DOI: 10.1111/cgf.12750
Pacific Graphics 2015
N. J. Mitra, J. Stam, and K. Xu
(Guest Editors)

Volume 34 (2015), Number 7

A Suggestive Interface for Sketch-based Character Posing

P. Lv[1] P. J. Wang[2] W. W. Xu[3] J. X. Chai[4] M. M. Zhang[5] Z. G. Pan[3,*] M. L. Xu[1]

[1]College of Information Engineering, Zhengzhou University, China
[2]College of Computer Science and Engineering, Dalian Nationalities University, China
[3]Digital Media and HCI Research Center, Hangzhou Normal University, China
[4]Department of Computer Science and Engineering, Texas A&M University, USA
[5]College of Computer Science and Technology, Zhejiang University, China

Abstract

We present a user-friendly suggestive interface for sketch-based character posing. Our interface provides suggestive information on the sketching canvas in succession by combining image retrieval technique with 3D character posing, while the user is drawing. The system highlights the canvas region where the user should draw on and constrains the user's sketches in a reasonable solution space. This is based on an efficient image descriptor, which is used to measure the distance between the user's sketch and 2D views of 3D poses. In order to achieve faster query response, local sensitive hashing is involved in our system. In addition, sampling-based optimization algorithm is adopted to synthesize and optimize the retrieved 3D pose to match the user's sketches the best. Experiments show that our interface can provide smooth suggestive information to improve the reality of sketching poses and shorten the time required for 3D posing.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Animations

1 Introduction

Human character posing is one of fundamental functions in computer animation and has many applications-for example, rapid prototyping of 3D human animation, real-time character motion control, interactive humanoid robot manipulation and human-computer interactions. Traditionally, the user can obtain desired poses by directly manipulating skeleton joints underneath the character body using forward kinematics, or by dragging end-effector joints to specified position with a follow-up parent joints adaption driven by inverse kinematics, such as [RISC01, GMHP04, WC11]. However, these methods work completely in 3D scenario, which are difficult for inexperienced users. Consequently, the user needs to be aware of basic 3D graphics knowledge and gets enough training before using these systems and tools.

Conversely, sketch-based interface for character posing, which attempts to simulate the pencil-and-paper interface, is intuitive, simple, efficient and also can be easily accepted and employed by artists and animators. Unfortunately, there are two major obstacles for the wide use of the sketching interface. Firstly, it is an inherent ambiguous problem of interpreting 3D poses from 2D sketches due to lack of depth info. Secondly, when sketching, most users can not draw body joints and body proportion accurately. Some representative work have been done to solve above problems, for example, Choi et al. [CYI*12] proposed a human motion retrieval and visualizing system by using 2D stick figures as a unified medium for both sketches and human motion data. Although this system offered an interactive way of obtaining similar poses according to sketches, it still inherently was one kind of motion searching methods, which failed to address the problem of user experience in the 3D posing process. Lee et al. [LZC11] proposed a suggestive interface for free-hand drawing by composing relevant images into a shadow as hints. The main purpose of ShadowDraw was to improve the quality of users' sketching and the retrieved results were limited to existing data in the database. For character posing, generating new poses, which are not included in existing database, is necessary. Moreover, ShadowDraw built on carefully-collected images which should

have enough salient changeable edges. When these edges are spatially composed into shadow images, their inherent connectivity becomes less important. For 2D character poses, especially for 2D stick figures, they consist of fixed number edges, which are one-pixel width lines and have no salient changes. It is very difficult to blend these single lines, which may vary greatly in image plane, to generate meaningful shadow. More important, these edges have obvious inherent connectivity, which cannot be ignored. Therefore, ShadowDraw is not applicable to 3D human posing.

In this paper, we propose an interactive sketch-based character posing system to address above both problems at the same time, which instantly responses to each sketching stroke by showing potential candidate poses rendered on the canvas. Compared with ShadowDraw, our method is more difficult in that we need to match 2D sketches with 3D character poses in order to generate suggestive hints. To address this problem, we first collect representative poses from motion database, and then render these 3D poses into 2D images under appropriate views. The intractable 3D interpretation of 2D sketch is then converted into an image matching problem, which is easier in couple with our well-designed efficient image descriptor for measuring the distance between the user's sketch and those images in the database.

Specifically, during the sketching process, the incomplete sketch drawn by the user is sent to the pose database to query its nearest neighbors. The retrieved poses ranked at the top are then rendered into stylized shadow lines, and displayed on the canvas as the suggestive information. The advantages are three-fold:

- The interface can provide suggestive hints to highlight the possible region where the user should draw.
- The interface can guide the user to sketch within a reasonable solution space. And, since the hint lines and the confirmed rendered bones are both displayed on the same canvas, the proportion accuracy problem is well solved.
- The interface enable users, especially the inexperienced ones, to sketch more accurate strokes, based on suggestive and predicted hint poses, which can potentially match the 2D sketch.

If the 2D sketch cannot match anyone of these hint poses, our system will leverage a sampling-based optimization algorithm to synthesize and optimize the retrieved 3D pose to match the user's sketches the best. This important characteristic, combined with above three points focusing on user experience, distinguish our system as an efficient human posing method instead of a pose retrieval method such as [CYI*12] does.

2 Related work

Due to its simplicity and intuition, the sketching interface has been applied in various areas successfully, such as sketch-based image and shape retrieval [CNM05a, EHBA11, CWZZ11, SXY*11, ERB*12]. Using the sketch-based image retrieval technique, Chen et al. [CmCT*09] and Eitz et al. [ERH*11] developed their own image synthesis systems. On the other hand, the sketch-based shape retrieval approaches were adopted to compose 3D scene in [SI07, LF08]. Different from these retrieval methods built on the measurement of geometric similarity between the sketch and the content, the sketch recognition is also a critical problem needed to be solved in semantic level. Eitz et al. [EHA12] discussed the topic concerning how to sketch objects in perceptual level and developed an effective sketch recognition system based on a large amount of human sketches study. Huang et al. [HFL14] explored a data-driven approach to segment and label freehand sketches without any semantic segmentation information as input. Schneider et al. [ST14] proposed a method for sketch classification based on Fisher Vector, which can have nearly same performance with humans.

