PoseShop: Human Image Database Construction and Personalized Content Synthesis

Tao Chen[1] Ping Tan[2] Li-Qian Ma[1] Ming-Ming Cheng[1] Ariel Shamir[3] Shi-Min Hu[1]
[1]TNList, Department of Computer Science and Technology, Tsinghua University
[2]National University of Singapore [3]The Interdisciplinary Center

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 multi-frame *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 manipulation and synthesis operations. Missing content in an image has been completed or image parts replaced using an image database [1], [2], [3]; even whole images have been synthesized from scratch by exploiting segmentation and labeling of a large set of images [4], [5]. 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 [6], [7], [8]. 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 Chen et al. work [5], 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 segmented human characters, many more than popular image databases such as [7], [8].

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 vs. short shirt and pants, patterned vs. 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. First, 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. Second, we present 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

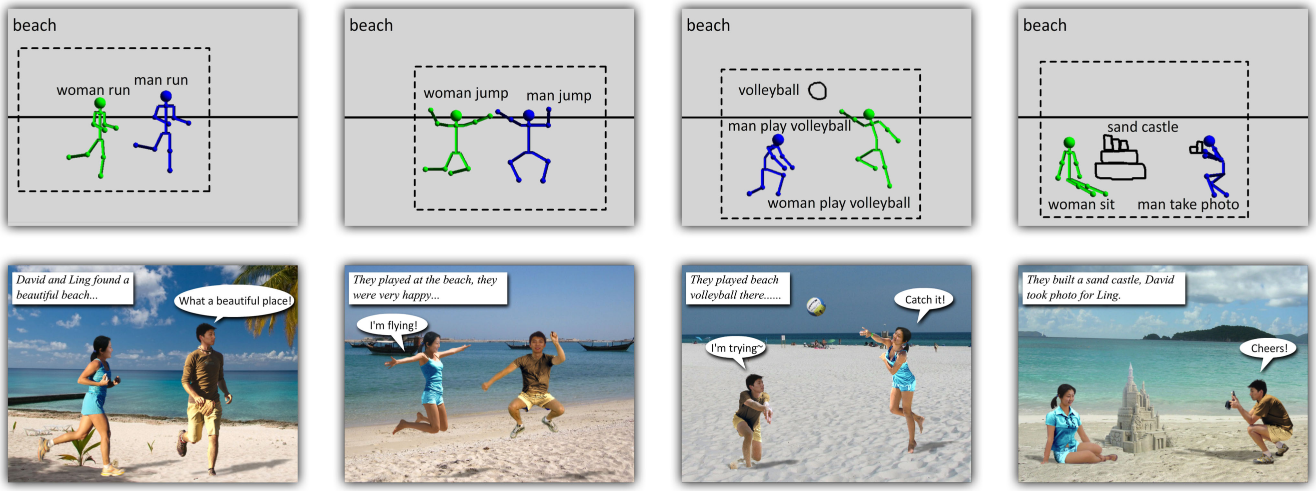


Fig. 1. Using our segmented human character database to create personalized comic-strips. The user provides sketches (first row) and some head images for each character, and the application generates a photorealistic comic-strip (second row). It finds people with similar pose, personalizes them using 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).

of specific images (e.g. of segmentation) are fed back to the database and stored.

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 [9], [10], [4], [5] 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 automatically 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. 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 non-professional 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 [11], Daimler pedestrian database [12], INRIA database for human detection [13] and HumanEva [14] 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 [6], Caltech 256 [7] and LabelMe [8]. 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 [3] and faces [2]. Amazon Mechanical Turk can also be used [15] 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 [16] and [17]. 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 [18] 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.

Our content synthesis is related to content aware image synthesis works, where a series of methods [9], [10], [1], [4], [19], [5] have been proposed with different strategies on selecting image components. Hays and Efros [1] completed missing image regions by images with similar low level descriptors in an online image database. Diakopoulos et al. [9] and Johnson et al. [10] specified image content with text labels and composed a picture according to an automatically labeled and segmented database. Lalonde et al. [4] further used camera pose and illumination condition to choose images for high quality composition. Eitz et al. [19] selected pictures for image composition according to sketches. Chen et al. [5] proposed the *Sketch2Photo* system that combines text labels and sketches for image selection. All these methods used generic databases with different object categories. 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 multi-frame comics composition.

Automatic comic-strips generation was studied previously by Kurlander et al. [20] 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 [21] from interactive computer games and in [22] from action videos. However, in all these systems, a general non-personal 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 [23]. However, this does not solve the appearance consistency and is difficult to scale for huge databases of images. Xu et al. [24] capture a small database of multi-view 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 on a filtering scheme and corresponding methodological improvements.

