PoseShop: Human Image Database Construction and Personalized Content Synthesis

Tao Chen, Ping Tan, *Member*, *IEEE*, Li-Qian Ma, Ming-Ming Cheng, *Member*, *IEEE*, Ariel Shamir, and Shi-Min Hu, *Member*, *IEEE*

Abstract—We present PoseShop—a pipeline to construct segmented human image database with minimal manual intervention. By downloading, analyzing, and filtering massive amounts of human images from the Internet, we achieve a database which contains 400 thousands human figures that are segmented out of their background. The human figures are organized based on action semantic, clothes attributes, and indexed by the shape of their poses. They can be queried using either silhouette sketch or a skeleton to find a given pose. We demonstrate applications for this database for multiframe *personalized* content synthesis in the form of comic-strips, where the main character is the user or his/her friends. We address the two challenges of such synthesis, namely personalization and consistency over a set of frames, by introducing head swapping and clothes swapping techniques. We also demonstrate an action correlation analysis application to show the usefulness of the database for vision application.

Index Terms—Image database, image composition

1 Introduction

In the last few years, many image processing applications have utilized databases of images to assist image analysis, manipulation, and synthesis operations. The cross-domain image matching [1] has demonstrated the amazing power of large-scale image database; with the help of web images, the best views of 3D shapes can be found [2]; missing content in an image has been completed or image parts replaced for enhancement purpose using an image database [3], [4], [5], [6], [7]; even whole images have been synthesized from scratch by exploiting segmentation and labeling of a large set of images [8], [9], [10]. Many of these works have capitalized on the large and growing number of images available online.

Not surprisingly, the most common object appearing in online images, as in images in general, are *people*. Human characters are probably the most important content to utilize for image manipulation and synthesis. However, despite billions of Internet images containing various human figures, existing human image databases are limited in scale to a few thousands only [11], [12], [13]. This is mainly due to the tedious work involving the creation of such a database. The main contribution of this paper is providing a pipeline that can easily build a large-scale *human characters* image database we call *PoseShop*, with minimal manual effort. Moreover, to support future synthesis and high-level utilization of the database, the characters in PoseShop are *segmented* from their background and indexed according to various attributes. The definition of such a pipeline demands three key tasks: acquisition, segmentation, and indexing. There is a need to acquire correct images from online repositories, detect, and then segment the human figures from them. To support efficient search queries, there is also a need to define an effective indexing scheme.

PoseShop can harvest unlimited images of people performing various actions from the Internet. Initially, a text query that combines a human characteristic ("man," "woman," "boy," "girl") with some action such as "running" or "jumping," is sent to an image search engine such as Google or Flickr. Currently, we focus on actions where the full body is visible (e.g., we do not search for "woman driving"). Inspired by the work of Chen et al. [9], we filter the query results to retain only "algorithm friendly" images, where the foreground objects are well separated from the background. Next, the human character in each image is segmented based on a novel adaptive skin color detector. Further filtering of false samples is carried out by comparing the segmented silhouette to contours characteristic of the given action. Statistical analysis shows that our segmentation and filtering scheme successfully improves the imperfect results of simple segmentation and provides segmented human figures with sufficient quantity and quality. Testing PoseShop on more than three million online images produced a database containing more than 400,000

- *T. Chen, L.-Q. Ma, M.-M. Cheng, and S.-M. Hu are with the TNList, Department of Computer Science, Tsinghua University, FIT Building, Beijing 100084, China. E-mail: chent@cg.cs.tsinghua.edu.cn, j.c.joyhunter@gmail.com, chengmingvictor@gmail.com, shimin@tsinghua.edu.cn.*
- *P. Tan is with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117576. E-mail: eletp@nus.edu.sg.*
- *A. Shamir is with the Efi Arazi School of Computer Science, The Interdisciplinary Center, PO Box 167, Herzelia 46150, Israel. E-mail: arik@idc.ac.il.*

Manuscript received 17 Nov. 2011; revised 29 Apr. 2012; accepted 11 June 2012; published online 21 June 2012.
Recommended for acceptance by D. Schmalstieg.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-2011-11-0290.
Digital Object Identifier no. 10.1109/TVCG.2012.148.

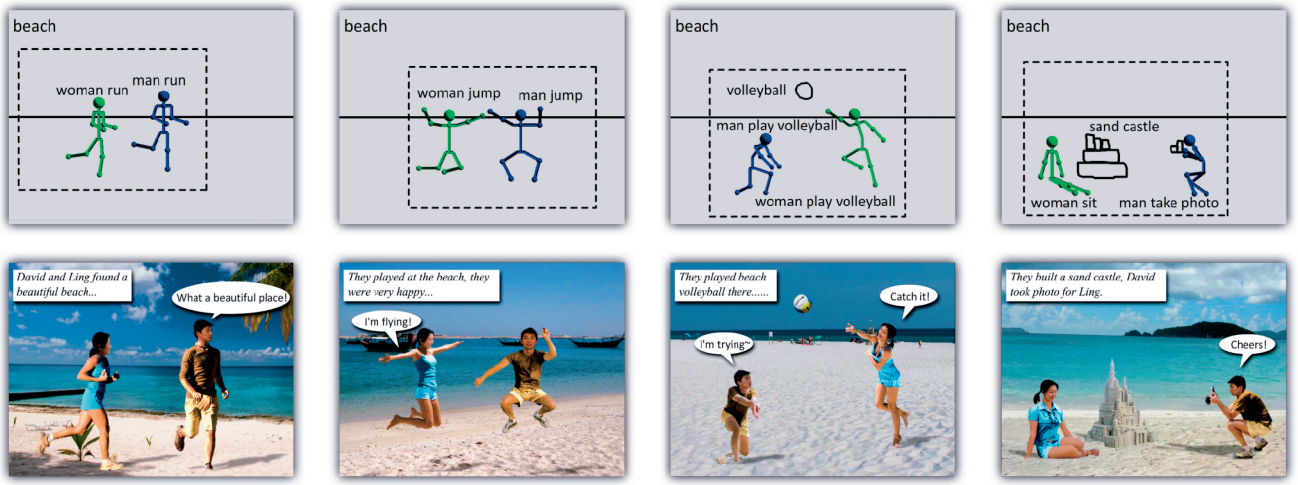


Fig. 1. Using our segmented human character database to create personalized comic-strips. The user provides sketches (first row) and some head images for each character, and the application generates a photorealistic comic-strip (second row). It finds people with similar pose, personalizes them using head swapping, maintaining consistency over different frames by replacing cloth and skin colors. Some manual refinement are applied such as adding text balloons and cropping (the dashed line rectangle in the input sketch).

segmented human characters, many more than popular image databases such as [12], [13].

To facilitate personal media synthesis and manipulation, we use multiple indexing schemes for the database. The images are indexed according to the person characteristic and action performed but also according to clothes attributes we recover such as long versus short shirt and pants, patterned versus single color. We also compute a contour signature from the segmented silhouette. Hence, the database can be queried using text attributes as well as pose attributes. We present a simple user-interface using a skeleton to define a pose and select images with similar poses (see Fig. 1).

To demonstrate the use of such database, we present two applications. The major application, which is also an important motivation of the database construction, is a system for *personal* content synthesis and manipulation in the form of personal comic-strips creation. This application is also used to enhance the database quality as user corrections of specific images (e.g., of segmentation) are fed back to the database and stored. Additionally, we show how human actions can be correlated based on visual appearance alone, using the segmented character database. This correlation is also used to enhance the query retrieval from the database itself.

1.1 Personalized Comics

Most results obtained when searching for images on the net, even for human images, are *impersonal*—they contain people whom you do not know. On the other hand, it seems that the main incentive for image manipulations would be personal: one would like to repair, change, or even synthesize images mostly of oneself, one's family and one's friends. Still, using personal images for general image synthesis is impractical as personal photo collections are too small to represent general poses and actions. Previous methods on content aware synthesis [14], [15], [8], [9] were restricted to the creation of a single image. Comic-strips can be used to compose more complex "stories" and often provide more appealing results.

In our personal comic-strip application, the user specifies a simple "story" by drawing casual sketches or positioning a skeleton figure in various poses. He or she only needs to provide some personal head images of the main characters, and the system generates photo-realistic results interactively utilizing a PoseShop database (see Fig. 1). The challenge in personalization requires changing the features of a general human character image to resemble the given person. We present a head-swapping technique and cloth color changing technique to achieve this. Moreover, synthesizing multiple frames for comics demands identity, clothing, colors, and lighting consistency across frames. We utilize our database indexing attributes to attain consistency across frames.

We compared our personalized comic-strips to similar compositions generated by an expert PhotoShop user, as well as the Sketch2Photo system. The comparison suggests that our human image database and comic-strips generation system not only save a great amount of time for the expert, but also offer nonprofessional users the ability to create interesting and vivid photo-realistic comic-strips.