The sketching interface is also used by researchers to assist creative or corrective drawings. iCanDraw [DPH10] helps users draw faces with a tutorial approach. A sketching beautification system [OK11] infered smooth lines from a sketch consisting of short, overlapping strokes , and a sketch-based digital painting system [XCW14] can automatically complete the tedious repetition while allowing user control. Elasticurves [TSB11] neatened sketches by smoothing strokes dynamically based on stroke speed. ShadowDraw [LZC11] used collected images to aid users in drawing by providing shadows of similar drawings. Fan et al. [FWX*13] presented a novel sketch-based modeling system similar to ShadowDraw which allows novice users to create 3D custom models by assembling parts based on a database of pre-segmented 3D models. Levi et al. [LG13] present the ArtiSketch system, which allows the conversion of a wealth of existing 2D content into 3D content. A novice user may describe a 3D model as a set of articulated 2D sketches of a shape from different viewpoints using this sytem. ArtiSketch then automatically converts the sketches to an articulated 3D object. Sucontphunt et al. [SMND08, STDN10] present a series of interactive 3D facial expression posing system through 2D portrait manipulation, which can save users plenty of time. In this paper, we restrict our attention to sketch-based character posing. Different from these works, we focus on sketch-based character posing in this paper.

In sketch-based character posing systems, the analysis and recognition of users' strokes are essential to obtain required information for the subsequent 3D pose reconstruction. In [DAC*03], they involved a template skeleton to automatically label the stick figure drawn by users automatically. Afterwards, they applied a set of joint angle constraints to prune those wrong reconstructed 3D configurations. Thorne et al. [TBvdP04] used a more complex algorithm to infer users' sketches as a skeleton, but they did not address the posing problem. In addition, some researchers [JSMH12] asked the user to label the joints and limbs of hand-drawn character manually in their scenario. Choi et al. [CYI*12] realized the goal of sketching a character pose by converting this problem to the nearest neighbor search. They converted motion data to a sequence of stick figures first. And then

a feature-based comparison method between these stick figures and users' sketches was applied to search nearest neighbor results. Our method is inspired by these work, but gets rid of the skeleton recognition and joint labeling problem, which are more prone to error. Instead we use the pre-defined sketching order and an efficient image descriptor to achieve the same purpose.

3 System Overview

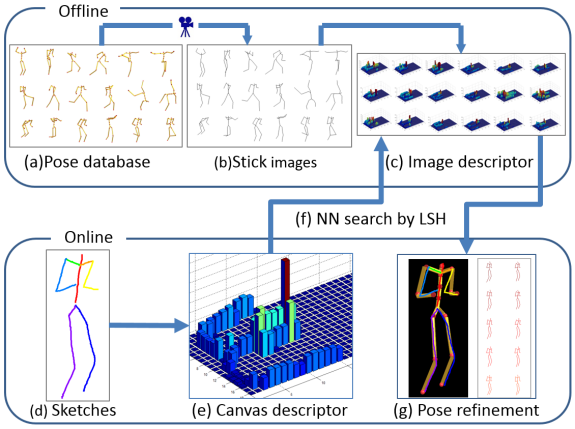


Figure 1: *System overview. An illustration of (a) 3D pose database, (b) stick images under optimal view, (c) database image descriptor, (d) users' sketches, (e) canvas descriptor, (f) nearest neighbor search by local sensitive hashing and (g) pose refinement based on sampling method.*

This paper presents a suggestive interface to guide the sketching process for human poses. Our interface is a data-driven approach which can automatically generate guiding strokes. These strokes instruct users to draw skeletons with right proportion, as shown in Figure 1. While the sketching evolves, our interface queries the 2D pose image database to get the poses that are closely matched with the user's sketches, and then render these 3D poses along with 2D images on the canvas as a guidance for next stroke. As a suggestive interface, our system is well designed to support incomplete sketched pose matching by calculating the best guidance for the next stroke. To this end, we propose to compare the sketches with the rendered images at semantic body part level, and balance bones with different lengths based on a smart weighting scheme in favor of the best matching result.

In order to avoid the cumbersome semantic understanding problem caused by free-style sketching, we constrain the user to draw skeletons in a specific order, a policy that is widely adopted by pose and motion retrieval methods [DAC*03,TBvdP04,LIM*12,CLAL12,XTFX15]. After users finish skeleton sketching, the retrieved top rated pose will go through a pose refinement process to match the user's sketches the best. Since there might be no exact matching pose in the database, this refinement process is essential.

Overall, our system consists of three major technical components as shown in Figure 1: motion data preprocessing, image descriptor estimation and image retrieval based on this descriptor, and sampling-based pose refinement. We will unfold each component in detail in following sections.

4 Motion Data Preprocessing

Our pose data is selected from the CMU mocap database, which contains a corpus of motion data, such as walking, running, dancing and basketball playing etc. The abundance in types and styles of this database enable our interface to support a rich set of poses with varied diversity.

Because the neighboring poses within one motion clip may be too similar to be discerned, it is unnecessary to put all frames of the clip into our pose database. So we need to select some representative poses in CMU mocap database to construct our pose database. Denoting a character pose at frame i by $p_i = \{q_0^i, q_1^i, ..., q_{K-1}^i\}$, where q_k^i is the global position of the k-th joint represented by three-component vector and K is the total joint number, the difference between two poses p_i and p_j can be defined using the method in [KGP02] as following:

$$dist1(p_i,p_j) = \min_{\theta,x_0,z_0} \sum_{f=0}^{f=F-1} \sum_{k=0}^{k=K-1} \omega_k \cdot \left\| q_k^{i+f} - \mathrm{T}_{\theta,x_0,z_0} q_k^{j+f-F+1} \right\|^2 \quad (1)$$

The distance $dist1$ is calculated based on the point clouds formed over two windows of respective frames with user-defined length F, one bordered at the beginning by p_i and the other bordered at the end by p_j. Before calculating the minimal weighted sum of squared distances, an arbitrary rigid 2D transformation is applied to the second point cloud in order to align coordinate systems for these two point clouds. The linear transformation $\mathrm{T}_{\theta,x_0,z_0}$ rotates the target point cloud about the vertical axis by θ degrees and then translates it by (x_0,z_0). The weight ω_k is chosen to tune the weights for different joints and taper off towards the end of the window. The formula 1 has a closed-form solution, which can be referred in [KGP02]. If the $dist1(p_i,p_j)$ is larger than the predefined threshold λ, these two frames will be kept.

After character poses are selected, further processing in term of geometric and semantic levels will be put in place respectively. Firstly, in geometric level, all the character poses are translated to the origin of the world coordinate because sketch-based posing is irrelevant to the root position of the character. After that, these poses will be rotated to different optimal views, which are determined by calculating the camera path using the method in [ACOY*08]. Secondly, some typical poses with various semantic categories are selected and annotated manually to help people sketch poses in semantic level, such as walking or playing football. Based on these annotated poses, our system automatically searches similar poses in the rest of the data set and annotates them by using the distance metric defined in equation 1. Please note that we remove end-effector joints such as wrists and toes. These joints have very limited influence on the final sketched pose, and the sketching process can be simplified without them. Finally, these 3D poses are rendered to 2D im-

ages in one-pixel width black lines and white background, as shown in Figure 2.

