	32%	40%	43%	**49%**	46%	17%	18%
50	33%	32%	39%	42%	**48%**	46%	17%	18%
60	33%	32%	39%	41%	**47%**	45%	17%	18%
70	32%	31%	38%	40%	**47%**	45%	16%	17%
80	32%	31%	38%	39%	**46%**	45%	16%	17%
90	32%	31%	37%	38%	**46%**	45%	16%	17%
100	32%	32%	37%	37%	**45%**	**45%**	16%	16%

4.7.3 Experimental Results on Real Image Dataset

We next conducted experiments on real image dataset using joint-chamfer matching method [3]. Human silhouettes are first extracted from the real images with a graphic software[1]. By interactively specifying in an image some small areas of human region and of background, the software extracted the whole human region properly from the image. Many of these images were difficult (even for people) to infer the underlying 3D human poses from only silhouettes.

The silhouette images were normalized using the method described in Sec. 4.6. Normalization results were satisfactory, and so the optional multiscale normalization was not used in this experiment. Some examples of normalization are shown in Fig. 4.9, in which the first to third rows are color images, silhouettes before and after normalization, and the last row shows the edge images extracted from the normalized images.

In order to avoid the problem of inverse body orientation, we assume that body orientation is known to be outward or inward of the image plane. Figures 4.10 to 4.12 show examples of estimation results obtained by the joint-chamfer method. For each test image, 3D rendering of top 3 ranks are shown. Notice that, even though a large amount of information for describing human body poses are lost in normalized silhouettes, the results are

[3]Eigen-chamfer methods were not tested due to time limitation

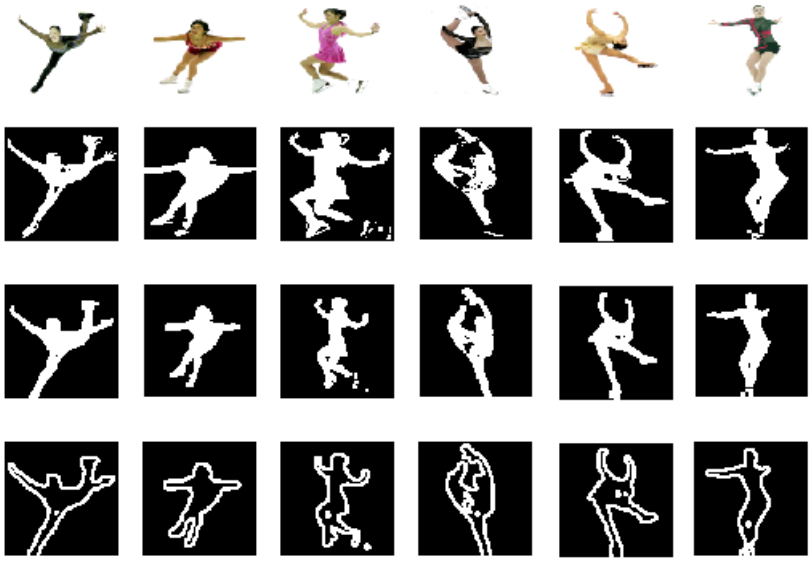
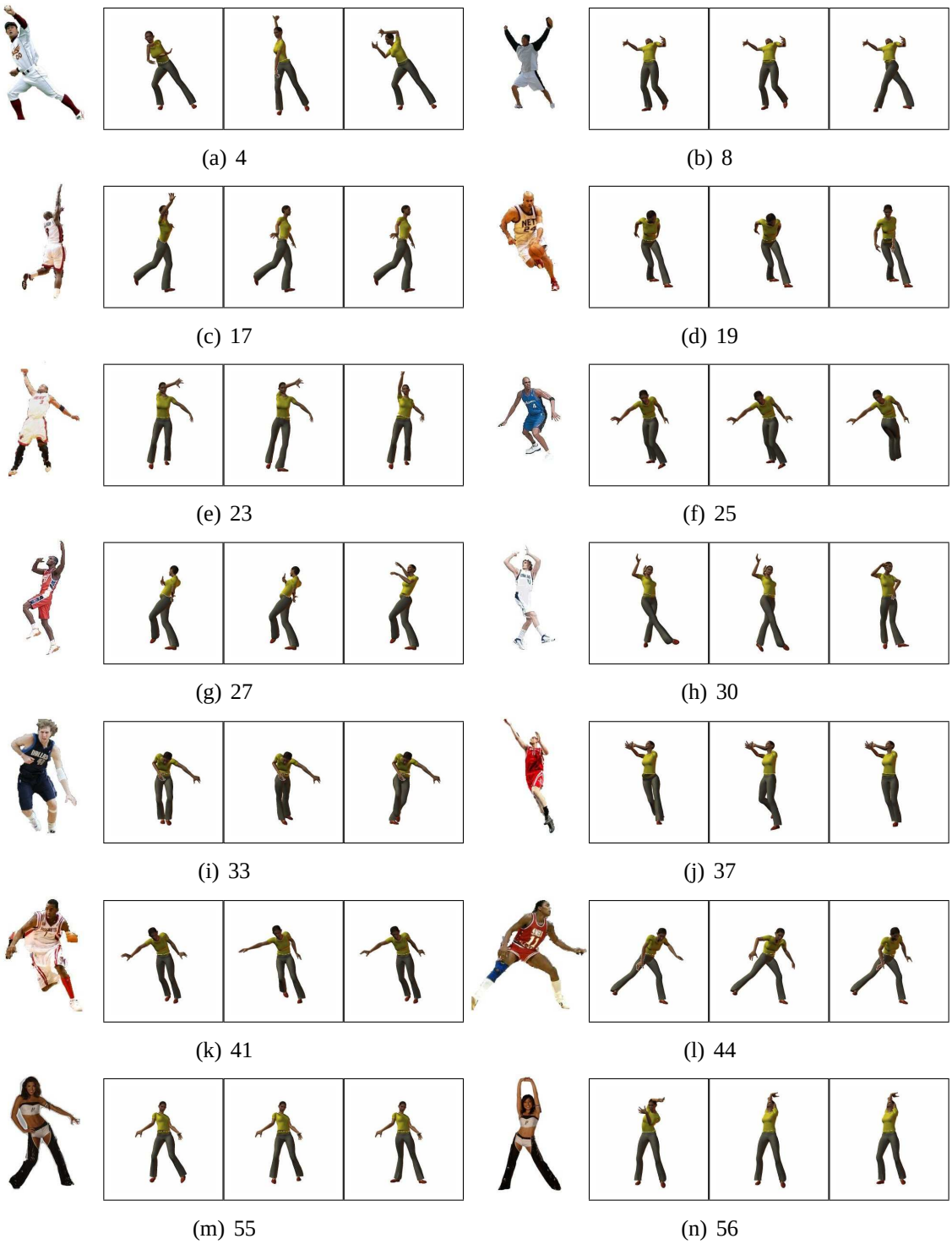


Figure 4.9: Examples of image normalization

visually close to what can be considered the right pose for input images. Good estimations are commonly achieved for lower-body poses, but biased estimates of upper-body poses sometimes result from high occlusion. When we consider more combinations than only the rank-1 combination, clear improvements are achieved. Figure 4.13 shows such examples. In the figure, the last three boxes in turn show rank-1, rank-2, and rank-3 combinations. We found that the rank-3 combination of the left column and the rank-2 combination of the right column are clearly much closer to the right pose than the rank-1 combination.

Since there are no ground truth poses for these real images, we can only subjectively evaluate the quality of resulting estimations. We built a browser-based rating system that presents the top 3 ranked combinations for each test image to participants who subjectively find the best of three combinations and rate the quality with four levels: *great* (4 points) when the estimate is entirely consistent to human perception, *good* (3 points) when one half-body estimation is good but the other is a little biased, and *average* (2 points) when one half-body estimation is good but the other is strongly biased, and *bad* (1 point) when no good half-body estimation is found. A summary of the subjective evaluations (by three persons) are shown in Fig.4.14. The average rate over all test images is 2.7 points. For some categories such as figure skating, although they are unrelated to the database's poses the evaluation results are good. However, some other categories such as kung fu and golf are not so good due to indistinct contour (caused by image downsampling or occlusion) and



(a) 4 (b) 8

(c) 17 (d) 19

(e) 23 (f) 25

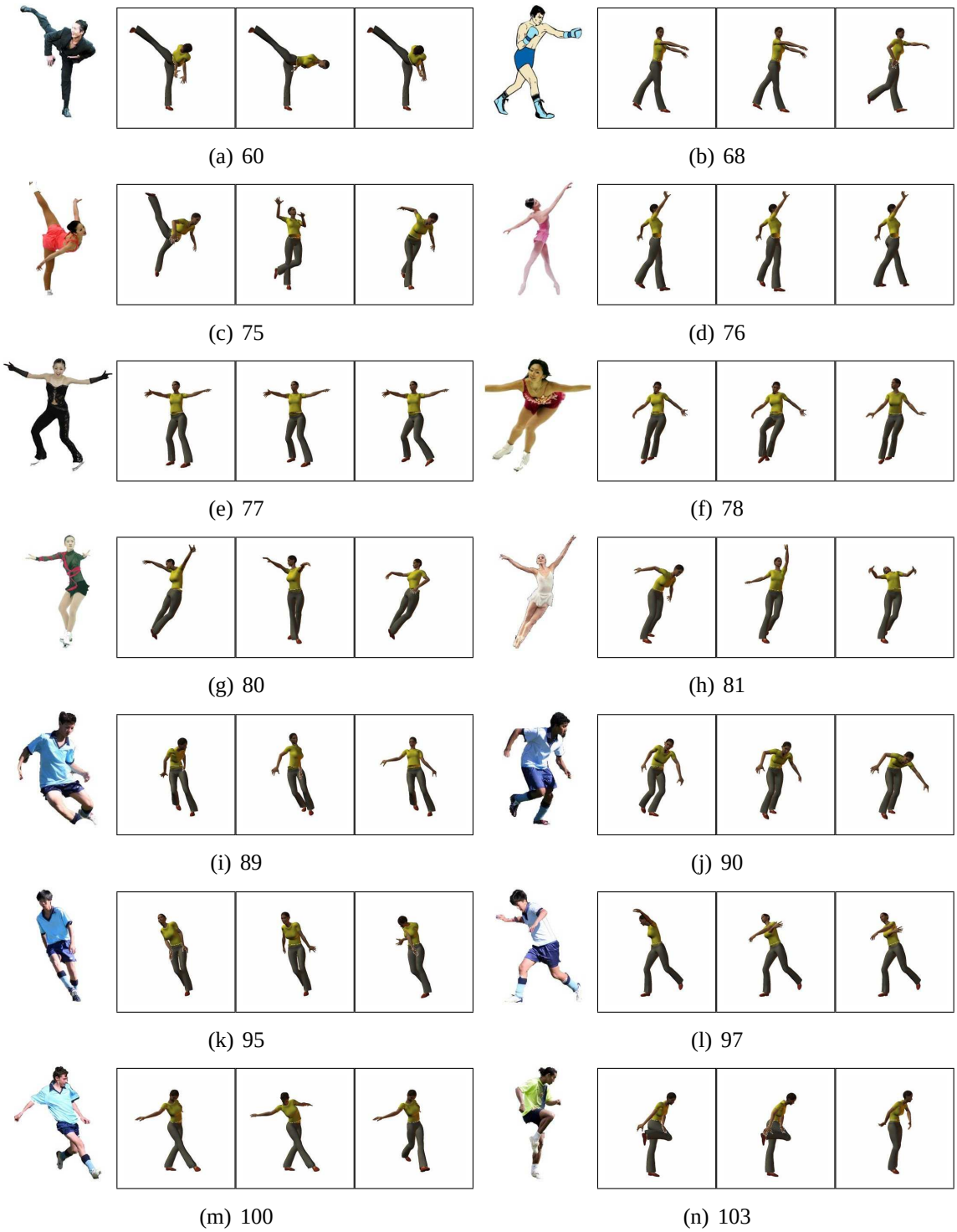
(g) 27 (h) 30

(i) 33 (j) 37

(k) 41 (l) 44

(m) 55 (n) 56

Figure 4.10: Examples of pose estimation on real images(1-14) by the joint-chamfer matching method. The first is the input image and the last three are top three candidates renderings.