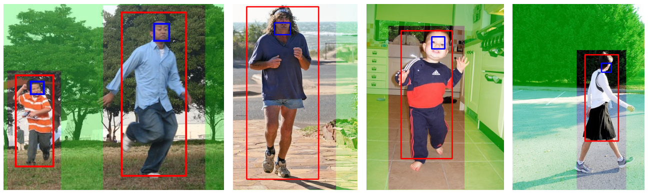


Fig. 2. Human detection results. The red frame is the bounding boxes of detected human. The blue frame indicates detected faces. The region overlayed in green contains 'definite background pixels'.

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 about 300 verbs and phrases [1]. We combine each verb/phrase with 'man', 'woman', 'boy' or 'girl' respectively, and search for images using the combined keywords online. This combination generates over 1,200 queries, where each one typically returns around 3,000 images. Excluding repeated images, we collect around 3 million images, each with an action label. Note that our database construction pipeline can be easily extended to more keywords and image collections.

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

Human detection On each downloaded image we apply the human detector described in [25] 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. We apply conservative parameters (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

[1] This list is provided in the supplementary file.

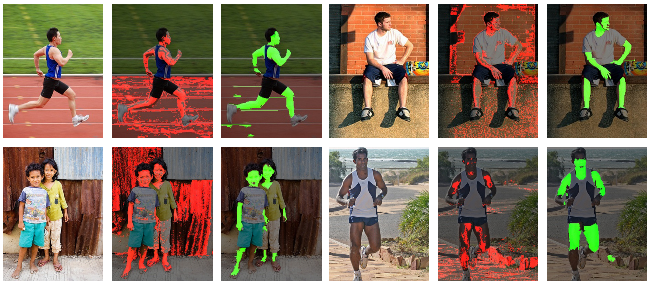


Fig. 3. Skin detection results. For each example, from left to right, they are the original image, detected skin pixels (red) according to [Jones and Rehg 2002], detected skin pixels (green) according to our method.

humans are found. From images that contain multiple human characters, we only keep the non-overlapping ones. Typically, we detect about 4,000 distinct human characters for each keyword.

Algorithm friendly image filtering Following the idea of [5], we only maintain "algorithm friendly" images. These are images where the foreground is well separated from the background. We apply a generic image segmentation algorithm [26] (with parameter $k = 500$) on each of the human character images. Next, we sort all images downloaded from the same keyword according to the number of segments that lie outside the character bounding box. For each keyword, we only keep 50% of the images with the lower segments count. After this filtering, typically, 2,000 human characters are left for each keyword.

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 be more accurate. We employ accurate skin detection to alleviate this problem. The result of our skin detection is also used to estimate clothing attributes in a later stage.

Existing skin detection methods build appearance models for skin and non-skin pixels to detect skin regions. Here, we adopt the Gaussian Mixture Model (GMM) with 16 Gaussian components in the RGB color space [27], 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 non-skin at every pixel. A pixel is considered as skin if this ratio is above a 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 non-skin 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 by minimizing the following function:

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

The key idea 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 non-skin (background). Therefore, we want to 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 [27] 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 step, we refine the skin model. For this refinement, we adjust its Gaussian weights according to the detected skin 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 $(t-1)^{\text{th}}$ iteration, the weight vector at t^{th} 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, $\mathbf{\Pi}_1^t = \gamma(\sum_{i\in A} p_1^i, ..., \sum_{i\in A} p_n^i)$, where A indicates all skin pixels detected in the face region, p_j^i is the probability that the i-th pixel is generated by the j-th 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 the above 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 the morphology 'close' operation 3 times to connect scattered skin regions. Our adaptive skin detection generates better results than the original approach of [27] (see Fig. 3). We compare the ratio of pixels incorrectly detected as skin and non-skin on 100 images with manually marked ground truth. The false skin and false non-skin ratio for [27] are 47.78% and 46.58% respectively, while our detection reduced these ratios to 28.95% and 27.51% on the same images. More comparison and analysis are included in our supplementary files.

Segmentation Each detected human character is segmented from its background with the help of skin detection by graph-cut based segmentation techniques [28], [29]. Hernandez et al. also use GrabCut to segment human from videos [30], but their method relies on temporal redundancy of video, which will 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.

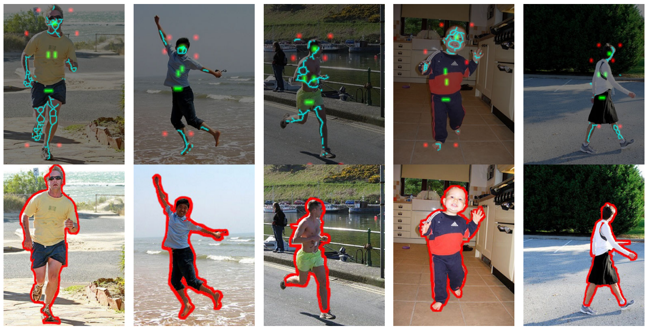


Fig. 4. Character segmentation results. The first row shows the automatically identified 'definite human pixels' (green) and 'definite background pixels' (red). The second row are results from graph-cut based segmentation.