2 Related Work

Many image databases have been built to train detection and recognition algorithms. Representative databases include Yale face database [16], Daimler pedestrian database [17], INRIA database for human detection [18] and HumanEva [19] for pose estimation. Typically, the image quality of these databases is limited (e.g., grayscale images of size between 200 to 300 pixels). Hence, they are not suitable for high-quality image composition tasks. There are also databases with better image quality like PASCAL VOC2006 [11], Caltech 256 [12] and LabelMe [13]. However, all of them include not more than a few thousands human images, which limit their usefulness for synthesis applications. Furthermore, these databases only provide a

bounding box around the human object and not the actual segmented characters, while manual segmentation is possible, it cannot scale up for large image numbers.

While there are many images available online, building an image database is still a nontrivial task because very few of these images are annotated. Recently, algorithms have been proposed to automatically analyze online images to build large scale databases for skies [5] and faces [4]. Amazon Mechanical Turk can also be used [20] to collect annotated images. However, such a database was missing for human characters.

Recognizing actions from a single image is a very difficult task. This problem was previously studied in [21] and [22]. In our work, we reverse this question—instead of classifying human actions in general images, we search for images of predefined human actions. A similar idea had been used in [23] to create models for classifiers of a small number of activities (five). These classifiers were then used for action recognition and annotation of video sequences. In our work, we have used more than 300 verbs as queries to build our database. Chao et al. provide an interface for retrieving human motion from hand-drawn sketches [24], we also provide a similar database querying interface.

Our content synthesis is related to content aware image synthesis works, where a series of methods [14], [15], [3], [8], [25], [9], [6], [7], [10] have been proposed with different strategies on selecting image components. Hays and Efros [3] completed missing image regions by images with similar low-level descriptors in an online image database. Diakopoulos et al. [14] and Johnson et al. [15] specified image content with text labels and composed a picture according to an automatically labeled and segmented database. Lalonde et al. [8] further used camera pose and illumination condition to choose images for high-quality composition. Eitz et al. [25] selected pictures for image composition according to sketches. Chen et al. [9] proposed the *Sketch2Photo* system that combines text labels and sketches for image selection. All these methods used generic databases with different object categories. Wang et al. [6] and Chia et al. [7] utilize image database or Internet images for image color theme enhancement and colorization respectively. Huang et al. [10] create Arcimboldo-like collage from Internet images. Further, these methods are limited to the synthesis of a single image. In comparison, we build a database specifically for human characters with semantic attributes that enables flexible and consistent multiframe comics composition.

Automatic comic-strips generation was studied previously by Kurlander et al. [26] with cartoon images. Their technique evolved into a commercial software: Microsoft Comic Chat, and later also to Google Lively. Automatic generation of comics have also been presented in [27] from interactive computer games and in [28] from action videos. However, in all these systems, a general nonpersonal icon or avatar is used to depict a character. In comparison, our personalized comic-strips can generate comics of a personal characters.

To increase the consistency across frames in a synthesized media, one can use parametric reshaping methods of human figures as in [29]. However, this does not solve the appearance consistency and is difficult to scale for huge databases of images. Xu et al. [30] capture a small database of multiview video sequences of an actor performing various basic motions. The database is then used to synthesize plausible video sequences of humans according to user-defined body motions and viewpoints. Still, the variety of the synthesis is limited by the manually captured database and it cannot generate personalized content.

3 Data Collection and Preprocessing

In this section, we describe how we collect human images and extract human characters from them. We also compute clothing and action attributes for each extracted human character. Our method builds upon existing works on human detection, skin detection, and image segmentation, but none of these existing methods are able to provide a sufficient success rate for massive online images. Our database construction relies first on an image filtering scheme and second on improvements of these methods.

3.1 Acquisition

To download high-quality images from the Internet, we first create a list of verbs that are likely to be associated with human characters such as "talk," "walk," "run," "sit," etc. Some actions involve interaction between the character and a scene object. Hence, we generate another list of verb-object phrases such as "ride bicycle," "play tennis," "eat apple," etc. In this paper, we used a list containing around 300 verbs and phrases.[1] We combine each verb/phrase with "man," "woman," "boy," or "girl," respectively. Such combinations are referred to as "keywords" hereinafter. The combinations generate over 1,200 keywords, and we search for images using these keywords online (we use Flickr as our image search engine). Each keyword typically returns around 3,000 images. Excluding repeated images, we collect around 3 million images, each with an action label. Using our database construction pipeline, this collection can be easily extended to more keywords and images.

Once images have been downloaded, we proceed to detect and segment the human characters from the images. The results of online image queries are often very noisy, they contain both false positive images and complex images that are difficult to handle. We perform various filtering stages alongside the human character extraction to achieve better results. First, we discard complex images which are likely to fail when processed by detection and segmentation methods. Later, we apply contour consistency filtering that identifies a set of key poses for each action. Accurate segmentation of human characters is another challenge, since human characters usually have complicated appearances. Fortunately, human detection is well studied in computer vision, and automatic algorithms can provide an initial guess for accurate graph-cut-based foreground extraction method. In our experiments, we have found that characters wearing shorts and pants frequently "lose" their limbs or heads since these regions are elongated and contain different colors than the clothes. To

1. The list of verbs and phrases is provided in the supplementary file, available online.

Fig. 2. Human detection results. The red frame is the bounding boxes of detected human. The blue frame indicates detected faces. The region overlayed in green contains "definite background pixels."

fix these segmentation problems, we use a novel adaptive skin detection algorithm.

3.2 Human Detection

On each downloaded image, we apply human detection described in [31] to identify human characters. We chose this detector as it is effective in detecting humans in different poses. In addition, it detects the different semantic parts of the body such as head, body, and limbs. This method does not rely on face detection excessively, so it works well also when the face of the human character is facing away from the camera. It can fail when important human parts (e.g., the whole head) are occluded. However, for our database construction and synthesis purposes, it is better to discard such partial human images. We apply conservative parameter settings for the algorithm (threshold the probability at 0.3) to exclude ambiguous detections. Some examples of the detection results are shown in Fig. 2. We discard all images where no humans are found. From images that contain multiple human characters, we only keep the nonoverlapping ones. Typically, we detect about 4,000 distinct human characters for each keyword.

3.3 Algorithm Friendly Image Filtering

Automatic foreground object segmentation algorithm usually fails when the images contain complicated backgrounds. In [9], algorithm friendly image filtering is presented to rapidly discard images whose automatic processing is likely to give unreliable results. The image filtering contains two steps. First, a saliency region detection algorithm is applied to all the images. Then, a generic image segmentation is applied to the regions except the detected saliency regions of the image. Images producing excess segments are discarded. The remaining images are the "algorithm friendly" ones where the background is simple enough so that the foreground can be well separated. In our case, to adapt the idea, we replace the saliency region with the detected character bounding box. Then, we also apply the generic image segmentation algorithm of Felzenszwalb and Huttenlocher [32] (with parameter $k = 500$ which tends to preserve the integrity of the textured region instead of oversegmenting them). Next, we sort all images downloaded using the same keyword according to the number of segments that lie outside the character bounding box. For each keyword, we only keep 50 percent of the images with the lower segments count. After this filtering, typically 2,000 human characters are left for each keyword.

3.4 Adaptive Skin Detection

The automatic human detection often generate inaccurate bounding box. This is evident in the rightmost example of Fig. 2, where the legs are not included in the bounding box. For our segmented human extraction purposes, it is crucial to have accurate high accuracy. We employ accurate skin detection to alleviate this problem. The result of our skin detection is also used to estimate clothing attributes at a later stage.

Fig. 3. Skin detection results. For each example, from left to right, they are the original image, detected skin pixels (red) according to Jones and Rehg [33], detected skin pixels (green) according to our method.

Existing skin detection methods build appearance models for skin and nonskin pixels to detect skin regions. Here, we adopt the Gaussian Mixture Model (GMM) with 16 Gaussian components in the RGB color space [33], which was trained from 13,640 manually labeled images. Note that these models work best for Caucasian and Mongoloid skin color. We calculate the ratio of the probabilities of being skin and nonskin at every pixel. A pixel is considered as skin if this ratio is above a certain threshold T, i.e., $P(skin)/P(\neg skin) \geq T$. We have found that using a fixed threshold T for all images is unreliable in our settings. This leads to many false detections (see Fig. 3), that can effect the segmentation quality. For better skin detection, we use an adaptive skin detection algorithm that utilizes human semantic part detection results to learn specific skin and nonskin models for each human character and refine the threshold adaptively.

Our adaptive skin detection iterates between two steps until convergence. In the first step, we optimize the threshold T for a given skin color model. And in the second step, we refine the skin color model. The key idea of the first step is to maximize the amount of pixels classified as skin in an area known as skin (the face) and minimize it in an area known as nonskin (background). Though this refinement is less effective if the character is facing away the camera. Toward this end, we minimize the following function:

$$C(T) = \omega_1(1 - R_1(T))^2 + \omega_2 R_2(T)^2 + \omega_3(T - T_0)^2, \quad (1)$$

where we maximize $R_1(T)$, which is the ratio of skin pixels in the face area (the blue frame in Fig. 2), and minimize $R_2(T)$, which is the ratio of skin pixels in the background (the green regions in Fig. 2). We also do not want to deviate too much from $T_0 = 1.4$, which is the optimal threshold according to [33] that was found as a result of a training process from manually labeled images. ω_i are combination weights that are set to $0.1, 0.8$, and 0.1, respectively. For each image, we minimize this function by choosing the best value from a uniform sampling of T as follows: $T_i = 1.1 + i * 0.005,\ 0 \leq i \leq 100$.