(a) 60 (b) 68

(c) 75 (d) 76

(e) 77 (f) 78

(g) 80 (h) 81

(i) 89 (j) 90

(k) 95 (l) 97

(m) 100 (n) 103

Figure 4.11: Examples of pose estimation on real images (15-28) by the joint-chamfer matching method. The first is the input image and the last three are top three candidates renderings.

(a) 108 (b) 110

(c) 113 (d) 114

(e) 118 (f) 128

(g) 130 (h) 140

(i) 141 (j) 144

(k) 150 (l) 151

(m) 152 (n) 154

Figure 4.12: Examples of pose estimation on real images (29-42) by the joint-chamfer matching method. The first is the input image and the last three are top three candidates renderings.

(a) 68 (b) 80

Figure 4.13: Two examples showing better estimation in latter ranked combinations by the joint-chamfer matching method

very complex poses. Perhaps this performance is promising, considering the complexity of the task and the simple image information used. Finally, we show in Fig.4.15 some failed cases by our method. The reason of the failure is primarily due to the lack of data in the database.

4.7.4 Empirical Time and Memory Complexities

All proposed methods were mainly implemented by using MATLAB 7. Some intensive procedures were implemented in C++ for high performance. The PC for running the experiments had an Intel Pentium 4 CPU running at 3.2 GHz and 1 GB RAM.

Table 4.4 gives the comparison of computational time required by proposed methods and the normal chamfer distance[4]. The table shows that the joint-chamfer method performs the highest speed whose matching time is $0.05s$. With code optimization it can work in realtime. The eigen-chamfer methods takes much more time than joint-chamfer method at the matching step, however, it is still highly efficient if considering they perform 2 million matches in such short time. Adding silhouette cues does not increase too much time cost. The normal chamfer distance (by precomputing distance transforms) takes $5.9s$ at the matching step, about 10 times as much as by eigen-chamfer methods and 100 times by joint-chamfer method. The sorting step needs to be improved. We will replace MATLAB's internal sorting function with an efficient implementation written by C. Note that the total time cost includes the additional $0.05s$ cost by image processing.

Table 4.5 gives the comparison of memory usage required by the proposed methods and the normal chamfer distance. The advantage of approximate chamfer match methods is remarkable with respect to memory efficiency. The normal chamfer distance requires

[4]We estimate the possible time and memory required to the 2 million database from the empirical time and memory usages on a small database of $14,964$ full-body poses.

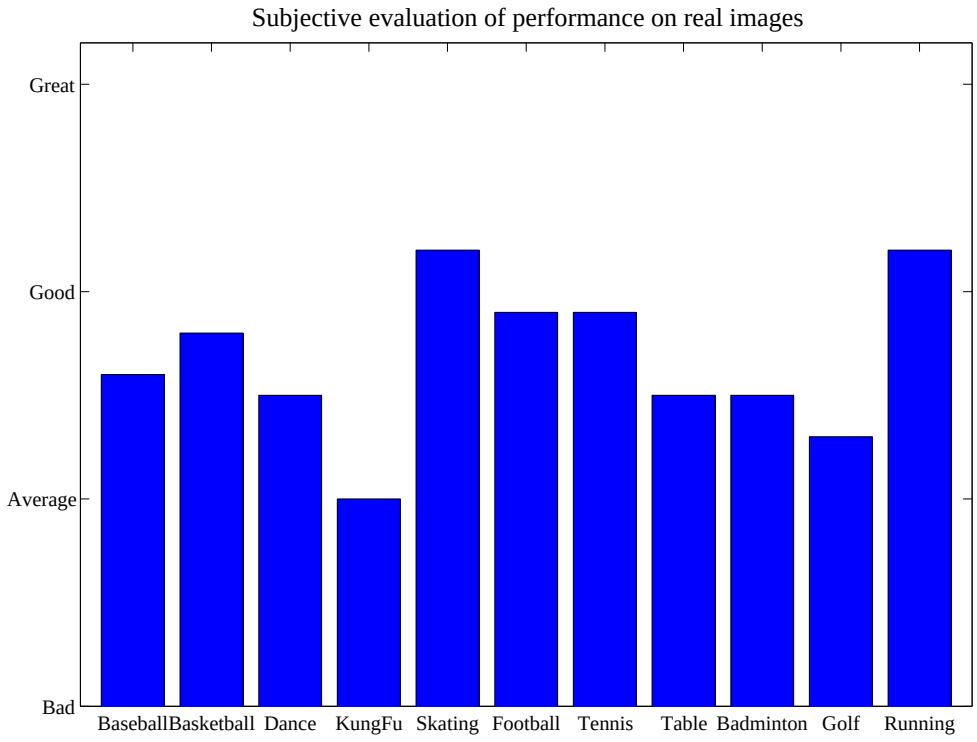
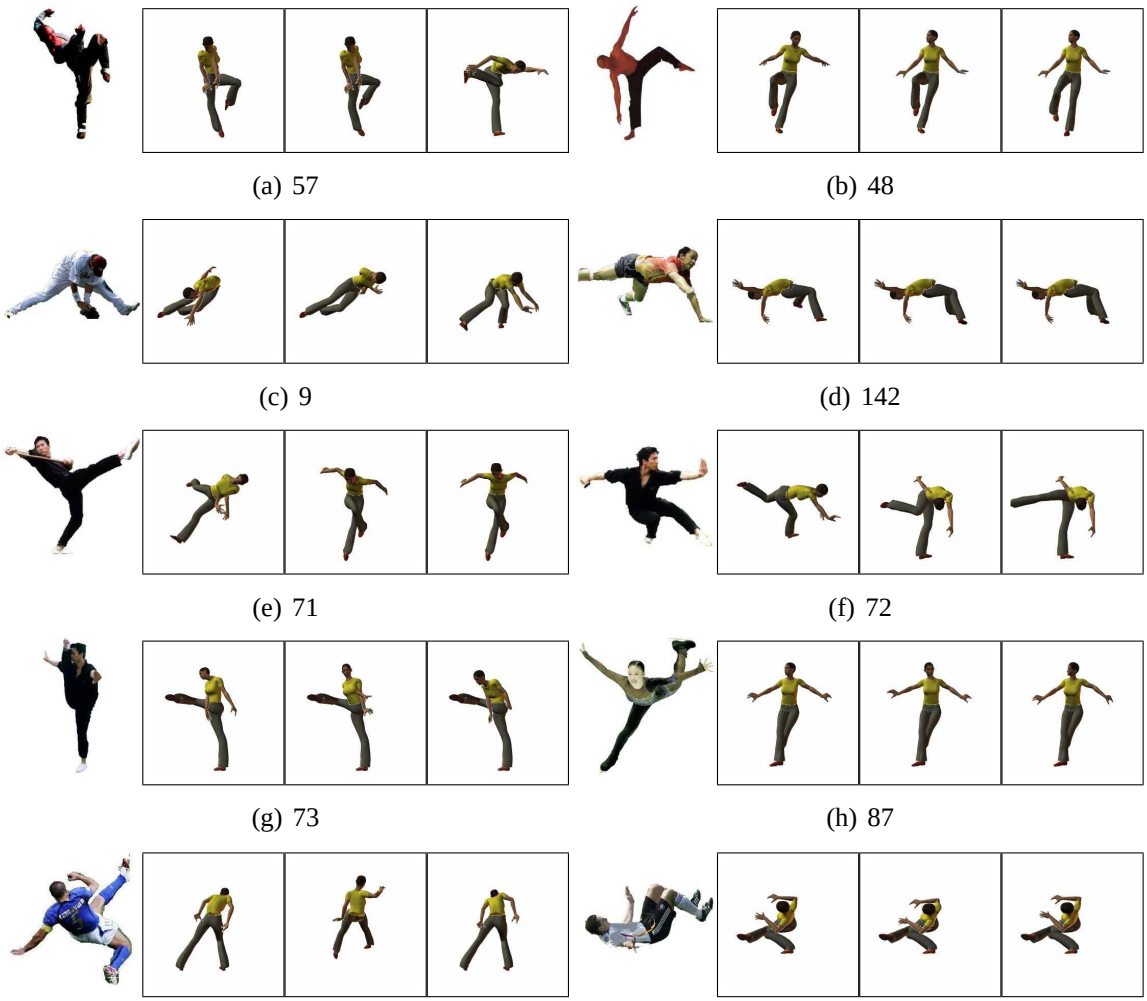


Figure 4.14: Subjective evaluation of performance on real images by the joint-chamfer matching method. From left to right: Baseball (14), Basketball (31), Dance (11), Kung Fu (18), Figure Skating (13), Football (32), Tennis (16), Table Tennis (4), Badminton (6), Golf (4), and Running (6). The number in parenthesis indicates data size.

20GB memory space which is infeasible for most of applications. While our joint-chamfer and eigen-chamfer methods requires a total of about 549MB and 540MB, respectively.

To further clarify the memory usage in our code implementation, the normal chamfer distance used uint8-precision (1 byte) to store distance transform which cannot be optimized any more; while the approximate chamfer distances mostly used uint16-precision (2 bytes) to store necessary data.



(a) 57 (b) 48

(c) 9 (d) 142

(e) 71 (f) 72

(g) 73 (h) 87

(i) 106 (j) 111

Figure 4.15: Failure examples of pose estimation on real images by the joint-chamfer matching method.

4.8 Discussions

4.8.1 Rotation Problem

Since the image normalization routine can only handle the variations of translation and scaling, at present our method is incapable of dealing with the case of tilted human figure. In our ongoing research, we are extending our method to handle the rotation problem via two solutions: (i) add rotated versions of training images into database; (ii) examine all rotated versions of the query image.

Table 4.4: Comparison of empirical computational time

Method	Chamfer Matching	Sorting	+ Silhouette Cue	Total
Normal Chamfer	5.90s	0.92s	0.08s	6.97s
Joint-Chamfer	0.05s	0.14s	0.08s	0.34s
Eigen-Chamfer F	0.65s	0.92s	0.08s	1.72s
Eigen-Chamfer R	0.62s	0.92s	0.08s	1.69s
Eigen-Chamfer BI	0.81s	0.92s	0.08s	1.88s

Table 4.5: Comparison of empirical memory usage

Method	DTs	Contours	Silhouettes
Normal Chamfer	18.6G	1.3G	39M
Joint-Chamfer	500M	10M	39M
	Eigenspaces	**Coefficents**	**Silhouettes**
Eigen-Chamfer F	40M	345M	39M
Eigen-Chamfer R	8M	388M	39M
Eigen-Chamfer BI	242M	262M	39M

4.8.2 Half-body vs. Parts

The proposed half-body method can be naturally extended to a parts method that uses detailed parts, *i.e.*,, two arms, two legs, and torso. In this section, We try to theoretically analyze the advantage and disadvantage of two methods from the following aspects: computational efficiency, capacity of combination, realistic pose assurance and self-occlusion.

- *Computational efficiency*: Both are efficient.
- *Capacity of combination*: The parts method is clearly more powerful than the half-body method.
- *Realistic pose assurance*: The half-body method is better than the parts method. The reason is that half-body method defines one constraint between half-bodies and half-bodies themselves are realistic subposes; whereas the parts method must provide proper constraints to ensure all parts to be combined into a globally realistic pose.
- *Ability of coping with self-occlusion problem*: The half-body method is better than the parts method. As shown in the top row of Fig.4.13, although the right arm is com-

pletely unseen, the half-body method can also retrieve proper half-body candidates, while the parts method is unlikely to choose correct candidates for right arm.

As a summary, two methods are computationally efficient. The parts method can produce much more poses by combination, but suffers heavily from unrealistic pose and self-occlusion problems. The half-body method can reasonably cope with the two problems, giving satisfactory results as shown in the experimental results.

4.9 Conclusion

In this chapter, two matching methods: eigen-chamfer and joint-chamfer are proposed to obtain for a given input image the candidate poses from a large database. By introducing the eigenspace approximation to distance transform, the eigen-chamfer method shifts the chamfer distance computation from the image space to a lower-dimensional subspace. The joint-chamfer method, alternatively, first efficiently retrieves half-body candidate poses by partial contour matching. Then valid combinations are selected subject to combination constraints, which are evaluated efficiently by a small number of simple arithmetic operations. Two matching method have advantages in both time and memory efficiencies.

Adding much more examples into the database can further improve the accuracy of pose estimation. The eigen-chamfer method can control the time and memory cost within an acceptable range through learning much more subspaces following the growth of database. The joint-chamfer method suffers little from the database's growth due to the increased half-body examples are relatively fewer.