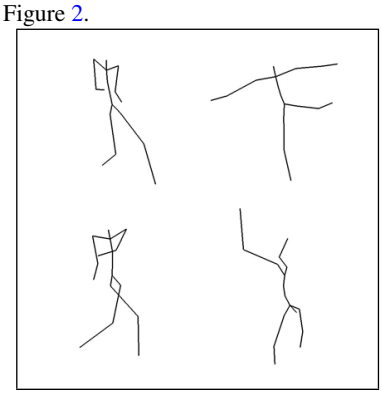


Figure 2: *2D character poses derived from 3D pose rendering under different selected optimal views.*

5 Image Descriptor

In order to provide a suggestive sketching interface, a distinctive image descriptor is used to establish the linkages between the user's sketches and 2D images of 3D poses in the database. To this end, we divide both the pose image in the database and sketching canvas into patches, and adopt an angular radial partition method introduced in [CNM05b] as their feature descriptor. Specifically, the pose image and canvas image are binary images represented by $I(\rho,\theta)$, . There is an edge at (ρ,θ) in the image when $I(\rho,\theta)=1$, otherwise, an edge does not exist. (ρ,θ) is the pixel polar coordinates on these edges. The whole image is partitioned into $M\times N$ sectors, where M is the number of radial partitions and N is the number of angular partitions. The angle between adjacent angular partitions is $\theta=2\pi/N$ and the radius difference of adjacent concentric circles is $\rho=R/M$ where R is the radius of the outermost surrounding circle of the image.

The number of pixels on the edges in each sector of I is counted to represent local patch feature $f(m,n)$, also noted as $\Gamma(k)(k=m\cdot M+n)$ in equation 2. The feature descriptor Γ of the whole image is a 2-dimensional histogram consisting of all these patch features. Figure 3 shows two descriptor examples: one for a pose image in the database and the other for partial users' sketches.

$$f(m,n)=\sum_{\rho=\frac{m\cdot R}{M}}^{\frac{(m+1)\cdot R}{M}}\sum_{\theta=\frac{n\cdot 2\pi}{N}}^{\frac{(n+1)\cdot 2\pi}{N}} I(\rho,\theta) \tag{2}$$

where $m=0,1,2,...,M-1$ and $n=0,1,2,...,N-1$.

Although this descriptor is effective and scale invariant, it has hard binning problem. This problem makes it very sensitive to stroke noises, which are unavoidable during the sketching process. In order to alleviate this problem, we introduce soft spatial binning presented in [TLC10] to replace the original binning method in our feature descriptor. Furthermore, the Gaussian weighting scheme as shown in Figure 4 is used in our system for soft binning. Considering that the pose image in the database and the query image for sketching canvas have different sizes, we adopt 16x16 Gaussian window with 8 sigma for the former, and 24x24 Gaussian window with 12 sigma for the latter.

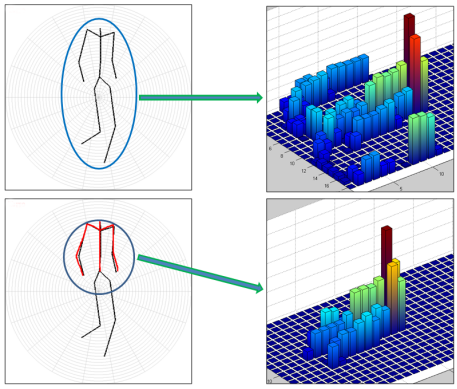


Figure 3: *Two image descriptors: the top is for pose image in the database, and the bottom is for user's sketches.*

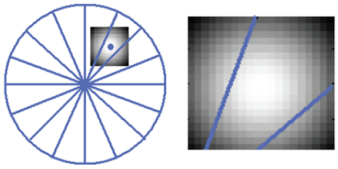


Figure 4: *Gaussian weighting scheme used for soft spatial binning. (a) weighting function for each pixel (b) contribution of a pixel to a spatial bin is proportional to the area of the weight map.*

We use the correlation coefficient, noted as $dist2$, to measure the difference between two image descriptors Γ_i and Γ_j as following:

$$dist2(\Gamma_i,\Gamma_j)=\frac{\sum_k(\Gamma_i(k)-\bar{\Gamma}_i)(\Gamma_j(k)-\bar{\Gamma}_j)}{\sqrt{\sum_k(\Gamma_i(k)-\bar{\Gamma}_i)^2\sum_k(\Gamma_j(k)-\bar{\Gamma}_j)^2}} \tag{3}$$

where

$$\bar{\Gamma}_i=\frac{1}{N}\sum_k\Gamma_i(k) \tag{4}$$

$dist2=1.0$ indicates these two descriptors are very similar, while $dist2=0.0$ means they are completely different.

6 Pose retrieval based on partial matching

Our sketching process consists of 7 strokes in fixed-order, one for trunk, two for shoulders and four for limbs as shown in Figure 5. Compared with the constraints, using fixed-order sketching has two more obvious advantages: (1) higher query efficiency. For free-hand sketching, each stroke needs to be compared with all other body parts in the database. After that, it can be determined to be which body part. That means it need much more comparisons than fixed-order method. (2) better user experience. With fixed sketching order, our system can eliminate lots of ambiguity. For example, the straight down arms can be easily mistaken as torso, which will confuse the user. More details can be referred in

experiments. When users start the sketching process, he/she can choose the type of character poses first. After that, our system will select a certain quantity of that type randomly and render their torsos on the canvas to indicate the area where the user should draw on. While the sketching evolves, our system will search the pose image database using the sketched lines before next guiding hints are updated on the canvas. The underlying query sketched image is generated by connecting those points into lines where the mouse moves with one pixel-width black pixel. To have better visual effect, we give different colors for different body parts, as shown in accompanying video demo.