In the second, refinement, step we adjust the Gaussian weights of the color model according to the detected skin

Fig. 4. Character segmentation results. The first row shows the automatically identified "definite human pixels" (green) and "definite background pixels" (red). The second row are results from graph-cut-based segmentation.

pixels in the face and background region. Basically, we increase (decrease) the weights of Gaussian components that are close to skin pixels detected in the face (background) region. Let $\mathbf{\Pi}^{t-1} = (\pi_1^{t-1}, ..., \pi_n^{t-1})$ be the Gaussian weights of the skin model in the $(t-1)$th iteration, the weight vector at tth iteration $\mathbf{\Pi}^t = (\pi_1^t, ..., \pi_n^t)$ is updated by

$$\mathbf{\Pi}^t = \alpha\mathbf{\Pi}_1^t - \alpha\mathbf{\Pi}_2^t + (1-2\alpha)\mathbf{\Pi}^{t-1}. \quad (2)$$

Here, $\mathbf{\Pi}_1^t = \gamma(\sum_{i\in A} p_1^i, \ldots, \sum_{i\in A} p_n^i)$, where A indicates all skin pixels detected in the face region, p_j^i is the probability that the ith pixel is generated by the jth Gaussian component. γ is the normalizing factor. $\mathbf{\Pi}_2^t$ is defined in a similar way for skin pixels in the background. In all the experiments, we set $\alpha = 0.1$.

We iterate these two steps until the detected skin region does not change or a maximum number of iteration (20) is reached. The resulting skin regions are often noisy and discontinuous. We further apply a morphological "close" operation three times on the pixel regions to connect scattered skin regions. Our adaptive skin detection generates better results than the original approach of Jones and Rehg [33] (see Fig. 3). We compare the ratio of pixels incorrectly detected as skin and nonskin on 100 images with manually marked ground truth. The false skin and false nonskin ratio for [33] are 47.78 and 46.58 percent, respectively, while our detection reduced these ratios to 28.95 and 27.51 percent on the same images. More comparison and analysis are included in our supplementary file, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2012.148.

3.5 Segmentation

Each detected human character is segmented from its background with the help of skin detection by graph-cut-based segmentation techniques [34], [35]. Hernandez et al. also use GrabCut to segment human from videos [36], but their method relies on temporal redundancy of video, which does not work on independent images. In fact, none of the existing human segmentation methods can ensure good segmentation results for all the human images. Our human segmentation algorithm depends on the human detection, skin detection, and most importantly, strict contour filtering.

A set **F** of foreground pixels are defined as "definite human pixels," and a set **B** as "definite background pixels." Then, a graph-cut algorithm is applied to extract the character boundary. **F** pixels (marked in green in Fig. 4) automatically include central pixels of each detected body part, e.g., a 5×5 region at the center of head, two 1×8 regions at the centers of left and right torso regions, and a 4×14 region at the center of the character bounding box. To help the segmentation with detected skin regions, we also apply a morphological "skeleton" operation on the skin regions, and all the skeleton pixels are also included in **F** pixels. **B** pixels (marked in red in Fig. 4) are defined by patches of 5×5 pixels coming from the two top corners of the head bounding box, the upper left corner of the left torso region and upper right corner of the right torso region, and the two lower corners of the feet region. All the human part regions are detected using the human detection method of Felzenszwalb et al. [31]. In the graph-cut-based segmentation, we expand the character bounding box by morphological dilation and apply one iteration of grab-cut [34] to this expanded region with the **F** and **B** pixels as prior labeling. Sometimes, this expanded region does not cover the whole character. We iteratively apply dilation followed by grab-cut until the segmentation boundary does not change or a maximum number of iterations (20) is reached. Some of our segmentation results are provided in Fig. 4. Note, for instance, that the human detection missed the legs of the rightmost character (see the rightmost of Fig. 2). With the help of skin region detection, our final segmentation includes both legs.

3.6 Cascade Contour Filtering

Once the characters are segmented, we can use their silhouettes for shape-based filtering. We manually choose 1-5 human characters with good segmentation as representative key-poses for each keyword. Typically, they are images of the same action from different views. For each keyword, it takes less than 30 seconds to choose the representative poses, since browsing the top 100 images is generally enough. Then, we check the consistency of the segmented contours of each character with each of the representative poses. In [9], the "shape context" feature measure [37] is suggested for contour filtering. Using this scheme, after filtering, there are still about 30 percent false positives. To achieve higher quality for our database, we employ a range of methods to check contour consistency. Since we need to process a large number of images, we apply these method in a cascading manner. The slower, more accurate, methods are only applied to images that pass the faster earlier methods tests. This scheme was initially designed to speed up the filtering, but it turns out that it also improves the filtering performance, as the filters are complementary.

We first apply a closed-form affine registration algorithm [38] to align the segmentation contour to the representative poses. The method represents the input point sets as probability density functions, and treats the registration as aligning the two distributions, which can be efficiently optimized by solving a system of polynomial equations. If the estimated affine transform contains significant shearing, change in aspect ratio or rotation, we consider it as an incorrect contour and discard the image. We rank images by these three criteria and use the average rank to keep about

1,500 images for each keyword. Next, we apply the shape context similarity criterion to rank the remaining images and keep only 1,000 images for each keyword. The shape context descriptor encodes for reference points sampled on the shape silhouette, the distribution of distances to all other points. Shape context similarities are evaluated for both the candidate shape and the flipped shape, by finding optimal matching over sampled feature points. Finally, we use hierarchical tree [39] as contour similarity measurement to rank remaining images and keep the 500 top-ranked ones. The method provides a hierarchical representation for the contours that captures shape information at multiple levels of resolution, which leads to richer geometric models and more accurate recognition results, at a higher computational complexity. At each step, if there are multiple representative poses, the best consistency images from all poses are used for ranking.

3.7 Attributes Estimation

We define three binary clothes attributes, namely "long sleeves" versus "T-shirts" (for shirts), "long pants" versus "shorts" (for pants) and "single color" versus "patterned" (fpr texture). The first two attributes are detected according to the ratio of skin pixels in upper and lower bodies, respectively. These three attributes are also specifically designed for the comic-strips synthesis application. We exclude the head region from the upper body before evaluating the ratio of skin pixels to all pixels. We use a threshold on the ratio to decide if shirt and pants are long or short. We use 0.03 for shirts and 0.07 for pants according to the results on 1,000 manually annotated images. To separate shirts from pants, we divide the segmented human figure horizontally at the middle, and refine the boundary using graph-cut. Note that this simple method cannot handle lying or sitting peoples; moreover, when the shirt and pants have similar colors or a dress is worn, manual corrections to the cut may be needed when the image is used.

The texture attribute is decided by evaluating the color distribution and edge complexities of nonskin pixel regions within the upper and lower parts of the body. The texture attribute is considered "patterned" if a cubic curve cannot be fitted to these pixels in RGB space (average per pixel error is larger than 10), or if the region contains a significant amount of edges (the ratio of Canny edge pixels is larger than 0.1). Otherwise, it is considered smooth—containing a single color. To segment the clothes from skin, we shrink the skin and nonskin regions by 5 pixels and use these as initial background and foreground segmentation, respectively. We then apply a graph-cut segmentation to refine the region of upper and lower clothes. Shirt and pants are simply defined as the upper and lower nonskin pixels, respectively. A brief evaluation of the clothes attributes estimation is provided in Section 5.

4 Database Organization

We organize the image database in a tree structure according to the image segmentation contours for efficient querying. The database can be queried by a user-provided skeleton which is automatically converted to a contour. To facilitate applications, we store a separate databases for men, women, boys, and girls images. Each action attribute corresponds to a list of images associated with it. Additionally, three flag bits are used to mark corresponding clothes attributes of the images.

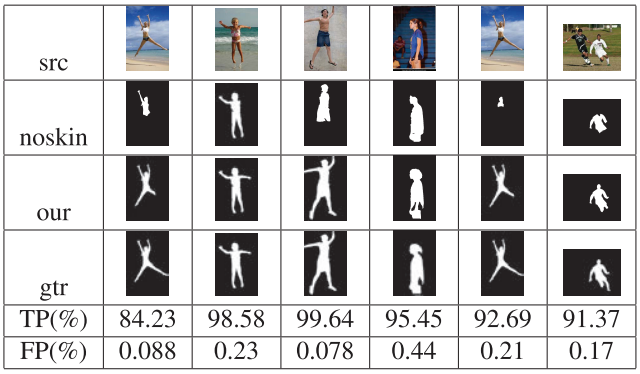
Fig. 5. The database query interface: the user can move the joints of a 2D skeleton to specify a pose, each movement triggers a query and the top-ranked segmented character image is returned. More top-ranking results can be chosen from thumbnails on the right.