A similar idea [28] to the eigen-chamfer method has been already proposed for object recognition problem. We plan to make a theoretical and empirical comparison between two methods.

Chapter 5

Re-Ranking Candidate Poses via Kernel Subspace

In the previous chapter, two efficient chamfer matching algorithms are introduced to retrieve for a query image a set of candidate poses. In this chapter, the goal is to re-rank these candidate pose such that candidates which are exact or close to the real pose can be assigned higher ranks. Two kernel subspace ranking methods are proposed. Kernel Principal Component Analysis (KPCA) [59] and Kernel Canonical Correlation Analysis (KCCA) [35, 4] are employed to learn nonlinear subspaces characterizing the underlying structure of image-pose pairs. Candidate poses are ranked by some special criteria based on the subspace projection. The new kernel subspace ranking is combined with the original (image similarity) ranking to yield better ranks.

5.1 Introduction

So far the candidate poses are ranked by the image similarity (measured by chamfer distance) to the query. However, the image similarity is not optimal and probably inconsistent to the desired pose similarity. In other word, sometimes when query is different to database images due to body size, clothes, *etc*, irrelevant poses may be overestimated while relevant poses which are close to the real pose may be underestimated. Thus, it is necessary to re-rank the candidate poses in combination with other complemental knowledge. This chapter presents two complemental ranking algorithms based on kernel subspace projection.

The image and 3D pose can be viewed as two representations of the human pose. If the underlying relationship between image and pose can be properly represented in a mathe-

matical form, this can help us to judge the fitness of a candidate pose for the query image. Two kernel subspace methods are considered for modeling this relationship.

The first method uses Kernel Principal Component Analysis (KPCA) to learn a joint subspace for image and pose. Image and pose are viewed as two parts of a (virtual) joint data. KPCA is applied to the training dataset of joint data and the extracted subspace retains major variances of the training dataset. If a pair of image and pose is a real pair, it should reside in the learned subspace and the subspace approximation should be close to the original joint data.

The second method uses Kernel Canonical Correlation Analysis (KCCA) to learn subspaces for image and pose separately. KCCA is applied to image and pose datasets separately such that two subspace projections are maximally correlated. If a pair of image and pose is a real pair, two subspace projections should have a high correlation score.

The ranks derived with the kernel subspace (*i.e.*,KPCA or KCCA) are combined with the original image similarity based ranks. Although there are many ways to combine ranks, here these two ranks are simply combined linearly,

$$\gamma * \text{Subspace-Rank}(c) + (1-\gamma) * \text{Similarity-Rank}(c), \quad \forall\, c \in \mathbb{C}, \tag{5.1}$$

where γ is a parameter to tune the weight of two ranks and $\mathbb{C}$ denotes the set of candidate poses. It is expected that two rankings will be complementary each other so that the combination of them could perform better than either does alone.

The following sections in turn describe the two ranking methods: KPCA-Rank and KCCA-Rank. The principle of KPCA and its ranking criteria: subspace projection loss are introduced at first. Afterwards, the principle of KCCA and its ranking criteria: correlation score are described. Finally, the experiment results and performance comparison are presented.

5.2 KPCA Ranking

Kernel PCA has been proved to have a better performance than many other nonlinear techniques in extracting interesting nonlinear structures of the data [59, 58]. Kernel PCA has applications such as signal denoising [34, 40, 32], complex output regression[72, 73], shape recognition [74, 57]. More recently, there are some works applying kernel PCA to human pose estimation [67] and motion denoising [66].

5.2.1 Kernel Principal Component Analysis

Given a set of data vectors $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \in \mathbb{R}^n$. Kernel PCA performs linear PCA in a higher dimensional feature space $\phi : x \rightarrow \phi(x)$. Although the vector $\mathbf{\Phi}(\mathbf{x})$ in the feature space is generally not known explicitly, it is possible to compute inner-product without explicitly mapping into the high dimensional feature space. The kernel inner-products between $\mathbf{x}_i$ and $\mathbf{x}_j$ is

$$k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{\Phi}(\mathbf{x}_i)'\mathbf{\Phi}(\mathbf{x}_j), \tag{5.2}$$

which allow us to compute the value of the inner-product in the feature space without having to explicitly compute the map $\mathbf{\Phi}$.

Denote the kernel matrix $\mathbf{K}$ with its (i, j) element $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, and the centering matrix

$$\mathbf{H} = \mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}', \tag{5.3}$$

where $\mathbf{I}$ is the $N \times N$ identity matrix, $\mathbf{1} = [1, \ldots, 1]'$ is an $N \times 1$ vector. Analogously in the linear PCA, kernel PCA involves an eigen decomposition to centered kernel matrix $\mathbf{HKH}$,

$$\mathbf{HKH} = \mathbf{E}\mathbf{\Lambda}\mathbf{E}', \tag{5.4}$$

where $\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_N]$ with $\mathbf{e}_i = [e_{i1}, \ldots, e_{iN}]'$ is the matrix containing the eigenvectors and $\mathbf{\Lambda} = diag(\lambda_1, \ldots, \lambda_N)$ contains the corresponding eigenvalues. The eigenvector $\mathbf{e}_i$ is normalized into $\frac{\mathbf{e}_i}{\sqrt{\lambda_i}}$.

Denote the mean of the ϕ-mapped data by $\bar{\mathbf{\Phi}} = \frac{1}{N}\Sigma_{i=1}^{N}\mathbf{\Phi}(\mathbf{x}_i)$ and define the centered map as: $\tilde{\mathbf{\Phi}} = \mathbf{\Phi} - \bar{\mathbf{\Phi}}$. The k^{th} eigenvector of the covariance matrix in the feature space can then be shown to be

$$\mathbf{v}_k = \Sigma_{i=1}^{N} e_{ki}\tilde{\mathbf{\Phi}}(\mathbf{x}_i). \tag{5.5}$$

To extract nonlinear principal components for the ϕ-mapping of a test point $\mathbf{x}$, we compute the projection onto the k^{th} eigenvector by

$$\begin{aligned} f_k &= \tilde{\mathbf{\Phi}}(\mathbf{x})'\mathbf{v}_k = \Sigma_{i=1}^{N} e_{ki}\tilde{\mathbf{\Phi}}(\mathbf{x})'\tilde{\mathbf{\Phi}}(\mathbf{x}_i) \\ &= \Sigma_{i=1}^{N} e_{ki}\tilde{k}(\mathbf{x}, \mathbf{x}_i). \end{aligned} \tag{5.6}$$

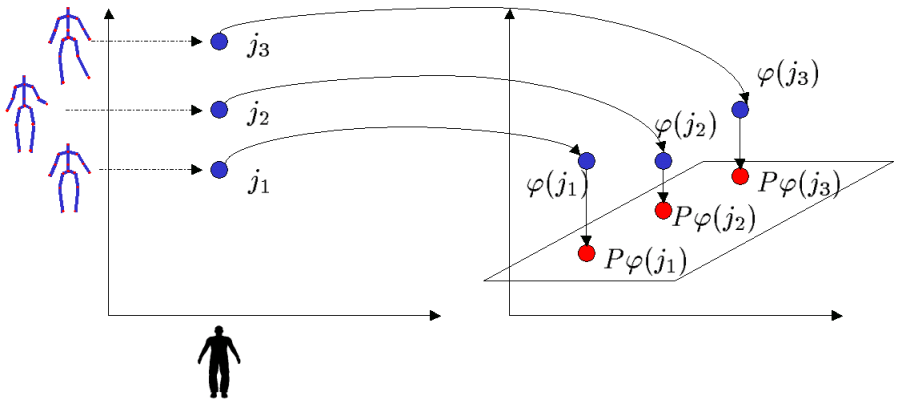


Figure 5.1: Illustration of ranking by projection loss

where

$$\begin{aligned}\tilde{k}(\mathbf{x},\mathbf{y}) &= (\mathbf{\Phi}(\mathbf{x})-\bar{\mathbf{\Phi}})'(\mathbf{\Phi}(\mathbf{y})-\bar{\mathbf{\Phi}}) \\ &= k(\mathbf{x},\mathbf{y}) - \frac{1}{N}\Sigma_{i=1}^{N}k(\mathbf{x},\mathbf{x}_i) - \frac{1}{N}\Sigma_{i=1}^{N}k(\mathbf{x}_i,\mathbf{y}) + \frac{1}{N^2}\Sigma_{i,j=1}^{N}k(\mathbf{x}_i,\mathbf{x}_j) \\ &= k(\mathbf{x},\mathbf{y}) - \frac{1}{N}\mathbf{1}'\mathbf{k}_{\mathbf{x}} - \frac{1}{N}\mathbf{1}'\mathbf{k}_{\mathbf{y}} + \frac{1}{N^2}\mathbf{1}'\mathbf{K}\mathbf{1}, \end{aligned} \tag{5.7}$$

is the centered kernel function. To reconstruct the ϕ-mapping of a vector $\mathbf{x}$ from its subspace projections f_k, we define the projection of $\mathbf{\Phi}(\mathbf{x})$ onto the subspace spanned by the first K eigenvectors, $\mathbf{P}_{\phi(\mathbf{x})}$, as

$$\begin{aligned}\mathbf{P}_{\mathbf{\Phi}(\mathbf{x})} &= \Sigma_{k=1}^{K} f_k \mathbf{v}_k + \bar{\mathbf{\Phi}} \\ &= \mathbf{V}\mathbf{f} + \bar{\mathbf{\Phi}}, \end{aligned} \tag{5.8}$$

where $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_K]$ and $\mathbf{f} = [f_1, \ldots, f_K]'$.

5.2.2 Ranking by Projection Loss

KPCA ranking method ranks candidate poses based on the projection loss between the joint data (in the feature space) and the subspace approximation (in the feature space). Denote the joint data vector be $\mathbf{z} = [\mathbf{x}, \mathbf{y}]'$, where $\mathbf{x}$ and $\mathbf{y}$ are the image and pose vectors, respectively.

The kernel function for the joint data $\mathbf{z}$ is defined as a product of two component kernels

$$k(\mathbf{z}_m, \mathbf{z}_n) \triangleq k(\mathbf{x}_m, \mathbf{x}_n) * k(\mathbf{y}_m, \mathbf{y}_n). \tag{5.9}$$

A notable advantage of the kernel approach is the ability to handle various data types, e.g. strings and images, by using an appropriate kernel function. Here, the component kernels are Gaussian radial basis function (rbf) of the form

$$k(\mathbf{x}_m, \mathbf{x}_n) \triangleq \exp\left(-\frac{1}{2\sigma_x^2}(d_{cham}^{\mathbf{x}_m,\mathbf{x}_n})^2\right) \tag{5.10}$$

$$k(\mathbf{y}_m, \mathbf{y}_n) \triangleq \exp\left(-\frac{1}{2\sigma_y^2} \parallel \mathbf{y}_m - \mathbf{y}_n \parallel^2\right). \tag{5.11}$$

The image kernel $k(\mathbf{x}_m, \mathbf{x}_n)$ uses the chamfer distance $d_{cham}^{(\mathbf{x}_m,\mathbf{x}_n)}$ to measure the distance between images.