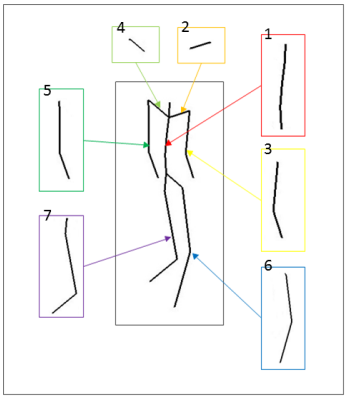


Figure 5: *7 fixed-order strokes in our sketching system, one for trunk, two for shoulders and four for limbs. The digital number shows the sketching order.*

Our feature descriptor is a global feature as defined in Section 5. Therefor, before it can support partial matching, we have to divide human body into seven separate parts as shown in Figure 5 and project these body parts onto different images to construct body parts image database. When users complete the i-th stroke, our system will compute its descriptor $\Gamma^{s(i)}$ and the corresponding image descriptor $\Gamma_j^{d(i)}$ (j is pose index), which is derived from the same body part image in the database. The distance between these two image descriptors will be computed and accumulated into historical values. After we repeat this procedure to each body part, we obtain the final distance between current sketched strokes and jth pose image, as shown in formula 5.

$$D(i,j) = \sum_{k=0}^{i} \omega(k) * dist2(\Gamma^{s(k)}, \Gamma_j^{d(k)}) \tag{5}$$

where $\omega(k)$ is an important weight parameter for tuning the contribution of i-th stroke to the final result. If the parameter is not set properly, the retrieved result will prefer to ceratin body parts such as legs more than the others. These parameters are determined by the characteristic of our image descriptor. Since seven stokes are required to create a pose, we need the same number of weight parameters in our system. In order to determine optimal values of these weights, we manually collect 18 character pose images from different motion types (2 from each type) and ask users to copy them by sketching. With this training data, we can obtain the following linear system of equations (Γ^{ts} is the descriptor of sketching strokes and Γ^{td} for selected pose images), which can be solved by the method of least squares.

$$\begin{cases} \omega(0) * dist2(\Gamma_0^{ts(0)}, \Gamma_0^{td(0)}) + \ldots + \omega(6) * dist2(\Gamma_0^{ts(6)}, \Gamma_0^{td(6)}) = 1.0 \\ \omega(0) * dist2(\Gamma_1^{ts(0)}, \Gamma_1^{td(0)}) + \ldots + \omega(6) * dist2(\Gamma_1^{ts(6)}, \Gamma_1^{td(6)}) = 1.0 \\ \cdots \\ \omega(0) * dist2(\Gamma_{17}^{ts(0)}, \Gamma_{17}^{td(0)}) + \ldots + \omega(6) * dist2(\Gamma_{17}^{ts(6)}, \Gamma_{17}^{td(6)}) = 1.0 \end{cases} \tag{6}$$

Obviously, if a brute force linear search is used, the computation cost will be too large to meet the requirement of the real-time interaction. So certain fast query method must be involved. Locality-Sensitive Hashing (LSH) is an algorithm for solving the approximate or exact Near Neighbor Search in high dimensional spaces, which is efficient to deal with our non-binary image descriptor. So we introduce local sensitive hashing method [AI08] and combine it with semantic information to solve our indexing problem. The idea of locality-sensitive hashing is to construct a family of hash functions $h \in H : R^d \rightarrow U$, for any points p,q, if $dist2(p,q) \leq R$, then the probability $\Pr[h(p) = h(q)] \geq p_1$, otherwise, if $dist2(p,q) > cR$, $\Pr[h(p) = h(q)] \leq p_2$. c is the scale factor. Especially, p_1 must be larger than p_2. In our system, the Gaussian distribution and uniform distribution are applied to generate the family of hash functions as equation 7.

$$h_{g,u}(p) = \left\lfloor \frac{g \cdot p + u}{U} \right\rfloor \tag{7}$$

where p is a one-dimensional vector formed by the image descriptor, g is a same dimensional vector with entries chosen independently from gaussian distribution and u is a real number chosen uniformly from the range $[0, U]$. We set the value of U to 4.0, which is recommended by [DIIM04].

The probability that two descriptors collide, under a hash function drawn uniformly at random from the hash family, can be computed theoretically with equation 8.

$$\begin{aligned} p(c) &= \Pr_{g,u}[h_{g,u}(p_1) == h_{g,u}(p_2)] \\ &= \int_0^U \tfrac{1}{c} f(\tfrac{t}{c})(1 - \tfrac{t}{U}) dt \end{aligned} \tag{8}$$

where $f(t)$ denotes the probability density function of the absolute value of the gaussian distribution and c is the L_2 distance between p_1 and p_2.

To maintain a low collision probability $p(c) \leq 0.1$, we draw k independent functions from the hash function family randomly and concatenate these values to form a bucket, which is used to encode an image descriptor. In our system, 6 buckets are applied and each bucket contains 2 hash function values and these parameters are estimated automatically.

For each retrieved image, it is related with a 3D character pose. The suggestive information presented on the canvas are rendered lines coming from the top 20 nearest neighbor poses searched by drawn strokes. Because our camera is fixed, these 3D poses can be rendered to stylized lines easily. According to the distance between the user's sketches and pose images, we use intensive lines with different width to rep-

resent these pose. The wider and brighter line means it is closer to the user's sketch.

7 Sampling-based pose refinement

In addition to pose generating by searching the nearest pose with the sketching input, users can also create variable poses which are different from ones existing in the database. In order to achieve this goal, one sampling-based refinement approach is involved in our system to optimize and refine the searching results. The optimization function is shown in formula 9. $p^{'}$ is the 3D sampling pose and the function P projects the 3D pose onto 2D image. In fact, any other optimization methods, which are able to achieve this refined goal, can be used here.

$$\min \left\| \Gamma_{\mathrm{P}(p')} - \Gamma_{sketch} \right\| \tag{9}$$

Our refined approach is based on standard particle swarm optimization(PSO) [Ken10] as Algorithm 1 in Appendix. PSO is a population based on stochastic optimization technique, inspired by social behavior of bird flocking or fish schooling. The system is initialized with a population of random solutions and searches for optimazation by updating generations. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles. In our system, we can take each retrieved pose as a particle without any changes. The nearest one is a very good starting point for optimization. Moreover, we can set different angle ranges for joints and control the sampling step flexibly. In detail, for those body parts which have matched users' sketches very well, we can fix their corresponding joints during the optimization in PSO, and give more variation for those body parts, which are different from users' strokes.

Here, the input for this algorithm is composed of neighbor character poses $(p_0^{'}, p_1^{'}, ..., p_n^{'})$ and the maximum iteration times ($iterMax$), while the output is the new character pose ($p_{best}^{'}$) which matches the user's sketches best. In input, $p_i^{'}$ consist of the position of root joint and the joint angle of all joints. Please note that $p_0^{'}$ is the nearest neighbor matching current user's sketches. Line 1~6 is to initialize all these particles. The position x_i of the ith particle is initialized with character pose searched according to the user's sketch. Its initial velocity v_i is set to 0.0. The initial historical best position ℓ_i is the same with x_i. To be mentioned, these initial poses are not obtained by all the user's sketches. While the user is sketching, we keep a certain amount of character poses retrieved with each single stroke. All the poses searched by these strokes are composed together into our initial data set. We will discuss the advantages of this method in the experiment. Line 9~10 is to update the current velocity and position for each particle. At each iteration, our algorithm will update the historical best position of the particle in Line 11~13. Line 14~28 is used to update the local best position for each particle. Here, we use the ring topology structure in our PSO algorithm, in order that the algorithm will compare current particle and its two neighbors to determine the local best position. Through iterating all the local best position of particles, we can obtain the global best result.