Gavrila [40] has provided a sophisticated construction method of template tree by recursive partitional clustering, which can be used for hierarchical shape matching and querying. In our case, we use a simpler clustering and indexing method for efficiency consideration. Our index tree is built by iteratively clustering images by affinity propagation [41] with default parameter value ($\text{preference} = 1$). The clustering method takes as input measures of similarity between pairs of data points. Real-valued messages are then exchanged between data points until a high-quality set of exemplars and corresponding clusters gradually emerges. To measure the similarity between two segmented character images for clustering, we use shape context [37] described above. In the cascade contour filtering stage, each image is associated with a representative pose having the most similar contour. We first group representative poses (3,000 poses in total 4×300 keywords) by affinity propagation. When a group contains less than 10 representative poses, we begin to group the character images associated with each pose, and the pose naturally becomes the parent node of the associated images. In practice, our iterative grouping leads to a tree of height 6.

To test our pipeline, beginning with 3 million online pictures our final database contains 400,000 segmented human images with clothes and action attributes. This indicates our database construction method has a conversion rate of 13.3 percent. However, given the unlimited image resources on the Internet and the low manual intervention requirement, the scale of the human database that PoseShop can generate is unlimited in theory. In practice, our test database of 400,000 elements is already much larger than existing ones.

4.1 Query Interface

Our database can return a character in a specific pose; hence, the contour of a human pose can be used as a query. To simplify this task, we designed an intuitive interface where a user can manipulate the joints of a 2D skeleton to specify a pose (see Fig. 5). The pose contour is generated from the skeleton using the silhouette of a human body mesh attached to the skeleton. Our system compares the query contour with those in the index tree structure from



TP(%)	84.23	98.58	99.64	95.45	92.69	91.37
FP(%)	0.088	0.23	0.078	0.44	0.21	0.17

Fig. 6. Per-pixel segmentation quality. "src," "noskin," "our," and "gtr" indicate original online images, segmentation masks without skin detection improvement, segmentation masks in our database, and manually labeled ground-truth segmentation masks, respectively. "TP" and "FP" are true positive rates and false positive rates, respectively. More samples are provided in the supplementary file, available online.

top to bottom, and the most similar contour in the leaf node is returned. As the quality of results varies, we also list thumbnails of 10 additional alternatives which are most similar to the query contour (Fig. 5(right)). The user can then choose to manually replace the returned image. To focus the search more, or to improve the query performance, the user can also provide action labels in addition to the skeleton. In such cases, only images with the specified action labels will be examined under the tree structure. Since the tree is built by first grouping representative poses, we only need to search in branches under the representative poses of the specified action.

5 Database Evaluation

To evaluate the database quality, we choose some top-ranked sample images associated with "girl run," "woman jump," "man play football," and "boy skating." (These images are provided in the supplementary file, available online.) We manually checked the top 200 images in each of these subsets. Among these four sets, 92.0, 89.0, 81.0, and 95.5 percent images, respectively, are true positives, i.e., images where a character performs the correct action. We also manually check the clothes attributes for these images. We find 78.0, 83.0, 70, and 75.5 percent of them, respectively, have correct clothes attributes.

We also evaluated the per-pixel segmentation accuracy of the database. We uniformly sample 200 segmented human figures from the database categories associated with above four keywords and manually segment their original images. Then, we compare the per-pixel segmentations in our database to their ground truth. The average true positive rates (the ratio of detected human pixels) are 91.5, 83.9, 88.5, and 89.2 percent for the four keyword, respectively. Without our improved adaptive skin detection, these rates drop to 87.3, 76.4, 75.0, and 82.8 percent. This demonstrates the effectiveness of our skin detection method. The average false positive rates (the ratio of nonhuman pixels labeled as human) are 1.2, 1.2, 0.8, and 0.9 percent. We provide all 200 segmentations (with and without skin detection improvement) and ground truth in Fig. 6 and the supplementary file, available online. This per-pixel segmentation accuracy indicates that although the automatic segmentation of the human figures in our database is imperfect, the main structure of the character (covered by around 85 percent pixels) is preserved. This enables efficient querying using a user provided sketch or our skeleton interface. Given the potential large scale of the database, we hope it will become a valuable asset for future computer vision and graphics communities.

There is a clear tradeoff between the database quality and number of images we include per action. When more downloaded images are included, the quality of the results drop. In Fig. 7, we plot the ratio of true positive images as a function of the number of images selected for each action (blue curves). It is typically over 90 percent in the top 100 images and gradually drops to about 80 percent in the top 500 images which is the number we selected to build our database. Our results compare favorably to several alternatives. The results without applying the cascade contour filtering but using algorithm friendly filtering and segmentation (green curves), where the segmented human images are sorted only by the image search engine, produce the lowest ratio. This ratio is smaller even from the ratio of images queried directly from the Internet with no filtering, especially in the top 100 images (purple curves). However, note that many of these nonfiltered images are too complex for high-quality segmentation, or contain partial human bodies that cannot be used for our full-pose human database. Our results also compare favorably to the Sketch2Photo [9] method (red curves), which uses a user drawn contour and a text label to search images from the Internet directly.

Fig. 7 illustrates two conclusions. First, just using human detection and our improved skin detection (green curve), human segmentation results could constantly provide about 50 percent true positives with correct content and pose. The contour filtering results (illustrated by Stetch2-Photo shown with the red curve) sometimes fall even below the green curve, since it only uses a general attention detection model, not specialized for humans, to segment the main subject. Our cascade contour filtering that combines specialized human detection and pose contour filtering gives best performance. In summary, because of more

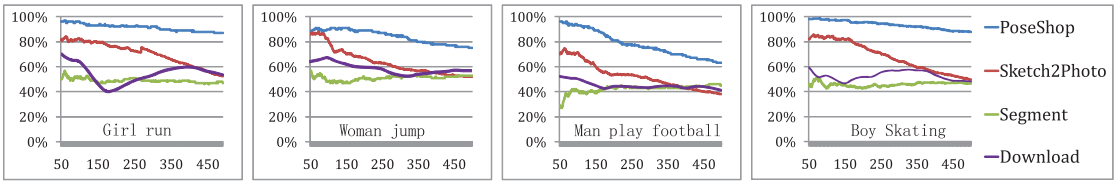


Fig. 7. Database quality versus number of images. When more images are selected for each action, the true positive ratio will drop. The true positive ratio of the proposed method is consistently higher than the one of Sketch2Photo (see text for more details).

Fig. 8. Images obtained by querying "a man jumping in T-shirts and shorts" (first row) and "a man hiking in long-sleeves and pants" (second row).

advanced detection, segmentation and filtering during database construction, the true positive ratio of PoseShop is consistently higher. Note that the search queries is also performed much faster than in a general Internet search.

Since a sequence of computationally demanding computer vision algorithms is employed in our database construction method, we provide some timing evaluation. These timings were measured for normalized images of 120k pixels on a single core of an Intel i7 CPU. The human detection and the iteratively graph-cut-based segmentation are the most time consuming steps, which take on average around 5 seconds and 18 seconds per image, respectively. The generic image segmentation [32], adaptive skin detection, and cloth attributes estimation together take about 1 second per image. However, these steps can be easily parallelized for a large image collection. The representative pose selection and cascade contour filtering and organization take around 5 minutes for each keyword. On our 16-thread configuration, the total process takes about 1 hour for each keyword. Our test database was built within two months. Considering most steps are automatic and the manual intervention for each keyword takes less than 1 minute (defining keywords and selecting representative poses), it is safe to claim that the proposed method can gradually build any size human pose database with the expanding amount of the Internet images.

For application purposes, it is more important to check the quality of queried images from the database. Fig. 8 shows some images returned by the following queries: "a man jumping in T-shirts and shorts" and "a man hiking in long sleeves and pants." Such queries take about 1 second to run in a computer with $2\times$ Quad Core i7 920 CPU and 12 G RAM. We further evaluate the query performance by the true positive ratio in the returned top 100 images. This ratio is provided for several actions in Fig. 9. On average, the clustering decrease true positives by 5 percent in the top 100 returned results. Still, most queries have high true positive ratio of above 70 percent.

6 Applications

Our character image database opens the door to many applications. There are mainly two possible types of applications: analysis and synthesis. For synthesis, our database and querying tools could be used with existing image composition systems like *Photo Clip Art* [8] or *Sketch2Photo* [9]. The additional attributes like actions and clothing type could make the compositions more controllable. For analysis, machine learning algorithms can be used to train classifiers for each action label in our database and recognize human pose for a given human silhouette.

In this paper, we demonstrate one application of each type. For synthesis application, we present an interactive system to generate personalized comic-strips. The unique challenges of this system is first to personalize general images from PoseShop database, and second, to maintain the consistency of the personal characters in all frames. We develop a series of image personalization and consistency techniques to achieve these goals. To demonstrate analysis application, we show that semantic meaning can be learn using visual representation from PoseShop database, such as the correlation of different action words.