For a given image $\mathbf{x}$ and the associated real pose $\mathbf{y}$, if the joint data $\mathbf{z} = [\mathbf{x}, \mathbf{y}]'$ is well characterized by the kernel subspace, the subspace approximation after projecting onto the kernel subspace, $\mathbf{P}_{\phi(\mathbf{z})}$, should satisfy $\mathbf{P}_{\phi(\mathbf{z})} \cong \phi(\mathbf{z})$. With this assumption, we thus rank candidate poses by the projection loss. The projection loss is defined as the squared distance between $\phi(\mathbf{z})$ and $\mathbf{P}_{\phi(\mathbf{z})}$, whose expansion is given as:

$$\begin{aligned}
d^2(\mathbf{P}_{\phi(\mathbf{z})}, \phi(\mathbf{z})) &= \parallel \mathbf{P}_{\phi(\mathbf{z})} - \phi(\mathbf{z}) \parallel^2 \\
&= \parallel (\mathbf{P}_{\phi(\mathbf{z})} - \bar{\phi}) - (\phi(\mathbf{z}) - \bar{\phi}) \parallel^2 \\
&= \parallel \mathbf{V}\mathbf{f} \parallel^2 -2(\mathbf{V}\mathbf{f})'\tilde{\phi}(\mathbf{z}) + \tilde{\phi}(\mathbf{z})'\tilde{\phi}(\mathbf{z}) \\
&= \mathbf{f}'\mathbf{f} - 2\mathbf{f}'(\mathbf{V}'\tilde{\phi}(\mathbf{z})) + \tilde{k}(\mathbf{z}, \mathbf{z}) \\
&= -\mathbf{f}'\mathbf{f} + k(\mathbf{z}, \mathbf{z}) - \frac{2}{N}\mathbf{1}'\mathbf{k}_\mathbf{z} + \frac{1}{N^2}\mathbf{1}'\mathbf{K}\mathbf{1} \\
&= -\mathbf{f}'\mathbf{f} - \frac{2}{N}\mathbf{1}'\mathbf{k}_\mathbf{z} + \Omega.
\end{aligned} \tag{5.12}$$

In (5.12) $\Omega = k(\mathbf{z}, \mathbf{z}) + \frac{1}{N^2}\mathbf{1}'\mathbf{K}\mathbf{1}$ and it is constant. Thus minimizing $d^2(\mathbf{P}_{\phi(\mathbf{z})}, \phi(\mathbf{z}))$ is equivalent to maximizing

$$\mathbf{f}'\mathbf{f} + \frac{2}{N}\mathbf{1}'\mathbf{k}_\mathbf{z}. \tag{5.13}$$

The candidate poses with higher scores of (5.13) are assigned to higher rank.

5.3 KCCA Ranking

Canonical Correlation Analysis (CCA) [26] is a technique for finding pairs of eigenvectors that maximize the correlation between the projections of paired variables onto their corresponding eigenvectors. In an attempt to increase the flexibility of the feature selection, kernelisation of CCA (KCCA) has been applied to map the hypotheses to a higher-dimensional

feature space [35, 4]. KCCA has been used applied to cross-language information retrieval [70] and retrieval of images from a text query without any reference to labeling associated with the image [24].

5.3.1 Kernel Canonical Correlation Analysis

Let $\mathbf{X}$ and $\mathbf{Y}$ denote sample measurements on m objects, with columns $\mathbf{x}_i \in \mathbb{R}^{n_x}$ and $\mathbf{y}_i \in \mathbb{R}^{n_y}$ describing different aspects of these objects. The aim of canonical correlation analysis is to find pairs of eigenvectors $\mathbf{v}_j$ and $\mathbf{w}_j$ that maximize the correlation between the canonical variates, $\mathbf{a}_j = \mathbf{X}'\mathbf{v}_j$ and $\mathbf{b}_j = \mathbf{Y}'\mathbf{w}_j$.

$$cor(\mathbf{a}_j, \mathbf{b}_j) = \frac{\langle \mathbf{a}_j, \mathbf{b}_j \rangle}{\| \mathbf{a}_j \| \| \mathbf{b}_j \|}. \tag{5.14}$$

Usually, this is formulated as a constraint optimization problem

$$\begin{aligned} &\underset{\mathbf{v}_j, \mathbf{w}_j}{\operatorname{argmax}} \, \mathbf{v}_j' \mathbf{X}\mathbf{Y}'\mathbf{w}_j \\ &\text{subject to } \; \mathbf{v}_j \mathbf{X}\mathbf{X}'\mathbf{v}_j = \mathbf{w}_j \mathbf{Y}\mathbf{Y}'\mathbf{w}_j = 1. \end{aligned} \tag{5.15}$$

Up to $r = \min(m, n_x, n_y)$ pairs of canonical vectors can be recursively obtained, which maximize (5.15) subject to corresponding variates being orthogonal to previously found pairs.

Because of its linearity, CCA may not extract useful descriptors of the data. This makes CCA improper when for example the correlation exist in some nonlinear relationship. Following the same idea of Kernel PCA, the kernelizing of CCA offers an alternative solution by first projecting the data into a higher dimensional feature space $\phi : x \to \phi(x)$ before performing CCA in the new feature space. Let $\mathbf{\Phi}_x$ be the matrix whose columns are the vectors $\mathbf{\Phi}_x(\mathbf{x}_i), i = 1, \ldots, N$, and similarly $\mathbf{\Phi}_y$ be a matrix with columns $\mathbf{\Phi}_y(\mathbf{y}_i), i = 1, \ldots, N$.

It is known that the canonical vectors can be represented as linear combinations $\mathbf{v}_j = \mathbf{\Phi}_x \alpha_j$ and $\mathbf{w}_j = \mathbf{\Phi}_y \beta_j$ using $\alpha_j, \beta_j \in \mathbb{R}^m$ as expansion coefficients. Substituting into (5.15) obtains the following

$$\begin{aligned} &\underset{\alpha_j, \beta_j}{\operatorname{argmax}} \, \alpha_j' \mathbf{\Phi}_x' \mathbf{\Phi}_x \mathbf{\Phi}_y' \mathbf{\Phi}_y \beta_j \\ &\text{subject to } \; \alpha_j' \mathbf{\Phi}_x' \mathbf{\Phi}_x \mathbf{\Phi}_x' \mathbf{\Phi}_x \alpha_j = \beta_j' \mathbf{\Phi}_y' \mathbf{\Phi}_y \mathbf{\Phi}_y' \mathbf{\Phi}_y \beta_j = 1. \end{aligned} \tag{5.16}$$

This is the dual form of the primal CCA optimization problem.

Let $\mathbf{K}_x = \mathbf{\Phi}'_x\mathbf{\Phi}_x$ and $\mathbf{K}_y = \mathbf{\Phi}'_y\mathbf{\Phi}_y$ denote the $m \times m$ kernel inner product matrices which can be constructed element-wise as $(\mathbf{K}_x)_{ij} = k_x(\mathbf{x}_i, \mathbf{x}_j)$ and $(\mathbf{K}_y)_{ij} = k_x(\mathbf{y}_i, \mathbf{y}_j)$ for $i, j = 1, \ldots, m$. Substituting into the dual form of CCA equation (5.16) gives

$$\begin{aligned} &\underset{\alpha_j, \beta_j}{\operatorname{argmax}}\, \alpha'_j \mathbf{K}_x \mathbf{K}_y \beta_j \\ &\text{subject to } \alpha'_j \mathbf{K}_x^2 \alpha_j = \beta'_j \mathbf{K}_y^2 \beta_j = 1. \end{aligned} \tag{5.17}$$

By using corresponding Lagrangian and Kuhn-Tucker conditions the above optimization problem can be rewritten as the eigenvalue problems

$$(\mathbf{K}_x^2)^{-1}\mathbf{K}_x\mathbf{K}_y(\mathbf{K}_y^2)^{-1}\mathbf{K}_y\mathbf{K}_x\alpha_j = \lambda_j^2\alpha_j \tag{5.18}$$

$$(\mathbf{K}_y^2)^{-1}\mathbf{K}_y\mathbf{K}_x(\mathbf{K}_x^2)^{-1}\mathbf{K}_x\mathbf{K}_y\beta_j = \lambda_j^2\beta_j. \tag{5.19}$$

The canonical vectors $\mathbf{v}_j = \mathbf{\Phi}_x\alpha_j$ and $\mathbf{w}_j = \mathbf{\Phi}_y\beta_j$ are obtained as vectors corresponding to the r positive eigenvalues $1 \geq \lambda_1^2, \ldots, \geq \lambda_r^2 > 0$. Note that the eigenvalues equal the squared canonical correlation coefficients such that $\lambda_j = cor(\mathbf{a}_j, \mathbf{b}_j)$.

5.3.2 Ranking by Correlation Score

The image and the associated pose are viewed as the paired variables. KCCA ranking method aims to rank the candidate poses in terms of the correlation scores between the projections of query image and pose (in the feature space) onto the subspace. The canonical variates $\mathbf{a}_q = [a_{q1}, \ldots, a_{qr}]'$ for query image $\mathbf{x}_q$ and $\mathbf{b}_c = [b_{c1}, \ldots, b_{cr}]'$ for all candidate poses $\mathbf{y}_c$ can easily be calculated by computing the score on the r kernel canonical eigenvectors,

$$a_{qj} = \Phi_x(\mathbf{x}_q)'\mathbf{v}_j = (\Phi_x(\mathbf{x}_q)'\mathbf{\Phi}_x)\alpha_j = \mathbf{k}_{xq}\alpha_j \tag{5.20}$$

$$b_{cj} = \Phi_y(\mathbf{y}_c)'\mathbf{w}_j = (\Phi_y(\mathbf{y}_c)'\mathbf{\Phi}_y)\beta_j = \mathbf{k}_{yc}\beta_j. \tag{5.21}$$

The candidate poses are ranked by the correlation scores $\mathbf{a}_q{}'\mathbf{b}_c$. The higher the score is, the more similar the candidate pose and the query image are.

5.4 Experiments

In this section, the performances of two re-ranking methods are evaluated. Training data is composed of 3000 samples uniformly selected out of the source dataset. For both KPCA-

Table 5.1: Experimental results on training dataset

Rank Order	1^{st}	$\leq 3^{rd}$	$\leq 5^{th}$	$\leq 10^{th}$
Hit ratio (KPCA)	79%	92%	93%	99%
Hit ratio (KCCA)	98%	99%	100%	100%

and KCCA-ranking methods, 500 eigenvectors were retained, and the standard deviation σ_x, σ_y in the kernel function (5.10) were set by using the average distances calculated from 95% successive pairs of examples in the source dataset [1].

The performances of the KPCA- and KCCA-ranking methods were first evaluated on the training data. For every image in the training set, it was paired with all available poses in the training data. The resulting 3000 pairs were ranked by KPCA- and KCCA-ranking methods, respectively. The HIT-1 rates are shown in Table 5.1. For KPCA-ranking method, 79% real pairs (*i.e.*,pairs of image and associated pose) appeared in the 1^{st} rank; 92%, 93% and 99% real pairs appeared in $\leq 3^{rd}$, $\leq 5^{th}$ and $\leq 10^{th}$ ranks, respectively. For KCCA-ranking method, there were in turn 98%, 99% and 100% real pairs appearing in the 1^{st}, 3^{rd} and 5^{th} ranks, respectively. It shows that, under the same condition, the KCCA-ranking method has a clearly better performance with respect to training data than KPCA-ranking method.

Next, the re-ranking experiments on the test dataset used in the previous chapter were conducted. The test dataset consists of 2000 samples (randomly selected from database) and their dilated versions. We re-ranked the candidate poses obtained by using bi-directional eigen-chamfer chamfer distance (Eigen-Chamfer BI) — which gave the best results among others. The 100 candidate poses were first ranked using KPCA- and KCCA-ranking methods, respectively. Afterwards, the obtained kernel subspace ranks (KR) were linearly combined with the image similarity ranks (CR), under different settings of weight parameter, $\gamma = 0, 0.1, \ldots, 1$. The statics of HIT-1, HIT-5 and ROS (rate of success) were calculated respectively for those refined rankings (see Sec. 4.7.1 for the definitions of HIT-1, HIT-5 and ROS).

Table 5.2 summarizes the ROS for database samples. The ROS of the 1^{st} to 100^{th} rank (at interval of 10) are listed. The highlighted numbers in each row indicate the highest values until that rank. The CR had the best ROS in the 1^{st} rank, however, from the 10^{th} rank, re-rankings, including from .9CR+.1KR to .7CR+.3KR, outperformed (or equal) the CR.