In Figure 6, two refined poses sampled by PSO algorithm are shown: the yellow skeleton with red joint is the nearest neighbor result and the cyan skeleton with blue joint is the refined result. We will discuss this PSO-based refined approach in detail in experiment section.

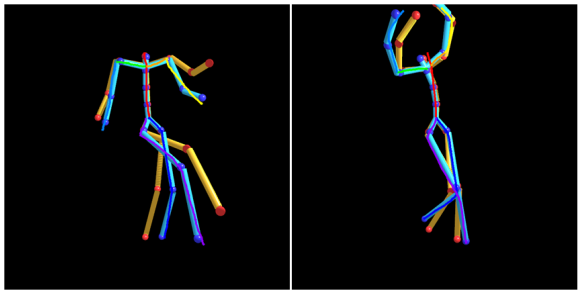


Figure 6: *The sampling results based PSO.*

8 Experiment

We collect 9 motion categories with about 100,000 different poses totally from the CMU mocap database. For each pose, we project it onto 8 images including 1 whole body image and 7 body part images under calculated optimal view. These images are 480x480 in terms of resolution. In order to get better user experience, the resolution of our sketching canvas is set to 720x720. To save the storage for our sparse descriptor, the compression method in [BR09] is used. All our experiments run on a PC with Intel i7 CPU with 3.5GHz and 8GB RAM.

Two metrics introduced in [SB10] are used in evaluation of our image descriptor. The first one, Mean of Query Rank (MQR), is the average of all query ranks is computed. The second one,recall ratio R_n, shows the ratio of retrieved target images in the best n candidates. 1000 random example images with noises are selected to perform queries from different resolution databases. The results can be seen in Table 1. The experiment shows that the query image can be found at the top 5 nearest neighbors definitely. And if the resolution of our descriptor is set to 18x30, the result will be best.

Resolution	R_1	R_5	R_{10}	R_{20}	R_{30}	MQR
9x10	98.8	100	100	100	100	6
12x20	98.2	100	100	100	100	10
18x30	99.2	100	100	100	100	4
36x40	98.8	100	100	100	100	7

Table 1: *R_n and MQR for different resolutions.*

At runtime, the image descriptors, hash family functions, bucket items and 3D mocap pose cost about 2.8GB memory in total. During the sketching process, it takes about 20ms to extract the image descriptor from the canvas and about 15ms to encode this query descriptor. If the brute force linear approach is applied, the search time will be 2.5~3s, even more for larger database. Obviously, this will decrease users' experience of an interactive system. For LSH method, it only needs 0.5~1s to have query response, which can fully satisfy the user's requirement. In Table 2, we compare the time of

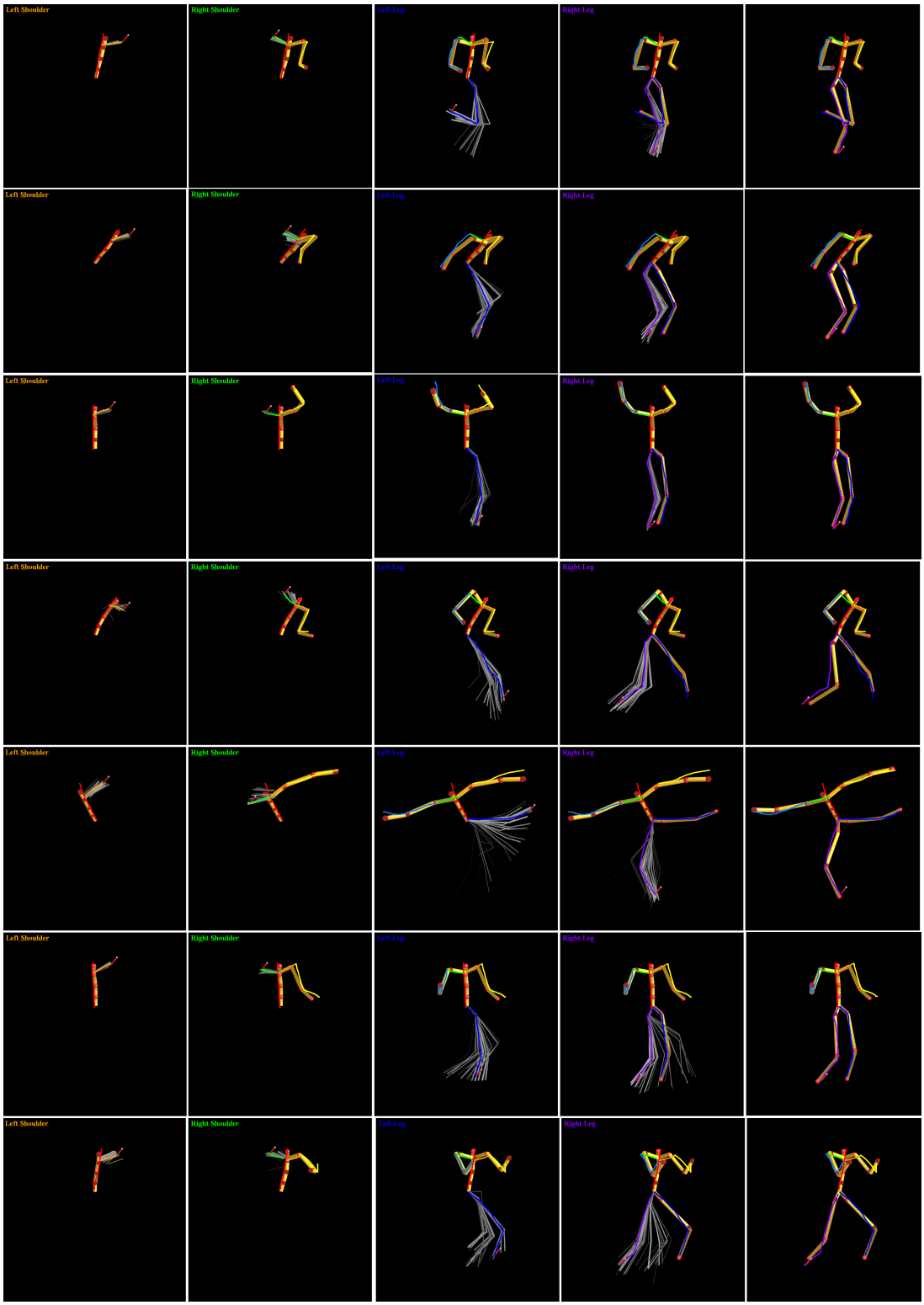


Figure 7: *Some sketching results by users: from top to bottom, they are poses about running, jumping, basketball playing, boxing, dancing, football playing and martial art.*

query response between fixed-order sketching and free-hand sketching. For free-hand sketching, it needs to be compared with all other body parts in the database to determine which body part it is. Through analyzing the time in Table 2, we find that the time used by free-hand is nearly about 20 more times than fixed-order, which conform to the general situation. To measure the sketching efficiency, we compare the results generated by our system with that obtained using 3ds max by an artist. The details can be referred in supplemental video. Figure 7 shows some users' sketching results.