6.1 Personalized Comic-Strips Synthesis

Motivated by *Sketch2Photo* system [9], we want to allow the generation of personalized comic-strips. The Sketch2Photo is a system that composes a realistic picture from a freehand sketch annotated with text labels. The composed picture is generated by seamlessly stitching several images searched from the Internet in agreement with the sketch and text labels. Personalized comic-strip are created from more than one image where coherency is important, and contain figures of the user or his/her friends and not just general people. The input to our application includes the comics story-board with some simple sketches and text labels, and head images of the main character(s). We use similar stages as those presented in *Sketch2Photo* system, but search for human characters in our segmented PoseShop human database instead of the Internet. The novel contributions include the personalization of the anonymous characters obtained from the database, and maintaining consistency across multiple frames.

In each frame, the user can position several skeletons to represent characters and also sketch other scene items. The user can also draw accessories for the character, and these are merged with the skeletons they are attached to. The overall shape is then used to query the character database. A textual label is provided for each character and scene item. Our system automatically searches for characters in the given pose using our database, and for suitable scene items over the internet, and then seamlessly composites them into a realistic picture. We personalize the different *anonymous* segmented human images using a head swapping technique. We also present a clothes color swapping

	dancing	fencing	gymnastics	hike	jump	kick	play football	play golf	paly tennis	run
TP (%)	81	68	71	80	80	60	72	70	74	87

Fig. 9. The true positive ratio among top 100 images of various queries.

Fig. 10. Head swap result. For each example, from left to right, they are the original image, result by automatic swapping and interactively refined result. The left two examples do not need any interaction.

method to maintain the visual consistency of characters across the comics frames. In the following, we explain these techniques and other aspects of our system.

6.1.1 Head Swap

People recognize characters by their faces, Bitouk et al. [4] designed a method to swap faces. In our system, however, we need to swap the whole head for better frame consistency because hair style is also visually important. We require the user to provide four head images of different directions with a homogenous background. For general use, the system also provides a default head library of 20 people, each with four directions. We use the orientation of the input skeleton to decide which of these four images to use.

Once the target and source head images are decided, we apply the following six steps to swap the head:

1. We first detect the face of each image according to the human detection result (face is indicated by the blue frame as in Fig. 2).
2. We estimate the shoulder (the green lines in Fig. 10) by fitting two line segments on both sides of the face according to the upper boundary of the clothes pixels, i.e., nonskin pixels within the character segment.
3. We connect the top of the two shoulder by a line (the red line in Fig. 10). The character part above this line is considered the head. Skin pixels on this line give the width of the neck. The neck is set as a rectangle whose height is 1/6 of its width (the blue rectangle in Fig. 10).
4. We then find a similarity transformation to register the two heads. We require the pasted neck to cover the upper edge of the original neck, which gives a feasible range of the transformation parameters.
5. We search for the optimal transformation within this range by exhaustively searching with a small step size, i.e., 1 pixels for translation and 1 degree for rotation (scaling is fixed as the middle value of its range). We represent each head by the points on its boundary and the Harris corners [42] within the face. The optimal transformation is the one that minimizes the set-to-set distance.[2]
6. Finally, we apply blending as proposed in [9] to paste the new head onto the target image.

2. The set to set distance between a set of points $p_i, 1 \le i \le N$ and $q_j, 1 \le j \le M$ is defined as $\sum_i \min_j d(p_i, q_j) + \sum_j \min_i d(p_i, q_j)$. $d(p_i, q_j)$ measures the distance between p_i and q_j.

Fig. 11. Clothes swapping results. In each example, the source image on the right is adjusted according to the target image on the left. The original source image before swapping is shown at the lower right corner.

6.1.2 Clothes Consistency

Swapping clothes between two character images is a challenging problem. However, if both characters wear single color clothes of the same style, clothes swapping can be reduced to color swapping. Hence, in our synthesis applications, we restrict the database queries to characters that have a "single color" attribute for clothes. Further, for consistency, we require the same type of clothes for the same character across all frames, i.e., either long or short pants and shirt. We use the clothes segmentation generated in PoseShop database and further refined it by alpha matting [43] to estimate the fractional clothes region.

Reinhard et al. [44] proposed to transfer color between two images by mapping the mean and variance of each channel of the $l\alpha\beta$ space. This method has difficulties when there are significant shading variations in clothes. We propose a different method for color swapping in the Lab color space. In the target image, we fit a cubic function to relate its a, b channels to the L channel, treating $a(L), b(L)$ as cubic functions of L. In the source image, we first map its L channel to the target L channel by a linear transformation such that their mean and variance are the same. However, when dark clothes are changed to bright, this can create significant quantization error. If the mean differences is more than 80 gray-levels, we perform median filtering. Next, we apply the cubic functions $a(L), b(L)$ to the target image to update the a, b channels from the source image. To maintain color consistency among multiple frames, we choose as target the clothes color from the frame that is closest to the average color of all frames. We do not use the average color itself, as large variations of clothes colors in different images often leads to poor average colors. Some results of clothes color swapping are shown in Fig. 11.

6.1.3 Skin and Shape Consistency

Since we have the skin detection of each human character, we can also adjust the skin color for consistency. We use the same method as clothes color swap, but here the target skin color distribution is computed as the average of all instances of the same character in all frames, and then applied to all of them. This minimizes the artifacts of large differences between chosen characters. Body shapes are first adjusted by simple uniform scaling. Next, we use the input skeletons to calculate the aspect ratios of the character and adjust the human body aspect ratio accordingly. We constrain the change to be no more than 10 percent.

6.1.4 Scene Item Consistency

Users may require the background or scene items, other than characters, to remain consistent over different frames.

Our system provides two ways to handle this. Either the user chooses to fix the items over all (or some) of the comics frames, or to constrain them to have similar colors. When the second option is chosen, the image selection will require the image component histogram to be consistent over different frames.

6.1.5 Combination Optimization

Since the image is composed of several characters and items, it is beneficial to find the best combination of scene items and background. When optimizing the combination of searched image components per frame, we apply additional constraints to maintain character consistency across frames. First, we require the same character to have the same clothes attributes across all frames such as "long shirt + long pants" or "T-shirts + long pants." Next, the components combination is selected by optimizing the following cost:

$$\arg\min_{B_i,I_i,S_i} \sum_i C_i(B_i,I_i,S_i) + \lambda \sum_{k\neq l} \|I_k - I_l\|^2.$$

Here, i is the frame index. Assuming, for simplicity of notation, that each frame contains one character and one scene item, then B_i, I_i, and S_i are the candidate background, human character, and scene item components in the ith frame, respectively. The first term in the sum, $C_i(B_i, I_i, S_i)$ measures the local cost of the ith frame using the hybrid blending cost of [9], which combines the Gabor feature difference, the summed pixelwise difference of the UV color components between source and target images, as well as the matting feasibility measure of source images. The second term $\|I_k - I_l\|^2$ measures the consistency between the pairs of frames k and l of the average skin and clothes color of the character using L2 distance in RGB space. We typically use a small λ (0.2 in our implementation) since consistency is usually maintained by our swapping techniques. We use 100 candidates for each character I_i and scene item S_i, and 20 candidates for each background image B_i. In our examples, we generated 4 frames in a comic-strip and we exhaustively search for the optimal combination among all possible combinations.

6.1.6 Interactive Refinement

Automatic segmentation, swapping of head, and consistency of clothes, skin, and body shapes are very challenging. Our automatic method often generates imperfect results. Hence, we allow the user to interactively improve the automatic results. To facilitate the interaction, our system generates four automatic compositions of each frame and the user can choose between them and follow with interactive refinement. He/she can replace incorrect image components and improve image segmentation. For instance, the automatical head swapping could have unnatural head orientation and neck connection due to incorrect segmentations. The clothes and skin color swapping also depend on precise segmentation of clothes and skin regions. These segmentation often need some interactive refinement by the user. The user can also manually adjust the orientation, scaling, and position of the replaced head image (see Fig. 10). Our body shape adjustment cannot work well for characters that are too fat or too thin. In this case, the user can manually adjust the aspect ratio or just change to a different human character. Shadows can be added by pasting a flattened and black character image at a user-specified position manually. Further, both conversation balloons and hallos for human characters can be created manually similar to conventional comic-strips. The user can also zoom in or crop a region from the automatic compositions to focus more on the characters and not on their whole body.

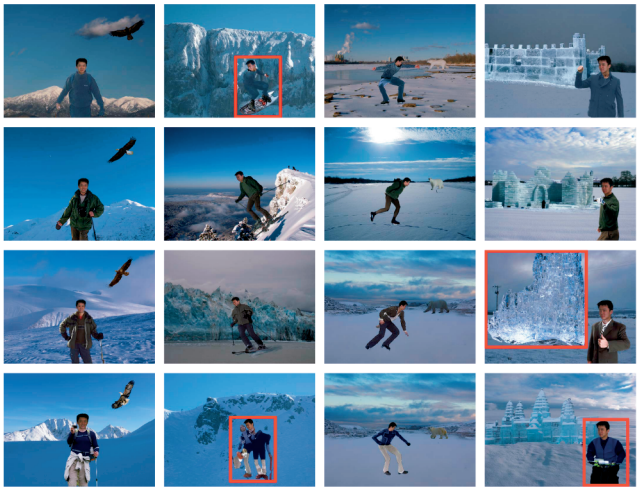


Fig. 12. Four automatic compositions created for "ski story." Red frames indicate incorrect scene items.