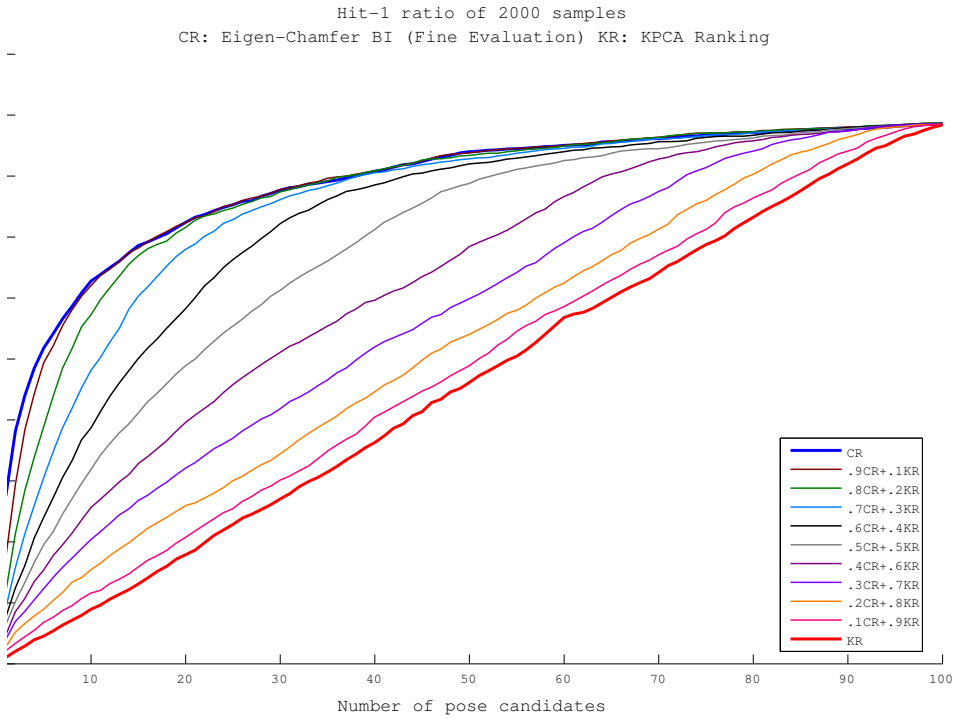
[1] Because the source dataset is a motion capture sequence, successive examples are usually close to each other. The average pose distance has been previously used in the chapter of database construction.

Table 5.2: ROS comparison of re-ranking methods using different weights (2000 samples)

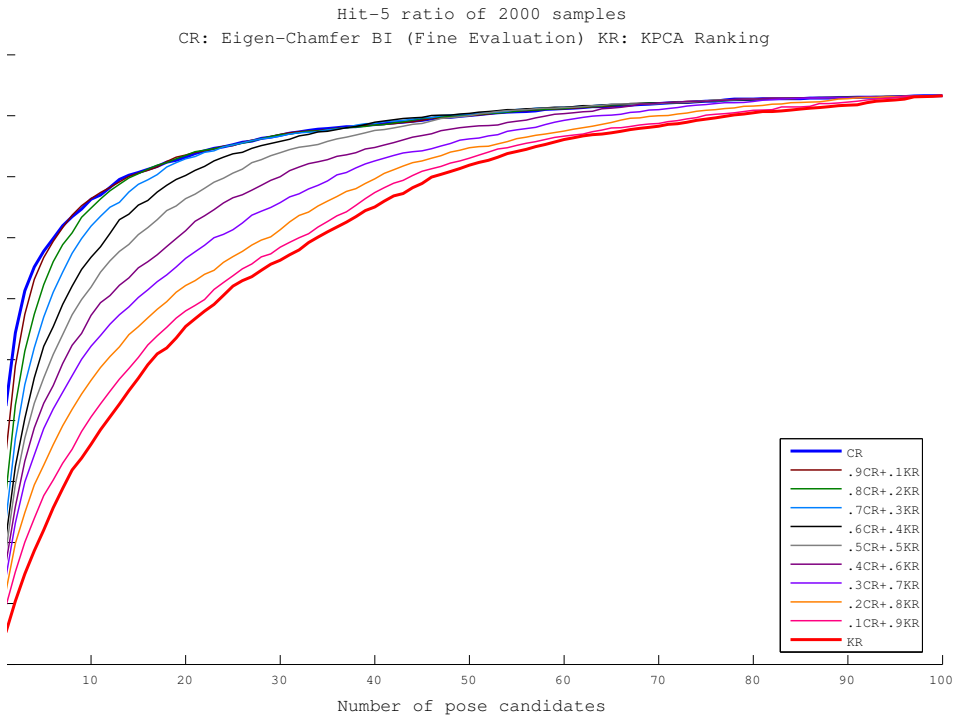
CR: Eigen-Chamfer BI (*Fine Evaluation*), KR: KPCA-Rank											
	CR	.9CR + .1KR	.8CR + .2KR	.7CR + .3KR	.6CR + .4KR	.5CR + .5KR	.4CR + .6KR	.3CR + .7KR	.2CR + .8KR	.1CR + .9KR	KR
1	**86%**	84%	81%	79%	78%	76%	75%	73%	72%	69%	65%
10	**78%**	**78%**	77%	75%	73%	71%	69%	67%	65%	63%	62%
20	**74%**	**74%**	**74%**	**74%**	72%	70%	68%	66%	64%	62%	62%
30	**71%**	**71%**	**71%**	**71%**	**71%**	69%	67%	65%	64%	63%	62%
40	68%	**69%**	**69%**	**69%**	**69%**	68%	66%	64%	63%	63%	62%
50	66%	**67%**	**67%**	**67%**	**67%**	**67%**	65%	64%	63%	63%	62%
60	64%	**65%**	**65%**	**65%**	**65%**	**65%**	64%	64%	63%	62%	62%
70	63%	63%	63%	63%	**64%**	**64%**	63%	63%	62%	62%	62%
80	61%	61%	61%	**62%**	**62%**	**62%**	**62%**	**62%**	61%	61%	61%
90	59%	**60%**	**60%**	**60%**	**60%**	**60%**	**60%**	**60%**	**60%**	**60%**	**60%**
100	58%	58%	58%	58%	58%	58%	58%	58%	58%	58%	58%
CR: Eigen-Chamfer BI (*Fine Evaluation*), KR: KCCA-Rank											
1	**86%**	85%	82%	81%	79%	78%	76%	75%	73%	70%	67%
10	**78%**	**78%**	**78%**	**78%**	76%	75%	74%	72%	70%	67%	66%
20	74%	74%	**75%**	**75%**	74%	73%	71%	70%	68%	67%	66%
30	71%	71%	**72%**	**72%**	**72%**	71%	70%	68%	67%	66%	66%
40	68%	69%	69%	**70%**	**70%**	69%	68%	67%	66%	66%	66%
50	66%	**67%**	**67%**	**67%**	**67%**	**67%**	66%	66%	65%	65%	65%
60	64%	**65%**	**65%**	**65%**	**65%**	**65%**	**65%**	**65%**	64%	64%	64%
70	63%	63%	63%	63%	63%	**64%**	63%	63%	63%	63%	63%
80	61%	61%	61%	**62%**	**62%**	**62%**	**62%**	**62%**	**62%**	61%	61%
90	59%	59%	**60%**	**60%**	**60%**	**60%**	**60%**	**60%**	**60%**	**60%**	**60%**
100	58%	58%	58%	58%	58%	58%	58%	58%	58%	58%	58%

Table 5.3: Rate of success before and after re-ranking (2000 dilated samples)

CR: Eigen-Chamfer BI (*Coarse Evaluation*), KR: KPCA Ranking											
	CR	.9CR + .1MR	.8CR + .2MR	.7CR + .3MR	.6CR + .4MR	.5CR + .5MR	.4CR + .6MR	.3CR + .7MR	.2CR + .8MR	.1CR + .9MR	MR
1	55%	61%	**62%**	61%	61%	61%	60%	60%	59%	59%	58%
10	52%	54%	57%	**58%**	**58%**	**58%**	**58%**	**58%**	**58%**	57%	57%
20	51%	52%	53%	55%	56%	**57%**	**57%**	**57%**	**57%**	56%	56%
30	49%	50%	52%	53%	55%	55%	**56%**	**56%**	**56%**	55%	55%
40	49%	49%	50%	51%	53%	54%	**55%**	**55%**	**55%**	54%	54%
50	48%	48%	49%	50%	51%	53%	**54%**	**54%**	**54%**	53%	53%
60	47%	48%	48%	49%	50%	51%	52%	**53%**	52%	52%	52%
70	47%	47%	48%	48%	49%	50%	**51%**	**51%**	**51%**	**51%**	**51%**
80	46%	46%	47%	48%	48%	49%	49%	**50%**	**50%**	**50%**	**50%**
90	46%	46%	46%	47%	47%	47%	47%	**48%**	**48%**	**48%**	**48%**
100	45%	45%	45%	45%	45%	45%	45%	45%	45%	45%	45%
CR: Eigen-Chamfer BI (*Fine Evaluation*), KR: KCCA-Rank											
1	55%	62%	**63%**	**63%**	62%	62%	62%	61%	61%	60%	60%
10	52%	54%	57%	58%	59%	59%	59%	**60%**	**60%**	59%	59%
20	51%	52%	53%	55%	57%	57%	**58%**	**58%**	**58%**	**58%**	**58%**
30	49%	50%	51%	53%	54%	56%	**57%**	**57%**	**57%**	**57%**	**57%**
40	49%	49%	50%	51%	53%	54%	55%	55%	**56%**	55%	55%
50	48%	48%	49%	50%	51%	53%	**54%**	**54%**	**54%**	**54%**	**54%**
60	47%	48%	48%	49%	50%	51%	**52%**	**52%**	**52%**	**52%**	**52%**
70	47%	47%	48%	48%	49%	50%	50%	**51%**	**51%**	**51%**	**51%**
80	46%	46%	47%	47%	48%	48%	**49%**	**49%**	**49%**	**49%**	**49%**
90	46%	46%	46%	**47%**	**47%**	**47%**	**47%**	**47%**	**47%**	**47%**	**47%**
100	45%	45%	45%	45%	45%	45%	45%	45%	45%	45%	45%

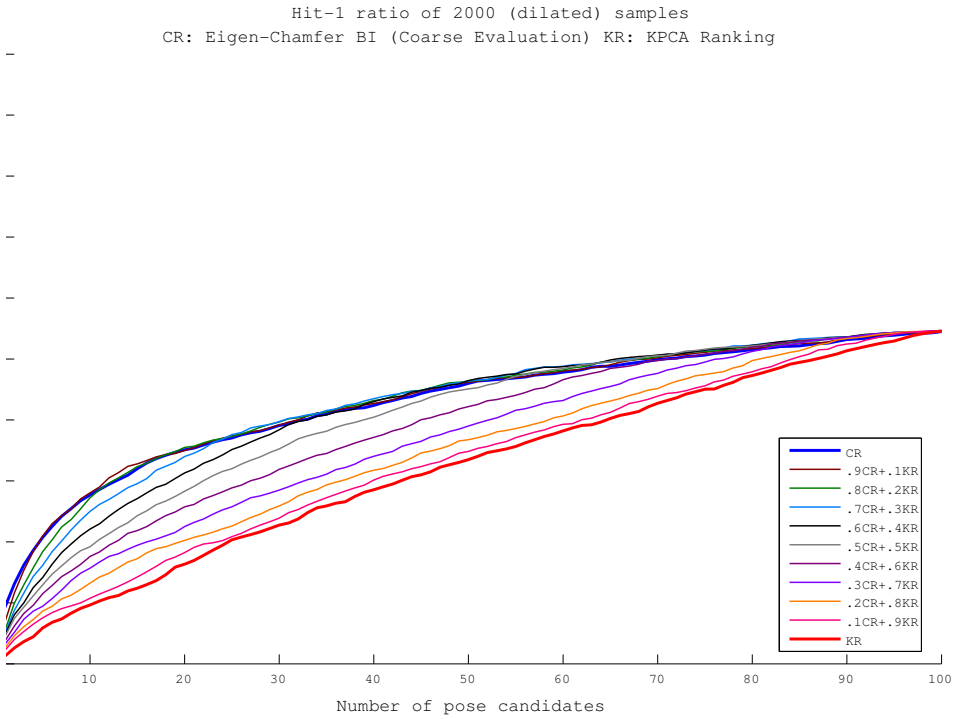


(a)

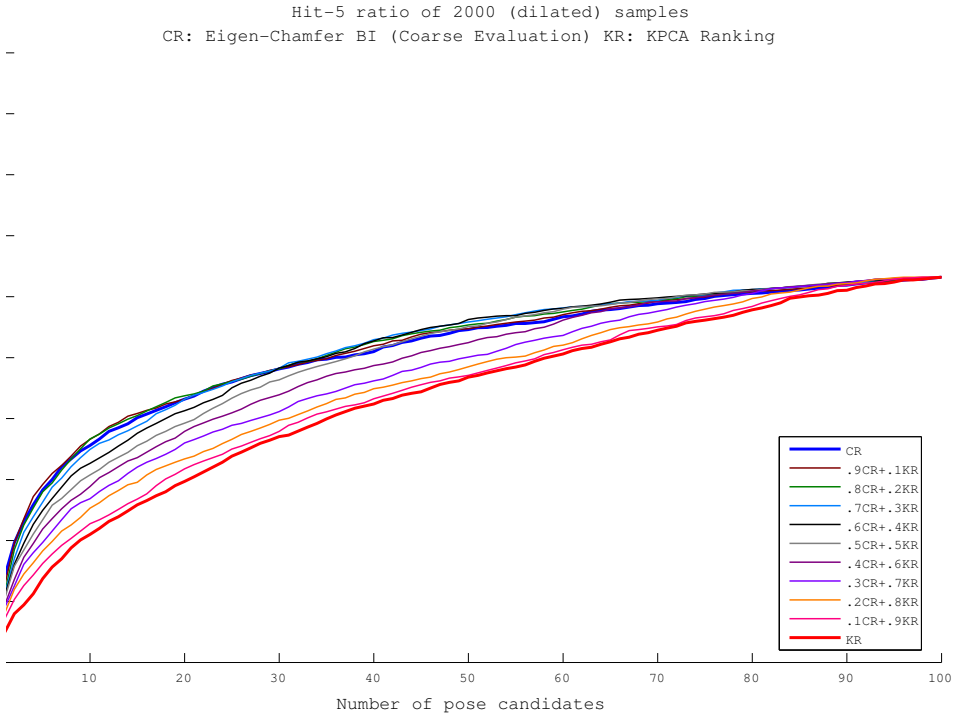


(b)

Figure 5.2: HIT-1 and HIT-5 rates for 2000 samples (KPCA).