	walking	jumping	basketball	dancing
free-hand	0.575	0.592	0.612	0.608
fixed-order	10.635	11.248	12.240	12.768

Table 2: *Query time measured by second for different types of character pose. The time for each type is averaged by 20 times query.*

In fact, if we use all the drawn strokes as a whole to query the database, the result can not fit the sketches very well, except some very important body parts, such as trunk or legs. That means, with global query strategy, we cannot well balance the different body part sketches, because of the image descriptor adopted in our method. However, our partial matching strategy can address this problem and obtain good result that can efficiently fit all body parts as Figure 8 shows.

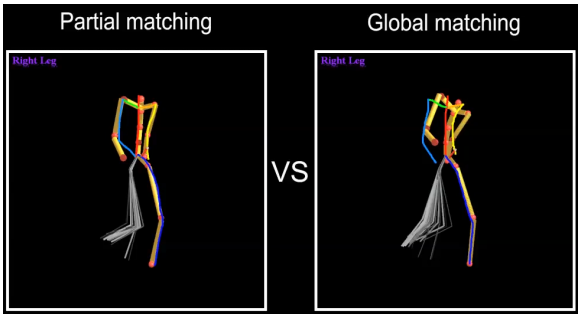


Figure 8: *The results of partial and global matching.*

As shown in Figure 9, we obtain two bad refined results because our system can not get adequate particles that can fit sketches well by employing global query strategy. On the contrary, our sampling-based refinement process can get excellent results using the well-fitting retrieved poses under partial matching strategy.

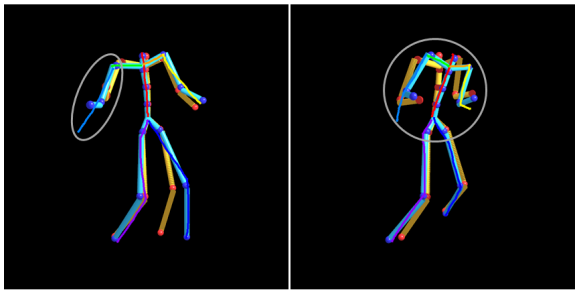


Figure 9: *The bad results generated by PSO method with insufficient particles.*

In section 7, we present a PSO-based pose refinement method. The PSO-based method can approach the optimal solution after its continuous iterations due to its good convergence property. We record the sketching process of one testing user to draw a new walking pose which does not exist in current database, and adopt PSO-based method to generate this new pose under different settings as shown in Figure 10. In our experiment, we involve different numbers of particle and iteration to refine our sketched pose. The accuracy in 10 is used to measure the matching between users' sketches with generated pose and defined by formula 3. We can see that the accuracy will be higher with more particles. This is very straightforward as long as the particles in the swarm can be distributed reasonably, it can usually find the optimal solution, even for complex motions. Even if more iterations are adopted, the accuracy is still limited when there are not enough particles as shown in Figure 9. The visual comparison can be seen in Figure 11. However, this method also has some disadvantages. Firstly, since it treats each dimension data of human pose as an independent component and there are no strong constraints between each other, it is easy to produce some false results in the condition that the motion range of joints is not specified properly. Secondly, this method only can ensure good results when the number of particles is sufficient. Once the users are dealing with complex character poses under non-parallel environment, our system will be not efficient enough with this constraint. In our experiments, we use 50 particles and the iteration times is set to 30. It takes about 7~8 minutes in average. The time cost can be further reduced to less than 1 minutes after parallelization.

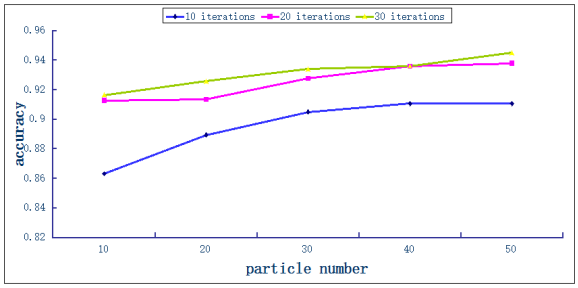
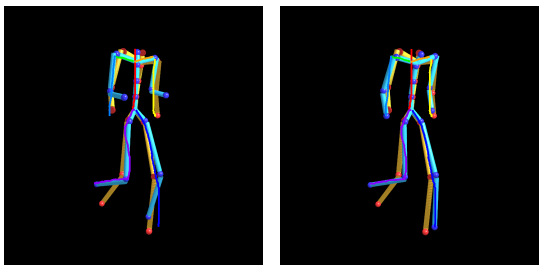


Figure 10: *The accuracy of PSO-based refinment with different numbers of particle and iteration.*



(a) 10 particles,10 iterations (b) 50 particles,30 iterations

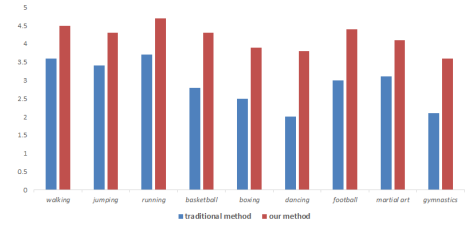
Figure 11: *The refined pose with different number of particle and iteration.*

8.1 User Studies

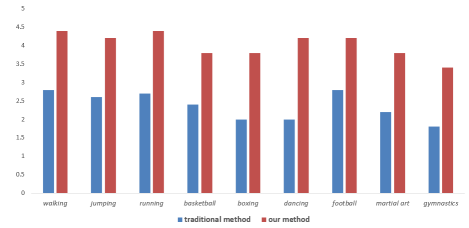
We design a user study to evaluate whether our system can help a user to sketch character poses correctly and easily. Two kinds of interfaces are provided to complete this job. The traditional method render the nearest poses in a separate window, while our interface display the suggestive information directly on the canvas.