When a user refines a segmentation of a specific image from the PoseShop database, the results are recorded and stored back in the database itself. Hence, personal comics creation and similar types of applications that utilize the database, can also be used to continually enhance the database quality.

6.1.7 Example Results

We composed several comic-strips to demonstrate the application of our PoseShop database in personalized content synthesis. In each comic-strip, we sketched four frames containing one or two human characters. We obtain the candidate character images from our database in a few minutes. In contrast, the query and filtering of other scene items and backgrounds from the Internet can take a significant amount of time (10-20 minutes for each scene item). Next, the combination optimization takes around 5 minutes, and outputs four automatic compositions. In these automatic compositions, there are typically two to three incorrect scene items or human characters in all four frames (see Fig. 12). The interactive refinement begins from these results and usually takes around 10 minutes for each comic-strip (please refer to the supplementary video, available online, for more details). The most time-consuming interaction is segmentation refinement of the characters and scene items, which usually takes 4 to 5 minutes, i.e., 1 minute for each frame. Other manual refinements include adjusting heads, pointing shadow directions, adding talking balloons and cropping.

Fig. 1 shows a beach vacation comics generated with our system. The first row shows the input sketches where the

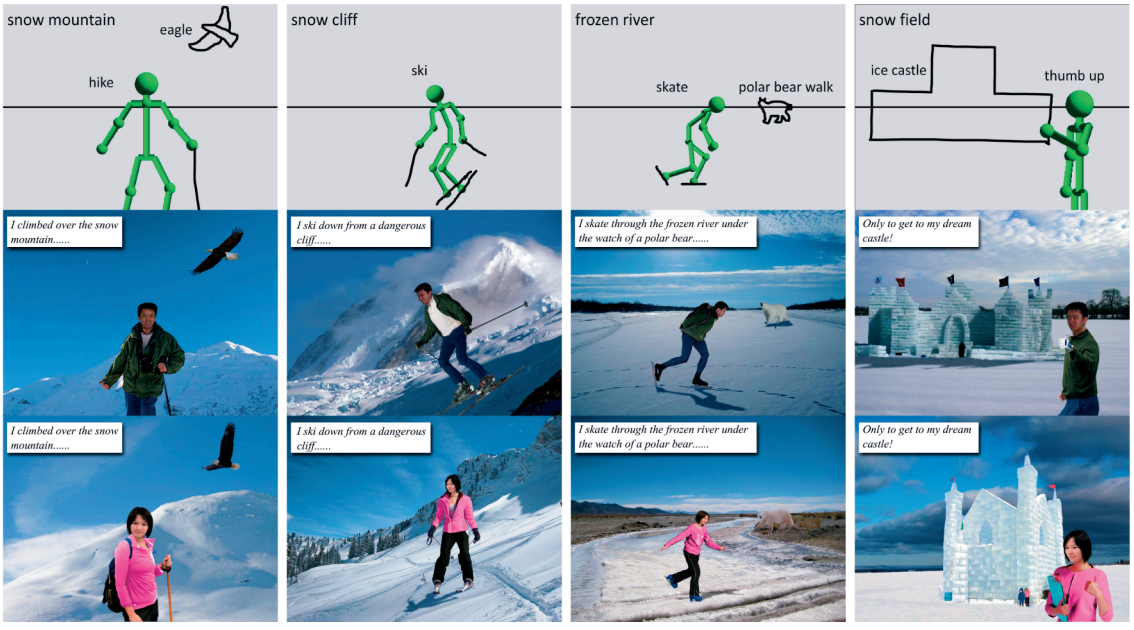


Fig. 13. Different personalized comics of the same story. These two comics are generated with the same sketch but different character labels ("man" versus "woman"). Head images are provided from a male and female users, respectively.

same character is shown in the same color over different frames. The second row shows the generated comic-strip (the top four automatic compositions of this example are included in the supplementary video, available online). These frames closely match the semantic meaning of the sketch. Thanks to the head and clothes swapping technique, human characters appear consistent over different frames, although they are composed from images of different people. However, there are still visible illumination artifacts since our method does not estimate lighting conditions. For example, the bodies have directional lighting, while the heads have uniform illumination.

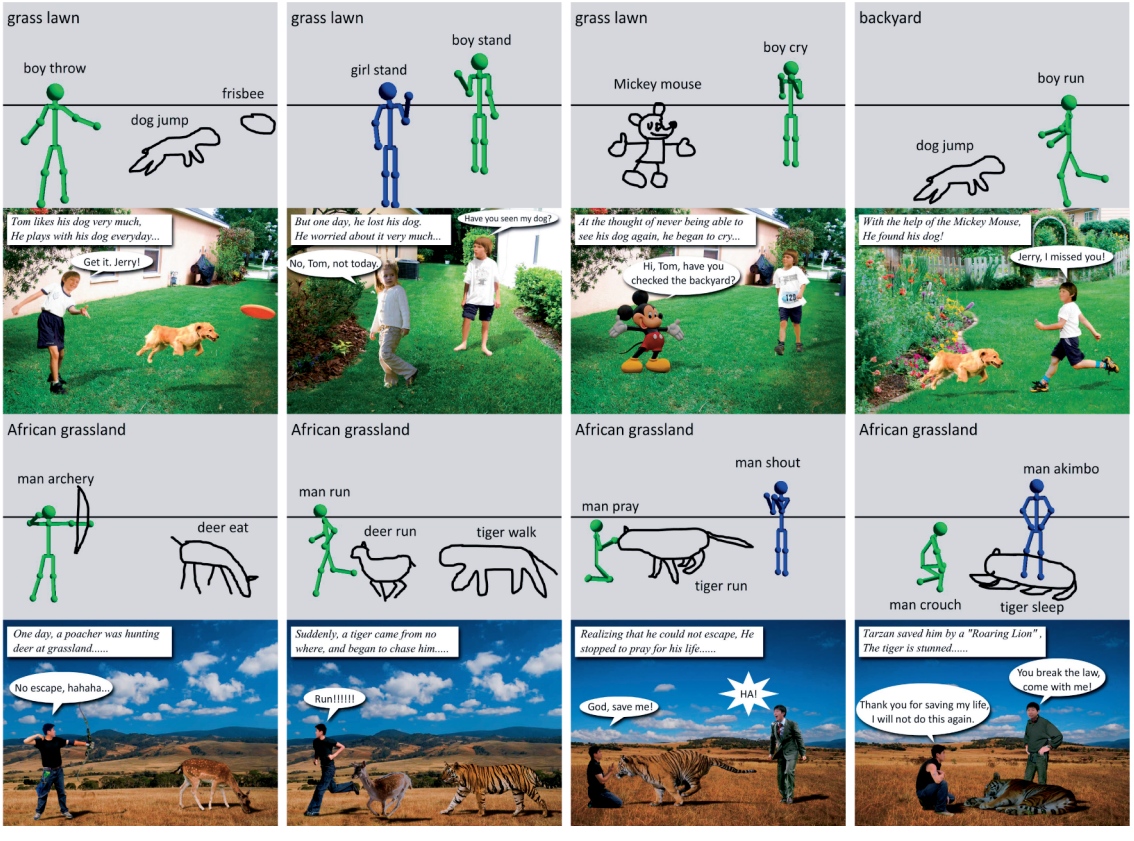


Fig. 14. Personalized comic-strips generated from our system. For each example, we show the input sketches in the first row and generated comics in the second row.

Fig. 15. Similar comic-strip compositions of Fig. 1 generated by a PhotoShop expert user(first row), and by the Sketch2Photo system (second row). More details are provided in the supplementary file, available online.

Fig. 13 provides two different personalized comics of the same story. These two comic-strips are generated from the same sketch with different text labels: "man" versus "woman," and male and female heads, respectively. This example shows that our system can achieve personalized stylization. The top four automatic compositions of the male version are also provided at Fig. 12. One of them does not have any incorrect items, and needs only manual refinement of the segmentation details. More comic-strips are provided in Fig. 14. There are still a few inconsistencies in some results, despite our swapping technique. For example, a character in the last two frames of the tiger example has inconsistent clothing, as our limited clothes types does not distinguish between suits and jackets. The image components used for these compositions are provided in the supplementary files, available online.

To evaluate our comic-strip composition system, we asked a PhotoShop expert (thousands hours experience) to generate comic-strip compositions according to the sketches in this paper. We also asked a novice to our system to generate three additional groups of comic-strips after a 30 minutes training period. These comic-strips were compared by five evaluators. According to their evaluation and time recordings, we conclude that while the composition qualities are comparable, our system saves a significant amount of manual intervention time. The expert had a difficult time finding good image candidates (2-4 hours for each comic-strip), as well as to achieve desirable composition (1-2 hours for each comic-strip) by rotoscoping and blending. In contrast, it took 10-20 minutes of user interaction using our system. We also compare our results to a group of compositions generated by the Sketch2Photo system. The inconsistencies of characters in such compositions are evident and the lack of shadows further reduces the sense of realism. Additional results and details of this evaluation are provided in Fig. 15 and the supplementary file, available online.