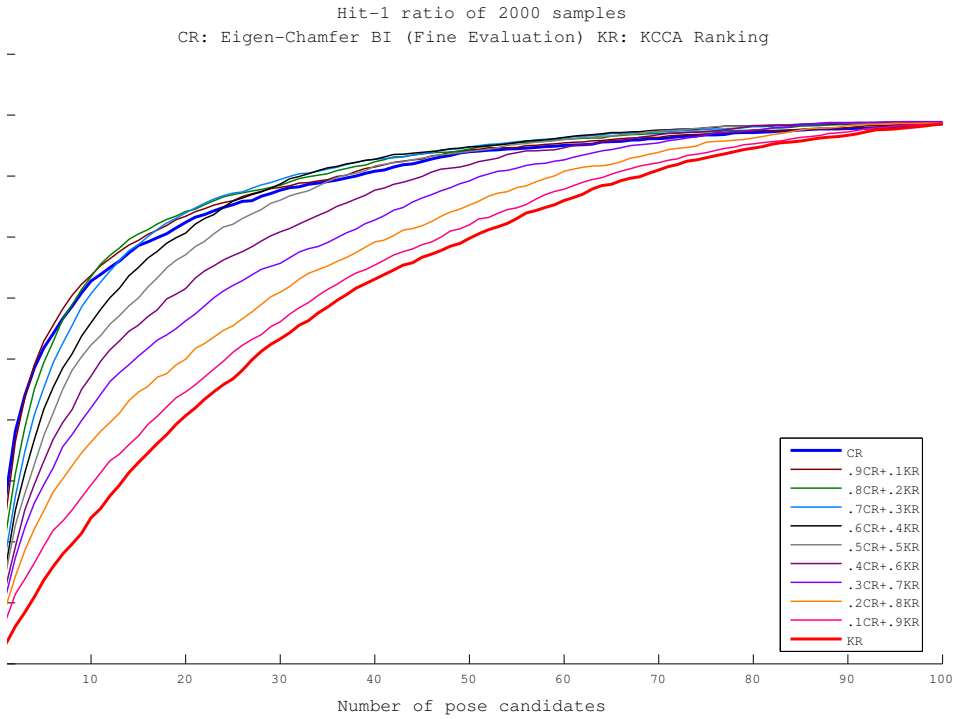


(a)

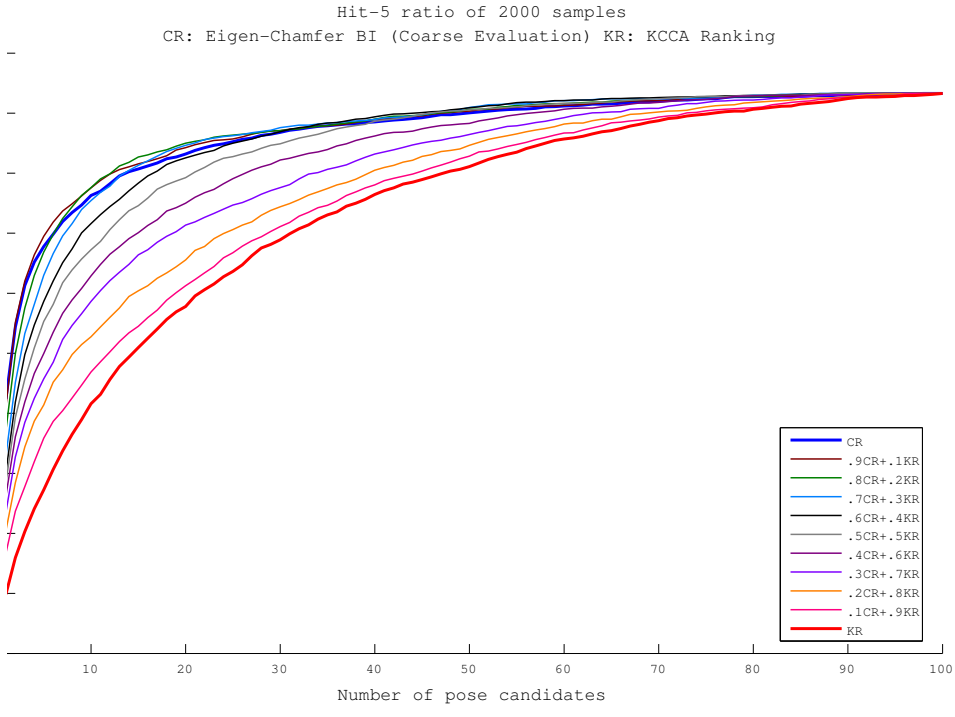


(b)

Figure 5.3: HIT-1 and HIT-5 rates for 2000 dilated samples (KPCA).

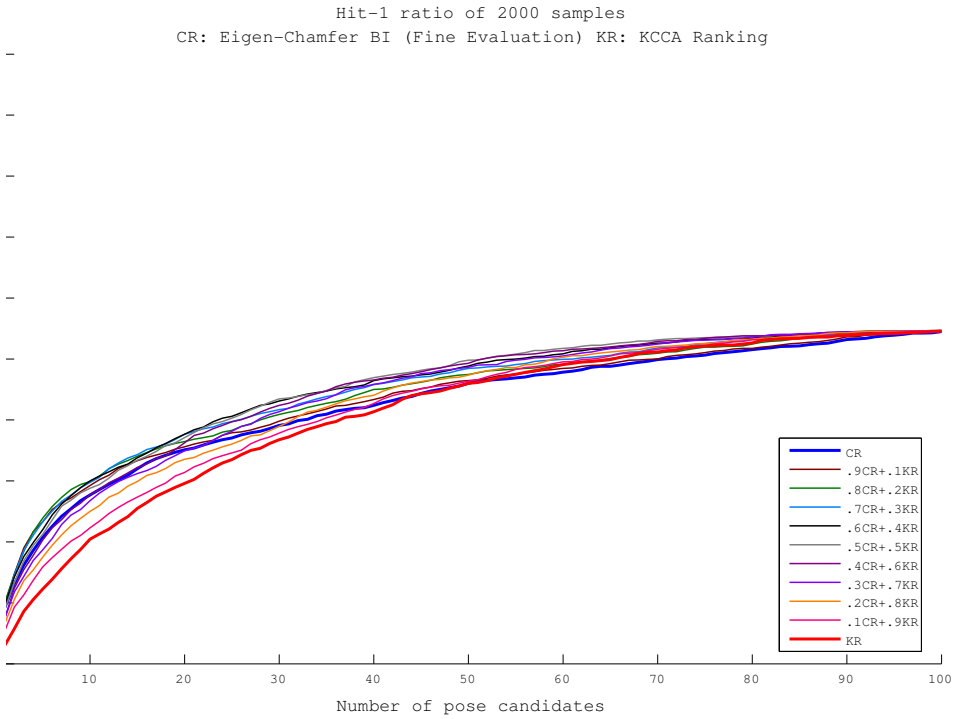


(a)

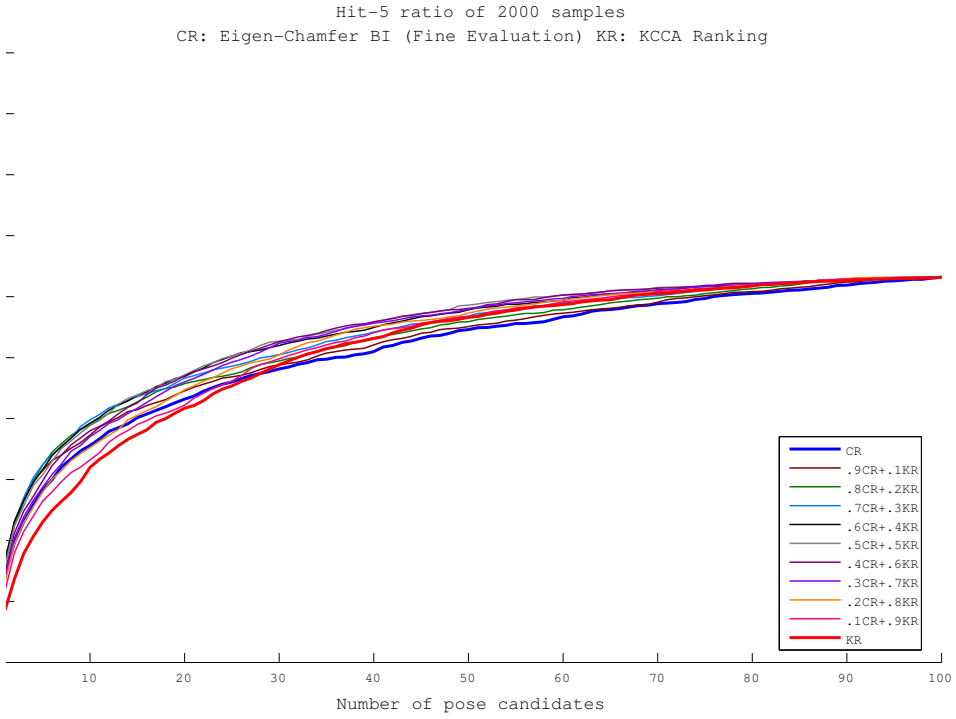


(b)

Figure 5.4: HIT-1 and HIT-5 rates for 2000 samples (KCCA).



(a)



(b)

Figure 5.5: HIT-1 and HIT-5 rates for 2000 dilated samples (KCCA).

Overall, the .7CR+.3KR using KCCA-ranking method had the best performance among all rankings, and the .9CR+.1KR using KPCA-ranking method also outperformed CR in general. Table 5.3 summarizes the ROS for dilated database samples. The CR performed the worst everywhere. Overall, the .3CR+.7KR using KPCA-ranking method had the best performance among all rankings, improving the CR with 5 point. As a summary, the re-ranking methods have shown the capability of ranking candidate poses in a better order.

Figures 5.2 and 5.3 show the graphs of HIT-1 and HIT-5 rates by the KPCA-ranking method, respectively. Differently from the ROS, the HIT-1 and HIT-5 rates by the CR (blue) are almost best everywhere. Figures 5.4 and 5.5 show the graphs of HIT-1 and HIT-5 rates by the KCCA-ranking methods. Differently from the KPCA method, the HIT-1 and HIT-5 rates here have a similar trend as that occurring in ROS. In the case of database samples, .9CR+.1KR (brown) and .8CR+.2KR (green) outperformed the CR. In the case of dilated samples, re-rankings from .9CR+.1KR to .5CR+.5KR outperformed the CR everywhere and from the rank higher rank 40 the CR becomes the worst among all.

From the above results, it can be concluded that under the same conditions (training data, kernel functions and the number of eigenvectors) the KCCA-ranking method outperforms the KPCA-ranking method. When test data are from database, the re-ranking methods only improve the rankings. When test data are images, the re-ranking methods improve clearly.

5.5 Conclusion

We have proposed two kernel subspace methods to re-rank the candidate poses. Kernel PCA and Kernel CCA are employed to learn nonlinear subspaces characterizing the underlying structure of image-pose pairs. Candidate poses are ranked based on the subspace projection. The ranking criteria is subspace projection loss for KPCA-ranking and correlation score for KCCA-ranking. The kernel subspace ranking methods have proved to improve the results by image similarly ranking.