Our users consists of 10 experienced people with computer science background (from subject1 to subject10) and 5 novice users (from subject11 to subject15), whose ages are 6-50 years old. All these subjects are required to draw 9 poses with corresponding images as reference. Another 10 people are in charge of scoring their results ranging from poorest(1) to best(5) for each category pose. The score of sketched poses, which is computed by averaging different users' scores for one same pose, can be seen in Figure 12.

From Figure 12,we can draw two conclusions: 1. Overall, using our interface, the scores are almost above 4.0 both for experienced users and novice users, which means the sketched poses can by be accepted by most of people. 2. For normal locomotion poses such as walking, running or jumping, they are more easier to be drawn and can have better scores than other complex ones. The reason is that these poses are so highly coordinated that they can be expressed better by those underling hints. This tells us for other poses, such as dancing or gymnastics, if more poses are kept, the result will be better. It is obvious that with the help of our suggestive interface, users can pose the character better using our interface than traditional method. This is because our interface solves the topology and body proportion of human skeleton efficiently, which is a thorny issue for sketch-based system. More results can be seen in the accompanying video. In addition, for those people who have no background knowledge, once they truly understand the underlying constraints expressed by those intensive lines in our interface, they can create reasonable poses quickly.



(a) subject1~subject10.



(b) subject11~subject15.

Figure 12: *The average score obtained by different users.*

9 Conclusion

We have presented a friendly sketch-based posing interface to assist non-professional people to create 3D character poses. By providing suggestive information on the sketching canvas, the user can sketch realistic 3D poses quickly with our system. We have demonstrated that our method can retrieve relevant 3D character poses by image retrieval technique in real time based on incomplete and evolving sketches drawn by the user.

From the user study experiments, we learned that our suggestive interface can really help people to create character pose quickly, especially for those who are short of background knowledge, since the user does not need to worry about the details of topology and skeleton proportion of human character any more. More important, based on retrieved poses, a new pose can be synthesized to match the user's sketches the best by using the sampling method.

The first limitation of our system is scalability. In our experiment, both the size of database and the diversity of poses are not big enough for users to freely sketch any pose of human. If we scale database greatly, the storage of descriptors and retrieved response might be difficult. We can address this problem by design and apply more compact descriptor. In addition, the sampling method used in our system is not efficient enough for interactive interface. Faster sampling approach should be explored in our future work. In the user study, some users express their comfortlessness when they are required to follow the skeletal structure and body proportions of the motion data. We also want to improve user's drawing experiences in the future.

Acknowledgements

We thank all anonymous reviewers for their constructive comments. This work is supported by the Natural Science Foundation of China (NSFC) (61332017, 61170318, 61300089, 61173124, 61322204, 61472370). Prof. Zhigeng Pan is the corresponding author for this paper.

References

[ACOY*08] Assa J., Cohen-Or D., Yeh I.-C., Lee T.-Y., et al.: Motion overview of human actions. *ACM Transactions on Graphics (TOG) 27*, 5 (2008), 115. 3

[AI08] Andoni A., Indyk P.: Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communications of the ACM 51*, 1 (JAN 2008), 117–122. 5

[BR09] Burtscher M., Ratanaworabhan P.: FPC: A High-Speed Compressor for Double-Precision Floating-Point Data. *IEEE Transactions on Computers 58* (2009), 18–31. 6

[CLAL12] Chao M.-W., Lin C.-H., Assa J., Lee T.-Y.: Human Motion Retrieval from Hand-Drawn Sketch. *IEEE Transactions on Visualization and Computer Graphics 18* (2012), 729–740. 3

[CmCT*09] Chen T., ming Cheng M., Tan P., Shamir A., min Hu S.: Sketch2Photo: internet image montage. *ACM Transactions on Graphics 28* (2009). 2

[CNM05a] Chalechale A., Naghdy G., Mertins A.: Sketch-based image matching Using Angular partitioning. *IEEE Transactions on Systems, Man, and Cybernetics 35* (2005), 28–41. 2

[CNM05b] Chalechale A., Naghdy G., Mertins A.: Sketch-based image matching using angular partitioning. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on 35*, 1 (jan. 2005), 28 – 41. 4

[CWZZ11] Cao Y., Wang C., Zhang L., Zhang L.: Edgel index for large-scale sketch-based image search. In *Computer Vision and Pattern Recognition* (2011), pp. 761–768. 2

[CYI*12] Choi M. G., Yang K., Igarashi T., Mitani J., Lee J.: Retrieval and visualization of human motion data via stick figures. *Computer Graphics Forum 31*, 7pt1 (2012), 2057–2065. 1, 2

[DAC*03] Davis J., Agrawala M., Chuang E., Popović Z., Salesin D.: A sketching interface for articulated figure animation. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), SCA '03, Eurographics Association, pp. 320–328. 2, 3

[DIIM04] Datar M., Immorlica N., Indyk P., Mirrokni V. S.: Locality-sensitive hashing scheme based on p-stable distributions. In *Symposium on Computational Geometry* (2004), pp. 253–262. 5

[DPH10] Dixon D., Prasad M., Hammond T.: icandraw: Using sketch recognition and corrective feedback to assist a user in drawing human faces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2010), CHI '10, ACM, pp. 897–906. 2

[EHA12] Eitz M., Hays J., Alexa M.: How do humans sketch objects? *ACM Trans. Graph. 31*, 4 (July 2012), 44:1–44:10. 2

[EHBA11] Eitz M., Hildebrand K., Boubekeur T., Alexa M.: Sketch-Based Image Retrieval: Benchmark and Bag-of-Features Descriptors. *IEEE Transactions on Visualization and Computer Graphics 17* (2011), 1624–1636. 2

[ERB*12] Eitz M., Richter R., Boubekeur T., Hildebrand K., Alexa M.: Sketch-based shape retrieval. *ACM Trans. Graph. 31*, 4 (July 2012), 31:1–31:10. 2

[ERH*11] Eitz M., Richter R., Hildebrand K., Boubekeur T., Alexa M.: Photosketcher: Interactive Sketch-Based Image Synthesis. *IEEE Computer Graphics and Applications 31* (2011), 56–66. 2

[FWX*13] Fan L., Wang R., Xu L., Deng J., Liu L.: Modeling by drawing with shadow guidance. *Computer Graphics Forum 32*, 7 (2013), 157–166(10). 2

[GMHP04] Grochow K., Martin S., Hertzmann A., Popović Z.: Style-based inverse kinematics. *ACM Transactions on Graphics (TOG) 23*, 3 (2004), 522–531. 1