6.2 Action Correlation Analysis

In our database, each character image is associated with the action keyword used to download it. On the other hand, different actions might share similar poses. Our goal is to find semantic connections between different action keywords utilizing the PoseShop database. We identify correlated *actions* based on the human poses in these actions. The basic idea is if action **A** and **B** are correlated, then images associated with these two actions should be similar.

To measure the correlation of action **A** and **B**, we first utilize the ranking of images of action **A** using our cascade contour filtering. We denote the image ranked at position 100 as the 100-benchmark image, and calculate its similarity to the representative poses of **A** as δ. Then, we apply cascade contour filtering to all images of action **B** by using the representative poses of **A** as matching shapes. If more than 50 images can be found from the images of **B** that have a better similarity measure than δ, we consider action **B** to be related to **A**. Next, we repeat the process above but swap **A** and **B** to measure if **A** is related to **B**. Note that this type of correlation could be nonsymmetric. For example, *play football* is related to *kick*, but most *kick* images are not related to *play football*.

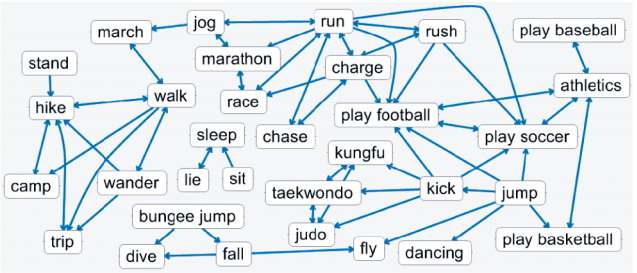


Fig. 16. The correlation graph of different actions. We use an arrow **A**→ **B** to indicate action **B** is related to action **A** (according to the definition in Section 6.2). Note that this correlation is not symmetric, i.e., **A**→ **B** does not imply **B**→ **A**.

Fig. 16 shows an example of the correlation graph of 32 selected actions. According to the graph, most correlations we found are consistent with the semantic meaning of the action keywords. It took 140 minutes to calculate this graph on a 2 × Quad Core i7 920 CPU and 12 G RAM computer. The construction of this correlation graph is an interesting simulation of human recognition process. It shows how semantic meaning—though in this case, it is merely the correlation between action words—can be learned only from the visual information associated with them.

We use action correlation analysis to improve the results of querying the PoseShop database. If **B** is found to be related to **A**, some images of **B** can also be associated with the action label of **A**. For a given query of action **A**, we compare also images of action **B** with the representative poses of **A**. An image with similarity that is ranked higher than the 200-benchmark image of **A** will also be labeled with action label of **A**. Now, each image could be associated with multiple action labels in the database. For example, the same image might be associated with the actions *play football*, *play soccer*, and *kick*. This promotes the higher versatility and ability to find good matches in the database.

7 Limitations

Our database has a number of limitations. First, a small amount of false samples still exist in the database, i.e., an image labeled as "man walking" may not have a walking man in it. Second, the segmentation is often imprecise. Our human segmentation result may be affected by false positive skin areas and incorrect initialization pixels. As a result, in our content synthesis application, user interaction is mostly necessary to refine the segmentation. However, once the segmentation is refined, it can be carried back and stored in the database. We plan to make the database and content synthesis systems publicly available online so that

refined segmentations can be collected when users improve their compositions. The segmentation could also be improved by applying other state-of-the-art human segmentation methods such as [45]. Third, our clothes attributes are rather limited at this stage. Other types of clothes such as dresses or bathing suits are not supported by our system and clothes type detection could be an interesting future research. Typically, female characters are more difficult to handle since they often have patterned clothes and different hair styles. The comic-strips synthesis also discards textured clothing completely, which also limits its variability. Fourth, the swapping techniques for personalization and consistency are limited. Head swapping often needs manual correction at the neck connection. Skin and clothes color swapping are restricted by the segmentation accuracy. Lastly, since we do not estimate illumination conditions, our synthesized images might have inconsistent lighting effects. This problem can be solved by adopting recent illumination estimation method or environment-sensitive image cloning such as [46] and [47].

8 Conclusion

In this paper, we presented a pipeline to construct a large segmented human image database. In this pipeline, we first download and analyze millions of online images automatically. Human characters are detected and segmented from their background, and then annotated by their action and appearance attributes (clothes, skin). We built a database of these images by indexing using the segmentation contours. The database can be queried using these attributes or with a skeleton that defines the pose and shape. We provided quantitative evaluation of the database by measuring labeling accuracy and per-pixel segmentation accuracy. We further presented techniques for head and clothes swapping for image personalization and consistency. With our database and these swapping techniques, we presented an interactive system to generate personalized comic-strips, where the user can create comics of himself or his/her friends.

We plan to make the database and the query library publicly available to assist related works. In the future, we plan to utilize our human database for various other image analysis and synthesis tasks. It can be used to assist machine learning techniques to classify actions, clothes, and poses. The database itself could be extended to include more types of actions and other types of clothes, and further semantic analysis could be applied. The database construction pipeline can also be adopted to generate image database of other objects, such as animals and vehicles, with the aid of corresponding detection algorithms. Several parts in the comic-strips application could also be improved such as automatic positioning and recomposition, better segmentation and possible enhancement with automatic text and story understanding.

Acknowledgments

The authors would like to thank reviewers for helpful suggestions. This work was supported by the National Basic Research Project of China (Project Number 2011CB302205), the Natural Science Foundation of China (Project Number 61120106007 and 61103079) and the National High Technology Research & Development Program of China (Project Number 2012AA011802). Ping Tan was partially supported by the Singapore ASTAR PSF project R-263-000-698-305. This research was also partly supported by the Israel Science Foundation (grant no. 324/11). Shi-Min Hu is the corresponding author.

References

[1] A. Shrivastava, T. Malisiewicz, A. Gupta, and A.A. Efros, "Data-Driven Visual Similarity for Cross-Domain Image Matching," *ACM Trans. Graphics,* vol. 30, no. 6, pp. 154:1-154:10, Dec. 2011.

[2] H. Liu, L. Zhang, and H. Huang, "Web-Image Driven Best Views of 3D Shapes," *Visual Computer,* vol. 28, no. 3, pp. 279-287, Mar. 2012.

[3] J.H. Hays and A.A. Efros, "Scene Completion Using Millions of Photographs," *ACM Trans. Graphics,* vol. 26, no. 3, pp. 4:1-4:7, 2007.

[4] D. Bitouk, N. Kumar, S. Dhillon, P. Belhumeur, and S.K. Nayar, "Face Swapping: Automatically Replacing Faces in Photographs," *ACM Trans. Graphics,* vol. 27, no. 3, pp. 39:1-39:8, 2008.

[5] L. Tao, L. Yuan, and J. Sun, "Skyfinder: Attribute-Based Sky Image Search," *ACM Trans. Graphics,* vol. 28, no. 3, pp. 68:1-68:5, 2009.

[6] B. Wang, Y. Yu, T.-T. Wong, C. Chen, and Y.-Q. Xu, "Data-Driven Image Color Theme Enhancement," *ACM Trans. Graphics,* vol. 29, no. 6, pp. 146:1-146:10, Dec. 2010.

[7] A.Y.-S. Chia, S. Zhuo, R.K. Gupta, Y.-W. Tai, S.-Y. Cho, P. Tan, and S. Lin, "Semantic Colorization with Internet Images," *ACM Trans. Graphics,* vol. 30, no. 6, pp. 156:1-156:8, Dec. 2011.

[8] J.-F. Lalonde, D. Hoiem, A.A. Efros, C. Rother, J. Winn, and A. Criminisi, "Photo Clip Art," *ACM Trans. Graphics,* vol. 26, no. 3, pp. 3:1-3:10, 2007.

[9] T. Chen, M.-M. Cheng, P. Tan, A. Shamir, and S.-M. Hu, "Sketch2photo: Internet Image Montage," *ACM Trans. Graphics,* vol. 28, no. 5, pp. 124:1-124:10, 2009.

[10] H. Huang, L. Zhang, and H.-C. Zhang, "Arcimboldo-Like Collage Using Internet Images," *ACM Trans. Graphics,* vol. 30, no. 6, pp. 155:1-155:8, Dec. 2011.

[11] M. Everingham, A. Zisserman, C.K.I. Williams, and L. Van Gool, "The PASCAL Visual Object Classes Challenge," *Int'l J. Computer Vision,* vol. 88, pp. 303-338, http://www.pascal-network.org/challenges/VOC/voc2006/results.pdf, 2006.

[12] G. Griffin, A. Holub, and P. Perona, "Caltech-256 Object Category Dataset," technical report, http://resolver.caltech.edu/CaltechAUTHORS:CNS-TR-2007-001, 2007.

[13] B.C. Russell, A. Torralba, K.P. Murphy, and W.T. Freeman, "Labelme: A Database and Web-Based Tool for Image Annotation," *Int'l J. Computer Vision,* vol. 77, nos. 1-3, pp. 157-173, 2008.

[14] N. Diakopoulos, I. Essa, and R. Jain, "Content Based Image Synthesis," *Proc. Conf. Image and Video Retrieval (CIVR),* pp. 299-307, 2004.