So far only the experiment with database samples (and dilated versions) are conducted. These test data may be well characterized by the learned kernel subspace. A future work is to apply the re-ranking methods to real images.

Chapter 6

Conclusion

6.1 Summary

In this dissertation, we have proposed the time and memory efficient examples-based methods for recovering 3D human body pose from a single silhouette. We demonstrated the effectiveness of our approaches on a variety of synthetic and real-life datasets involving a wide range of pose variations.

In order to enable examples-based method applicable to the complicated vision task of pose-from-silhouette, (1) maintaining sufficient pose samples and (2) effective and efficient matching method, are two critical issues to be addressed. We constructed a large database of two millions of images-pose pairs by combining two half-body databases of each having 1.5×10^4 samples. Depending on this large database, we find out plausible candidate poses that are close to the real pose of the query image. We presented two approximate chamfer matching algorithms — eigen-chamfer and joint-chamfer — to achieve this goal.

The eigen-chamfer uses a subspace approximation of distance transform during computing chamfer distance. The subspace approximation realizes computational efficiency. The reason is that the majority of computing cost are able to be finished in offline step. The online matching involves as small number of arithmetic operations as the dimensions of subspace, while a normal chamfer distance needs several hundred operations depending on the point number of a contour. The memory efficiency is significant because only small number of eigen coefficients and basic distances are stored whereas other solutions need to preserve distance transforms for whole database. The eigen-chamfer match method performs better than the normal chamfer distance when input image has small shape variation compared to database images.

Alternatively, we approximated chamfer distance from knowledge of underlying structure in the human body pose. The joint-chamfer matching method exploits the half-body representation and combination constraint to implicitly represent the full-body pose. This approach basically utilizes a partial-to-whole searching strategy to find full-body pose candidates. From the retrieved half-body candidates by using partial chamfer matching, the pre-computed combination constraint picks out the valid combinations, which are further evaluated. The further evaluation is also efficient because it involves light computational cost of combination of half-body distances. Thus, this approach is computationally extremely efficient and the current implementation can work in near real-time. Additionally, this is efficient in memory complexity because the size of half-body database is clearly compact.

Another contribution of this work is re-ranking candidate poses using kernel subspace. We use kernel PCA and kernel CCA to learn nonlinear subspaces characterizing the underlying structure of image-pose pairs. Candidate poses are ranked based on the subspace projection with the ranking criteria being subspace projection loss for KPCA-ranking and correlation score for KCCA-ranking. The kernel subspace ranking methods are complementary to image similarly ranking so that the combination of them perform better than either does alone.

6.2 Future Work

Our work provides an important step towards solving complicated 3D human pose recovery problem. Several interesting problems remain for future work. Beside the weak combination constraint, there are many stronger hidden factors used by human perception such as action category and body orientation, by which the estimation performance can be further improved. Within the context of 3D human pose estimation, one important topic is how to adapt the system to real images or video sequences. In addition, we have interest in applying approximate chamfer matching algorithms to other large-scale / large-class object recognition problems.

Acknowledgements

First and foremost I would like to thank my supervisor, Professor Noboru Ohnishi of Nagoya University for his instructive guidance, constant personal encouragement and infinite patience during every stage of this research, and giving me the freedom to choose and explore my directions and style of research. I am grateful to the other members of the judging committee: Professors Yasuhito Suenaga, Hiroshi Murase, and Associate Professor Yoshinori Takeuchi, of Nagoya University, for their insightful comments on my research. I am very grateful to Associate Professor Hiroaki Kudo, Assistant Professor Tetsuya Matsumoto and again Associate Professor Yoshinori Takeuchi for the continuous scientific advices and constant support. I would also greatly thank Professor Toshimitsu Tanaka of Meijo University for the supervision of my Master's thesis. I am also indebted to Emeritus Professor Noboru Sugie of Nagoya University for constant support and encouragements.

The Ohnishi laboratory has been a wonderful place to work and live. I am sincerely grateful for the friendship and companion from Md. Khayrul Bashar, Ito Masanori, Kento Nishibori, Seongjun Yang, Ukrit Watchareeruetai and Xuanxuan Cheng.

Finally, I want to thank my family who have constantly supported and inspired me. In particular I thank my mother for her unstinting belief in me. I also thank my eldest sister and brother-in-law. Without their support, my studies would not have been possible.

Appendix A

Biovision BVH Format

The BVH file format was originally developed by Biovision, a motion capture services company, as a way to provide motion capture data to their customers. A BVH file has two parts, a header section which describes the hierarchy and initial pose of the skeleton; and a data section which contains the motion data. See an example BVH file below.

```
HIERARCHY
ROOT Hips {
    OFFSET  0.00    0.00    0.00
    CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
    JOINT Chest
    {
        OFFSET   0.00    5.21    0.00
        CHANNELS 3 Zrotation Xrotation Yrotation
        JOINT Neck
        {
            OFFSET   0.00    18.65   0.00
            CHANNELS 3 Zrotation Xrotation Yrotation
            JOINT Head
            {
                OFFSET   0.00    5.45    0.00
                CHANNELS 3 Zrotation Xrotation Yrotation
                End Site
                {
                    OFFSET   0.00    3.87    0.00
                }
            }
        }
        ...
    }
    ...
}
MOTION Frames: 2
Frame Time: 0.033333
8.03 35.01 88.36 -3.41 14.78 -164.35 7.81 35.10...
```

```
0.70 0.37 0.00 -8.62 0.00 -21.82 -87.31 ...
```

The start of the header section begins with the keyword ”HIERARCHY”. The following line starts with the keyword ”ROOT” followed by the name of the root segment of the hierarchy to be defined. The BVH format now becomes a recursive definition. Each segment of the hierarchy contains some data relevant to just that segment then it recursively defines its children. The first piece of information of a segment is the offset of that segment from its parent, or in the case of the root object the offset will generally be zero. The offset is specified by the keyword ”OFFSET” followed by the X,Y and Z offset of the segment from its parent. The offset information also indicates the length and direction used for drawing the parent segment. The line following the offset contains the channel header information. This has the ”CHANNELS” keyword followed by a number indicating the number of channels and then a list of that many labels indicating the type of each channel. On the line of data following the channels specification there can be one of two keywords, either you will find the ”JOINT” keyword or you will see the ”End Site” keyword. A joint definition is identical to the root definition except for the number of channels. This is where the recursion takes place, the rest of the parsing of the joint information proceeds just like a root. The end site information ends the recursion and indicates that the current segment is an end effector (has no children). The end site definition provides one more bit of information, it gives the length of the preceding segment just like the offset of a child defines the length and direction of its parents segment. For the BVH hierarchy, the world space is defined as a right handed coordinate system with the Y axis as the world up vector. Thus you will typically find that BVH skeletal segments are aligned along the Y or negative Y axis.

The motion section begins with the keyword ”MOTION” on a line by itself. This line is followed by a line indicating the number of frames that are in the file. On the line after the frames definition is the ”Frame Time:” definition, this indicates the sampling rate of the data. In the above example BVH file the sample rate is given as 0.033333, this is 30 frames a second the usual rate of sampling in a BVH file. The rest of the file contains the actual motion data. Each line is one sample of motion data. The numbers appear in the order of the channel specifications as the skeleton hierarchy was parsed.

The above introduction to BVH format is modified from the online material: http://www.cs.wisc.edu/graphics/Courses/cs-838-1999/Jeff/BVH.html

Appendix B

Computing Half-Body Combination Constraints

Notation p^u(p^l): *upper(lower)-body pose*, θ: *body orientation*, N:*the number of half-body poses*.

Calculating threshold distances of similar half-body poses

for$(i = 1; i \leqslant N - 1; i++)$

- $uDist(i) = (||p^u_{i+1} - p^u_i||)$ (Euclidean distance between successive upper-body poses);
- $lDist(i) = (||p^l_{i+1} - p^l_i||)$ (Euclidean distance between successive lower-body poses);

$\overrightarrow{uDist} = sort(uDist)$ (sort distance in ascending order);

$uThresholdDist = mean(\overrightarrow{uDist}(1 : N * 0.95))$ (mean value of 95% smaller distances);

$\overrightarrow{lDist} = sort(lDist)$ (sort distance in ascending order);

$lThresholdDist = mean(\overrightarrow{lDist}(1 : N * 0.95))$; (mean value of 95% smaller distances).

Calculating the half-body combination constraints

Create a $N \times N$ matrix $\mathbf{CT}$ to store constraint information, set $\mathbf{ct}_{ij} = 0$ for $\forall i, j$

for$(i = 1; i \leqslant N; i++)$

- Initialize neighbor sets $\mathcal{N}_u = \emptyset$, $\mathcal{N}_l = \emptyset$;
- for$(j = 1; j \leqslant N$ && $(\theta_j - 30° \leqslant \theta_i \leqslant \theta_j + 30°); j++)$
  - if $(||p^u_i - p^u_j|| < uThresholdDist)$ add j into $\mathcal{N}_u$;
  - if $(||p^l_i - p^l_j|| < lThresholdDist)$ add j into $\mathcal{N}_l$;
- Set $\mathbf{ct}_{ij} = 1$, for $j \in \mathcal{N}_u \cup \mathcal{N}_l$, meaning p^u_i and p^l_j are combinative.

Appendix C

Principal Component Analysis

Principal Component Analysis (PCA) [30] is a dimensionality reduction technique based on extracting the desired number of principal components of the multi-dimensional data. The first principal component is the linear combination of the original variables achieving the maximum variance; the k-th principal component is the linear combination with the k^{th} highest variance, subject to being orthogonal to the $k-1$ first principal components.

PCA is closely related to the Karhunen-Loève Transform (KLT), which was derived in the signal processing context as the orthogonal transform with the basis $\mathbf{\Phi} = [\phi_1, \ldots, \phi_N]'$ for any $k \leq N$ minimizes the average L_2 reconstruction error for data points x

$$\epsilon(\mathbf{x}) = \|\mathbf{x} - \Sigma_{i=1}^{k}(\phi_i{}'\mathbf{x})\phi_i\|. \tag{C.1}$$

One can show that, under the assumption that the data is zero-mean, the formulations of PCA and KLT are identical. Without loss of generality we will hereafter assume that the data is indeed zero-mean, that is, the mean c is always subtracted from the data. The basis vectors in KLT can be calculated in the following way. Let $\mathbf{X}$ be the $N \times M$ data matrix whose columns $\mathbf{x}_1, \ldots, \mathbf{x}_M$ are observations of a signal embedded in R^N; in the context of image, M is the number of train images and $N = mn$ is the number of pixels in an image. The KLT basis Φ is obtained by solving the eigenvalue problem $\Lambda = \mathbf{\Phi}'\Sigma\mathbf{\Phi}$, where Σ is the covariance matrix of the data

$$\mathbf{\Sigma} = \frac{1}{M}\Sigma_{i=1}^{M}\mathbf{x}_i{}'\mathbf{x}_i, \tag{C.2}$$

$\mathbf{\Phi} = [\phi_1 \ldots, \phi_m]'$ is the eigenvector matrix of $\mathbf{\Sigma}$, and Λ is the diagonal matrix with eigenvalues $\lambda_1 \geq \ldots \lambda_N$ of $\mathbf{\Sigma}$ on its main diagonal, so that ϕ_j is the eigenvector corresponding to the j^{th} largest eigenvalue. Then it can be shown that the eigenvalue λ_i is the variance of the data projected on ϕ_i.

Thus, to perform PCA and extract k principal components of the data, one must project the data onto ϕ_k – the first k columns of the KLT basis Φ, which correspond to the k highest eigenvalues of Σ. This can be seen as a linear projection $R^N \rightarrow R^k$ that retains the maximum variance of the signal. Another important property of PCA is that it decorrelates the data: the covariance matrix of ${\phi_k}'\mathbf{X}$ is always diagonal.

PCA may be implemented via Singular Value Decomposition (SVD): The SVD of an $M \times N$ matrix $X(M > N)$ is given by

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}', \tag{C.3}$$

where the $M \times N$ matrix $\mathbf{U}$ and the $N \times N$ matrix $\mathbf{V}$ have orthonormal columns, and the $N \times N$ matrix $\mathbf{D}$ has the singular values of $\mathbf{X}$ on its main diagonal and zero elsewhere. It can be shown that $\mathbf{U} = \mathbf{\Phi}$, so that SVD allows efficient and robust computation of PCA without the need to estimate the data covariance matrix $\mathbf{\Sigma}$. When the number of examples M is much smaller than the dimension N, this is a crucial advantage.