[HFL14] Huang Z., Fu H., Lau R. W. H.: Data-driven segmentation and labeling of freehand sketches. *ACM Trans. Graph. 33*, 6 (Nov. 2014), 175:1–175:10. 2

[JSMH12] Jain E., Sheikh Y., Mahler M., Hodgins J.: Three-dimensional proxies for hand-drawn characters. *ACM Trans. Graph. 31*, 1 (Feb. 2012), 8:1–8:16. 2

[Ken10] Kennedy J.: Particle swarm optimization. In *Encyclopedia of Machine Learning*. Springer, 2010, pp. 760–766. 6

[KGP02] Kovar L., Gleicher M., Pighin F.: Motion graphs. *ACM transactions on graphics (TOG) 21*, 3 (2002), 473–482. 3

[LF08] Lee J., Funkhouser T. A.: Sketch-Based Search and Composition of 3D Models. In *Sketch Based Interfaces and Modeling* (2008), pp. 97–104. 2

[LG13] Levi Z., Gotsman C.: Artisketch: A system for articulated sketch modeling. *Computer Graphics Forum 32*, 2 (2013), 235–244. 2

[LIM*12] Lin J., Igarashi T., Mitani J., Liao M., He Y.: A sketching interface for sitting pose design in the virtual environment. *Visualization and Computer Graphics, IEEE Transactions on 18*, 11 (nov. 2012), 1979 –1991. 3

[LZC11] Lee Y. J., Zitnick C. L., Cohen M. F.: Shadowdraw: real-time user guidance for freehand drawing. *ACM Trans. Graph. 30*, 4 (July 2011), 27:1–27:10. 1, 2

[OK11] Orbay G., Kara L.: Beautification of design sketches using trainable stroke clustering and curve fitting. *Visualization and Computer Graphics, IEEE Transactions on 17*, 5 (May 2011), 694–708. 2

[RISC01] Rose III C., Sloan P., Cohen M.: Artist-Directed Inverse-Kinematics Using Radial Basis Function Interpolation. *Computer Graphics Forum 20*, 3 (2001), 239–250. 1

[SB10] Saavedra J. M., Bustos B.: An improved histogram of edge local orientations for sketch-based image retrieval. In *Proceedings of the 32nd DAGM conference on Pattern recognition* (Berlin, Heidelberg, 2010), Springer-Verlag, pp. 432–441. 6

[SI07] Shin H., Igarashi T.: Magic canvas: interactive design of a 3-D scene prototype from freehand sketches. In *Graphics Interface* (2007), pp. 63–70. 2

[SMND08] Sucontphunt T., Mo Z., Neumann U., Deng Z.: Interactive 3d facial expression posing through 2d portrait manipulation. In *Proceedings - Graphics Interface* (Windsor, ON, Canada, 2008), pp. 177 – 184. 2

[ST14] Schneider R. G., Tuytelaars T.: Sketch classification and classification-driven analysis using fisher vectors. *ACM Trans. Graph. 33*, 6 (Nov. 2014), 174:1–174:9. 2

[STDN10] Sucontphunt T., Tunwattanapong B., Deng Z., Neumann U.: Crafting 3d faces using free form portrait sketching and plausible texture inference. In *Proceedings - Graphics Interface* (Ottawa, ON, Canada, 2010), pp. 209 – 216. 2

[SXY*11] Shao T., Xu W., Yin K., Wang J., Zhou K., Guo B.: Discriminative sketch-based 3d model retrieval via robust shape matching. *Computer Graphics Forum 30*, 7 (2011), 2011–2020. 2

[TBvdP04] Thorne M., Burke D., van de Panne M.: Motion doodles: an interface for sketching character motion. *ACM Trans. Graph. 23*, 3 (Aug. 2004), 424–431. 2, 3

[TLC10] Tang F., Lim S. H., Chang N. L.: An improved local feature descriptor via soft binning. In *Image Processing, IEEE International Conference* (2010), pp. 861–864. 4

[TSB11] Thiel Y., Singh K., Balakrishnan R.: Elasticurves: Exploiting stroke dynamics and inertia for the real-time neatening of sketched 2d curves. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2011), UIST '11, ACM, pp. 383–392. 2

[WC11] Wei X. K., Chai J.: Intuitive Interactive Human-Character Posing with Millions of Example Poses. *IEEE COMPUTER GRAPHICS AND APPLICATIONS 31*, 4 (JUL-AUG 2011), 78–88. 1

[XCW14] Xing J., Chen H.-T., Wei L.-Y.: Autocomplete painting repetitions. *ACM Trans. Graph. 33*, 6 (Nov. 2014), 172:1–172:11. 2

[XTFX15] Xiao J., Tang Z., Feng Y., Xiao Z.: Sketch-based human motion retrieval via selected 2d geometric posture descriptor. *Signal Processing 113*, 0 (2015), 1 – 8. 3

A Pose refinement algorithm

input : $p_0^{'}, p_1^{'}, ..., p_n^{'}, iterMax$
output: $p_{best}^{'}$

for $i \leftarrow 0$ **to** n **do**
  $x_i \leftarrow p_i^{'}$;
  $\ell_i \leftarrow p_i^{'}$;
  $J_i \leftarrow p_0^{'}$;
  $v_i \leftarrow 0.0$;
end
for $j \leftarrow 0$ **to** $iterMax$ **do**
  for $i \leftarrow 0$ **to** n **do**
    $v_i = v_i + c_1(\ell_i - x_i) + c2(J_i - x_i)$;
    $x_i = x_i + v_i$;
    if $\left\|\Gamma_{P(x_i)} - \Gamma_{sketch}\right\| < \left\|\Gamma_{P(\ell_i)} - \Gamma_{sketch}\right\|$ **then**
      $\ell_i = x_i$;
    end
    $index = \{\}$;
    if $i == 0$ **then**
      $index = \{n-1, 0, 1\}$;
    else
      if $i == n-1$ **then**
        $index = \{n-2, n-1, 0\}$;
      else
        $index = \{i-1, i, i+1\}$;
      end
    end
    for $k \leftarrow 0$ **to** 2 **do**
      if $\left\|\Gamma_{P(x_{index(k)})} - \Gamma_{sketch}\right\| < \left\|\Gamma_{P(\ell_i)} - \Gamma_{sketch}\right\|$ **then**
        $J_i = x_{index(k)}$;
      end
    end
  end
end
for $i \leftarrow 0$ **to** n **do**
  if $\left\|\Gamma_{P(J_i)} - \Gamma_{sketch}\right\| < \left\|\Gamma_{P(p_{best}^{'})} - \Gamma_{sketch}\right\|$ **then**
    $p_{best}^{'} = J_i$;
  end
end

Algorithm 1: Pose refinement based on PSO