[15] M. Johnson, G.J. Brostow, J. Shotton, O. Arandjelović, V. Kwatra, and R. Cipolla, "Semantic Photo Synthesis," *Computer Graphics Forum,* vol. 25, no. 3, pp. 407-413, 2006.

[16] A. Georghiades, P. Belhumeur, and D. Kriegman, "From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 6, pp. 643-660, June 2001.

[17] M. Enzweiler and D.M. Gavrila, "Monocular Pedestrian Detection: Survey and Experiments," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 31, no. 12, pp. 2179-2195, Dec. 2008.

[18] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition (CVPR),* pp. 886-893, 2005.

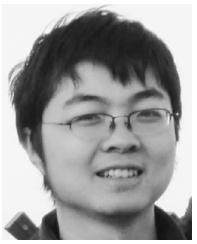
[19] L. Sigal and M.J. Black, "Humaneva: Synchronized Video and Motion Capture Dataset for Evaluation of Articulated Human Motion," technical report, 2006.

[20] A. Sorokin and D. Forsyth, "Utility Data Annotation with Amazon Mechanical Turk," *Proc. First IEEE Workshop Internet Vision at Computer Vision and Pattern Recognition,* pp. 1-8, June 2008.

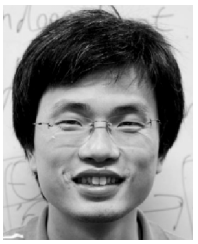
[21] J. Sullivan and S. Carlsson, "Recognizing and Tracking Human Action," *Proc. European Conf. Computer Vision (ECCV),* pp. 629-644, 2002.

[22] Y. Wang, H. Jiang, M.S. Drew, Z.-N. Li, and G. Mori, "Unsupervised Discovery of Action Classes," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition (CVPR),* pp. 1654-1661, 2006.

[23] N. Ikizler-Cinbis, R.G. Cinbis, and S. Sclaroff, "Learning Actions from the Web," *Proc. IEEE Int'l Conf. Computer Vision (ICCV),* pp. 995-1002, 2009.
[24] M.-W. Chao, C.-H. Lin, J. Assa, and T.-Y. Lee, "Human Motion Retrieval from Hand-Drawn Sketch," *IEEE Trans. Visualization and Computer Graphics,* vol. 18, no. 5, pp. 729-740, May 2012.
[25] M. Eitz, R. Richter, K. Hildebrand, T. Boubekeur, and M. Alexa, "Photosketcher: Interactive Sketch-Based Image Synthesis," *IEEE Computer Graphics and Applications,* vol. 31, no. 6, pp. 56-66, Nov./Dec. 2011.
[26] D. Kurlander, T. Skelly, and D. Salesin, "Comic Chat," *Proc. ACM SIGGRAPH,* pp. 225-236, 1996.
[27] A. Shamir, M. Rubinstein, and T. Levinboim, "Generating Comics from 3D Interactive Computer Graphics," *IEEE Computer Graphics and Applications,* vol. 26, no. 3, pp. 53-61, May 2006.
[28] J. Assa, Y. Caspi, and D. Cohen-Or, "Action Synopsis: Pose Selection and Illustration," *ACM Trans. Graphics,* pp. 667-676, 2005.
[29] S. Zhou, H. Fu, L. Liu, D. Cohen-Or, and X. Han, "Parametric Reshaping of Human Bodies in Images," *ACM Trans. Graphics,* vol. 29, pp. 126:1-126:10, July 2010.
[30] F. Xu, Y. Liu, C. Stoll, J. Tompkin, G. Bharaj, Q. Dai, H.-P. Seidel, J. Kautz, and C. Theobalt, "Video-Based Characters: Creating New Human Performances from a Multi-View Video Data base," *ACM Trans. Graphics,* vol. 30, pp. 32:1-32:10, Aug. 2011.
[31] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A Discriminatively Trained, Multiscale, Deformable Part Model," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR),* pp. 1-8, June 2008.
[32] P.F. Felzenszwalb and D.P. Huttenlocher, "Efficient Graph-Based Image Segmentation," *Int'l J. Computer Vision,* vol. 59, no. 2, pp. 167-181, 2004.
[33] M.J. Jones and J.M. Rehg, "Statistical Color Models with Application to Skin Detection," *Int'l J. Computer Vision,* vol. 46, no. 1, pp. 81-96, 2002.
[34] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive Foreground Extraction Using Iterated Graph Cuts," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 309-314, 2004.
[35] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, "Lazy Snapping," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 303-308, 2004.
[36] A. Hernandez, M. Reyes, S. Escalera, and P. Radeva, "Spatio-Temporal Grabcut Human Segmentation for Face and Pose Recovery," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition Workshops (CVPRW),* pp. 33-40, June 2010.
[37] S. Belongie, J. Malik, and J. Puzicha, "Shape Matching and Object Recognition Using Shape Contexts," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 4, pp. 509-522, Apr. 2002.
[38] J. Ho, A. Peter, A. Rangarajan, and M.-H. Yang, "An Algebraic Approach to Affine Registration of Point Sets," *Proc. IEEE Int'l Conf. Computer Vision (ICCV),* pp. 1-8, 2009.
[39] P.F. Felzenszwalb and J.D. Schwartz, "Hierarchical Matching of Deformable Shapes," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition (CVPR),* pp. 1-8, 2007.
[40] D. Gavrila, "A Bayesian, Exemplar-Based Approach to Hierarchical Shape Matching," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 29, no. 8, pp. 1408-1421, Aug. 2007.
[41] B.J. Frey and D. Dueck, "Clustering by Passing Messages between Data Points," *Science,* vol. 315, pp. 972-976, 2007.
[42] C. Harris and M. Stephens, "A Combined Corner and Edge Detection," *Proc. Fourth Alvey Vision Conf.,* pp. 147-151, 1988.
[43] A. Levin, D. Lischinski, and Y. Weiss, "A Closed-Form Solution to Natural Image Matting," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 30, no. 2, pp. 228-242, Feb. 2008.
[44] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley, "Color Transfer between Images," *IEEE Computer Graphics Applications,* vol. 21, no. 5, pp. 34-41, 2001.
[45] Z. Lin and L. Davis, "Shape-Based Human Detection and Segmentation via Hierarchical Part-Template Matching," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 32, no. 4, pp. 604-618, Apr. 2010.
[46] J.-F. Lalonde, A.A. Efros, and S.G. Narasimhan, "Estimating Natural Illumination from a Single Outdoor Image," *Proc. IEEE Int'l Conf. Computer Vision (ICCV),* pp. 183-190, 2009.
[47] Y. Zhang and R. Tong, "Environment-Sensitive Cloning in Images," *Visual Computing,* vol. 27, nos. 6-8, pp. 739-748, June 2011.



Tao Chen received the BS degree in fundamental science class and the PhD degree in computer science from Tsinghua University in 2005 and 2011, respectively. He is currently a postdoctoral researcher in Department of Computer Science and Technology, Tsinghua University, Beijing. His research interests include computer graphics, computer vision and image/video composition. He received the Netexplorateur Internet Invention Award of the World in 2009, and China Computer Federation Best Dissertation Award in 2011.



Ping Tan received the BS degree in applied mathematics from Shanghai Jiao Tong University, China, in 2000, and the PhD degree in computer science and engineering from the Hong Kong University of Science and Technology in 2007. He joined the Department of Electrical and Computer Engineering at the National University of Singapore as an assistant professor in 2007. He received the MIT TR35@Singapore Award in 2012. His research interests include computer vision and computer graphics. He is a member of the IEEE and ACM.

Li-Qian Ma received the BS degree from Tsinghua University. He is currently a postgraduate at the Department of Computer Science and Technology, Tsinghua University. His research interests include image/video editing and real-time rendering.

Ming-Ming Cheng received the BS degree from the Xidian University in 2007. He is currently working toward the PhD degree at Tsinghua University. His research interests include computer graphics, computer vision, image processing, and sketch-based image retrieval. He has received the Google PhD fellowship award, the IBM PhD fellowship award, and the "New PHD Researcher Award" from Chinese Ministry of Education. He is a member of the IEEE.

Ariel Shamir received the PhD degree in computer science in 2000 from the Hebrew University in Jerusalem. He is an associate professor at the School of Computer Science at the Interdisciplinary Center in Israel. He spent two years at the Center for Computational Visualization at the University of Texas in Austin. During 2006, he held the position of visiting scientist at Mitsubishi Electric Research Labs in Cambridge MA. He has numerous publications in journals and international refereed conferences. He has a broad commercial experience working with and consulting numerous companies. He is a member of the ACM SIGGRAPH, IEEE Computer, and Eurographics societies.

Shi-Min Hu received the PhD degree from Zhejiang University in 1996. He is currently a professor in the Department of Computer Science and Technology at Tsinghua University, Beijing. His research interests include digital geometry processing, video processing, rendering, computer animation, and computer-aided geometric design. He is associate editor-in-chief of *The Visual Computer*, and on the editorial boards of *Computer-Aided Design and Computer & Graphics*. He is a member of the IEEE and ACM. He is the corresponding author of this paper.