Bibliography

[1] Microsoft expression graphic designer(may 2006 ctp).

[2] Ankur Agarwal and Bill Triggs. Recovering 3d human pose from monocular images. *IEEE Trans. Pattern Anal. Mach. Intell*, 28(1):44–58, 2006.

[3] Jake K. Aggarwal and Quin Cai. Human motion analysis: A review. *Computer Vision and Image Understanding*, 73(3):428–440, 1999.

[4] Shotaro Akaho. A kernel method for canonical correlation analysis. In *International Meeting of Psychometric Society*, 2001.

[5] V. Athitsos, J. Alon, and S. Sclaroff. Efficient nearest neighbor classification using a cascade of approximate similarity measures. In *Computer Vision and Pattern Recognition*, pages 486–493, 2005.

[6] V. Athitsos, J. Alon, S. Sclaroff, and G. Kollios. Boostmap: A method for efficient approximate similarity rankings. In *Computer Vision and Pattern Recognition*, volume 2, pages 268–275, 2004.

[7] C. Barron and I. Kakadiaris. Estimating anthropometry and pose from a single image. In *Computer Vision and Pattern Recognition*, pages 669–676, 2000.

[8] H.G. Barrow, J.M. Tenenbaum, R.C. Bolles, and H.C. Wolf. Parametric correspondence and chamfer matching:two new techniques for image matching. *IJCAI*, pages 659–663, 1977.

[9] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell*, 24(4):509–522, 2002.

[10] P. Besl and N. McKay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14:239–256, 1992.

[11] Hui Cao, Noboru Ohnishi, Yoshinori Takeuchi, Tetsuya Matsumoto, and Hiroaki Kudo. Fast human pose retrieval using approximate chamfer distance. *IEEJ Transactions on Electronics, Information and Systems*, 126(12):1490–1496, 2006.

[12] Hui Cao, Noboru Ohnishi, Yoshinori Takeuchi, Tetsuya Matsumoto, and Hiroaki Kudo. Retrieval-combination approach to estimating 3d human pose from a monocular image. *to appear in The Journal of The Institute of Image Information and Television Engineers*, 61(2), 2007.

[13] Hui Cao, Noboru Ohnishi, Toshimitsu Tanaka, and Noboru Sugie. Static image based human pose recovery by optimal candidate decision. In *The IEICE workshop on Pattern Recognition Media Understanding (PRMU)*, pages 169–172, 2004.

[14] Hui Cao, Toshimitsu Tanaka, and Noboru Sugie. Data imputation method for human motion capture from monocular images. In *The Third IASTED International Conference on Artificial Intelligence and Applications*, pages 169–172, Benalmadena, Spain, 2003.

[15] C. Cedras and M. Shah. Motion-based recognition: A survey. *Image and Vision Computing*, 13(2):129–155, 1995.

[16] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[17] Richard O. Duda, D. G. Stork, and Peter E. Hart. Pattern classification (second edition). *John Wiley and Sons*, 2001.

[18] e frontier / Curious Labs. Poser 6: The premiere 3d figure design and animation solution, 2005.

[19] A. Elgammal and C.S. Lee. Inferring 3d body pose from silhouettes using activity manifold learning. In *Computer Vision and Pattern Recognition*, 2004.

[20] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61(1), 2005.

[21] Ming-Hsuan Yang Gang Hua and Ying Wu. Learning to estimate human pose with data driven belief propagation. In *Computer Vision and Pattern Recognition*, volume 2, pages 747–754, 2005.

[22] Dariu Gavrila. The visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, 73(1), 1999.

[23] Keith Grochow, Steven L. Martin, Aaron Hertzmann, and Zoran Popovic. Style-based inverse kinematics. *ACM Trans. Graph*, 23(3):522–531, 2004.

[24] David R. Hardoon, Sandor Szedm'ak, and John Shawe-Taylor. Canonical correlation analysis; an overview with application to learning methods. Technical report, Department of Computer Science Royal Holloway, University of London, 2003.

[25] D. Hogg. Model-based vision: A program to see a walking person. *Image and Vision Computing*, 1(1):5–20, 1983.

[26] Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28:321–377, 1936.

[27] Kuei Hu. Visual pattern recognition by moment invariants. *IRE Trans. on Information Theory*, 8:79187, 1962.

[28] Daniel P. Huttenlocher, Ryan H. Lilien, and Clark F. Olson. View-based recognition using an eigenspace approximation to the hausdorff measure. *IEEE Trans. Pattern Anal. Mach. Intell*, 21(9):951–955, 1999.

[29] S. Ioffe and D. Forsyth. Probabilistic methods for finding people. *IJCV*, 43(1):45–68, 2001.

[30] I. T. Jolliffe. *Principal Component Analysis*. Springer Verlag, New York, 1986.

[31] Yoshinari Kameda, Michihiko Minoh, and Katsuo Ikeda. A pose estimation method for an articulated object from its silhouette image. *The Transactions of the IEICE*, J79-D-II(1):26–35, 1996.

[32] K. I. Kim, M. Franz, and B. Scholkopf. Iterative kernel principal component analysis for image modeling. *IEEE Trans. Pattern Anal. Mach. Intell*, 27(9):1351–1366, 2005.

[33] Gregory Shakhnarovich Kristen Grauman and Trevor Darrell. Inferring 3d structure with a statistical image-based shape model. In *IEEE International Conference on Computer Vision*, 2003.

[34] James T. Kwok and Ivor W. Tsang. The pre-image problem in kernel methods. In *ICML*, pages 408–415, 2003.

[35] Pei Ling Lai and Colin Fyfe. Kernel and nonlinear canonical correlation analysis. In *IJCNN (4)*, 2000.

[36] Neil D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *NIPS*, 2003.

[37] H. J. Lee and Z. Chen. Determination of 3D human body postures from a single view. *Computer Vision, Graphics and Image Processing*, 30:148–168, 1985.

[38] Mun Wai Lee and Isaac Cohen. A model-based approach for estimating human 3d poses in static images. *IEEE Trans. Pattern Anal. Mach. Intell*, 28(6):905–916, 2006.

[39] John MacCormick and Andrew Blake. A probabilistic exclusion principle for tracking multiple objects. *International Journal of Computer Vision*, 39(1):57–71, 2000.

[40] S. Mika, B. Schölkopf, A. J. Smola, K.-R. Müller, M. Scholz, and G. Rätsch. Kernel pca and de-noising in feature spaces. In *NIPS*, 1999.

[41] K. Mikolajczyk, C. Schmid, and A. Zisserman. Human detection based on a probabilistic assembly of robust part detectors. In *European Conference on Computer Vision*, volume 1, pages 69–82, 2004.

[42] T. B. Moeslund, A. Hilton, and V. Kruger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 103(2-3):90–126, November 2006.

[43] Thomas B. Moeslund and Erik Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3):231–268, 2001.

[44] Anuj Mohan, Constantine Papageorgiou, and Tomaso Poggio. Example-based object detection in images by components. *IEEE Trans. Pattern Anal. Mach. Intell*, 23(4):349–361, 2001.

[45] G. Mori, X. Ren, A.A. Efros, and J. Malik. Recovering human body configurations: Combining segmentation and recognition. In *Computer Vision and Pattern Recognition*, volume 2, pages 326–333, 2004.

[46] Greg Mori and Jitendra Malik. Recovering 3d human body configurations using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell*, 28(7):1052–1062, 2006.

[47] Hiroshi Murase and Shree K. Nayar. Visual learning and recognition of 3-d objects from appearances. *Int. Journal of Computer Vision*, 14:5–24, 1995.

[48] Hiroshi Murase and Shree K. Nayar. Learning by a generation approach to appearance-based object recognition. 1:24–29, 1996.

[49] Vasu Parameswaran and Rama Chellappa. View independent human body pose estimation from a single perspective image. In *Computer Vision and Pattern Recognition*, pages 16–22, 2004.

[50] R. Plaenkers and P. Fua. Articulated soft objects for multi-view shape and motion capture. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 25(10), 2003.

[51] Liu Ren, Gregory Shakhnarovich, Jessica K. Hodgins, Hanspeter Pfister, and Paul Viola. Learning silhouette features for control of human motion. In *ACM Transactions on Graphics*, October 2005.

[52] Xiaofeng Ren, Alexander C. Berg, and Jitendra Malik. Recovering human body configurations using pairwise constraints between parts. In *International Conference on Computer Vision*, volume 1, pages 824–831, 2005.

[53] BVH File Repository. http://www.centralsource.com/blender/bvh/.

[54] R. Ronfard, C. Schmid, and B. Triggs. Learning to parse pictures of people. In *European Conference on Computer Vision*, pages 700–714, 2002.

[55] R. Rosales and S. Sclaroff. Specialized mappings and the estimation of human body pose from a single image. In *IEEE Workshop on Human Motion*, pages 19–24, 2000.

[56] Romer Rosales. *The Specialized Mappings Architecture with Applications to Vision-Based Estimation of Articulated Body Pose*. PhD thesis, Boston University, 2002.

[57] H. Sahbi. Kernel pca for similarity invariant shape recognition. *Neurocomputing*, 2006.

[58] B. Scholkopf, S. Mika, A. Smola, G. Ratsch, and K. R. Muller. Kernel pca pattern reconstruction via approximate pre-images. In L. Niklasson, M. Boden, and T. Ziemke, editors, *The 8th International Conference on Artificial Neural Networks*, pages 147–152, Berlin, 1998. Springer Verlag.

[59] B. Scholkopf, A. Smola, and K. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[60] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter sensitive hashing. In *IEEE International Conference on Computer Vision*, volume 2, pages 750–757, 2003.

[61] Aaron P. Shon, Keith Grochow, Aaron Hertzmann, and Rajesh P. N. Rao. Learning shared latent structure for image synthesis and robotic imitation. In *NIPS*, 2005.

[62] Leonid Sigal, Michael Isard, Benjamin H. Sigelman, and Michael J. Black. Attractive people: Assembling loose-limbed models using non-parametric belief propagation. In *NIPS*, 2003.

[63] Cristian Sminchisescu, Atul Kanaujia, Zhiguo Li, and Dimitris N. Metaxas. Discriminative density propagation for 3d human motion estimation. In *Computer Vision and Pattern Recognition*, volume 1, pages 390–397, 2005.

[64] Cristian Sminchisescu and Bill Triggs. Estimating articulated human motion with covariance scaled sampling. *I. J. Robotic Res.*, 22(6):371–392, 2003.

[65] Yang Song, Luis Goncalves, and Pietro Perona. Unsupervised learning of human motion. *IEEE Trans. Pattern Anal. Mach. Intell*, 25(7):814–827, 2003.

[66] T. Tangkuampien and D. Suter. Human motion de-noising via greedy kernel principal component analysis filtering. In *International Conference on Pattern Recognition*, pages III: 457–460, 2006.

[67] T. Tangkuampien and D. Suter. Real-time human pose inference using kernel principal component pre-image approximations. In *British Machine Vision Conference*, 2006.

[68] C. Taylor. Reconstruction of articulated objects from point correspondences in a single uncalibrated image. In *Computer Vision and Pattern Recognition*, pages 677–684, 2000.

[69] A. Thayananthan, R. Navaratnam, B. Stenger, P.H.S. Torr, and R. Cipolla. Multivariate relevance vector machines for tracking. In *European Conference on Computer Vision*, volume 3, pages 124–138, 2006.

[70] Alexei Vinokourov, John Shawe-Taylor, and Nello Cristianini. Inferring a semantic representation of text via cross-language correlation analysis. In *NIPS*, page 1473, 2002.

[71] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition*, 1:511–518, 2001.

[72] Jason Weston, Olivier Chapelle, André Elisseeff, Bernhard Schölkopf, and Vladimir Vapnik. Kernel dependency estimation. In *NIPS*, pages 873–880, 2002.

[73] Jason Weston, Bernhard Schölkopf, and Olivier Bousquet. Joint kernel maps. In *IWANN*, volume 3512, pages 176–191, 2005.

[74] S. Dambreville Y. Rathi and A. Tannenbaum. Statistical shape analysis using kernel pca. In *SPIE Symposium on Electronic Imaging*, 2006.