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 and skin regions. **B** pixels (marked in red in Fig. 4) are defined as the four corners of the head bounding box and the two lower corners of the feet. 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 right most of Fig. 2). With the help of skin region detection, our final segmentation includes both legs.

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. Then, we check the consistency of the segmented contours of each character with each of the representative poses. In [5], the "shape context" feature measure [31] is suggested for contour filtering. Using this scheme, after filtering, there are still about 30% 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 fast 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 to each other.

We first apply a closed-form affine registration algorithm [32] to align the segmentation contour to the representative poses. 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 1500 images for each keyword. Next, we apply the shape context [31] similarity criterion to rank the remaining images and keep only 1000 images for each keyword. Finally, we use hierarchical tree [33] as contour similarity measurement to rank remaining images and keep the 500 top ranked ones. At each step, if there are multiple representative poses, the best consistency images from all poses are used for ranking.

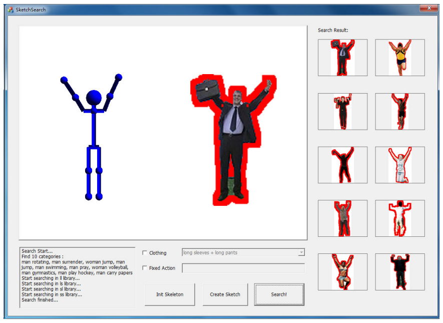
Attributes estimation We define three binary clothes attributes, namely 'long sleeves' vs. 'T-shirts' (shirt), 'long pants' vs. 'shorts' (pants) and 'single color' vs. 'patterned' (color). The first two attributes are detected according to the ratio of skin pixels in upper and lower bodies respectively. 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 1000 manually annotated images. To separate shirts from pants we divide the segmented human figure vertically 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 color attribute is decided by evaluating the color distribution and edge complexities of non-skin pixel regions within the upper and lower parts of the body. The color 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 non-skin 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 non-skin 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.

The index tree is built by iteratively clustering images by affinity propagation [34] with default parameter value (preference $= 1$). Shape context [31] is used to measure the similarity between two segmented character images for clustering. In the cascade contour filtering stage, each image is associated with a representative pose having the most similar contour. Hence, for computational efficiency, we first group representative poses (3000 poses in total). When a group contains less than 10 representative poses, each pose is replaced by the character images associated with it. This iterative grouping leads to a tree of height 6.



```
Search Start...
Find 10 categories :
man rotating, man surrender, woman jump, man
jump, man swimming, man pray, woman volleyball,
man gymnastics, man play hockey, man carry papers
Start searching in ll library...
Start searching in ls library...
Start searching in sl library...
Start searching in ss library...
Search finished...
```

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.

To test our pipeline, beginning with three 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%. 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 limitless 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. The user can provide the contour of a human pose as a query. To make things simpler, we designed an intuitive interface where the user can manipulate the joints of a 2D skeleton to specify a pose (see Fig. 5). A 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 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 ten 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.) 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% images are true positives, i.e. images with a character of the right action. We also manually check the clothes attributes for these images. We find 78.0%, 83.0%, 70% and 75.5% of them 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%. The average false positive rates (the ratio of non-human pixels labeled as human) are 1.2%, 1.2%, 0.8% and 0.9%. We provide all 200 segmentations and ground truth in the supplementary file. 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% 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 trade-off 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. 6, we plot the ratio of true positive images as a function of the number of images selected for each action (blue curve). It is typically over 90% in the top 100 images and gradually drops to about 80% in the top 500 images which is the number we selected to build our database. We compare our results to the results without applying the cascade contour filtering (but after algorithm friendly filtering and human segmentation), where the segmented human images are sorted only by the image search engine (green curve). We also compare our results to Sketch2Photo [5] (red curve), which use a user drawn contour and a text label to search images from the Internet directly.

Fig. 6 illustrates two conclusions. First, just using human detection and our improved skin detection (green curve), human segmentation results could constantly provide about 50% true positives with correct content and pose. The contour filtering results (illustrated by Stetch2Photo shown on the red curve) sometimes fall 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 much better performance. In summary, because of more advanced detection, segmentation and filtering during the database construction, the true positive ratio of PoseShop is consistently higher. Note that the search queries also perform much faster than in a general Internet search.

For application purposes, it is more important to check the quality of queried images from the database. Fig. 7 shows some images returned by the following queries: 'a man runs in T-shirts and shorts' and 'a man skiing in long sleeves and pants'. Such queries take about 1 second to run in a computer

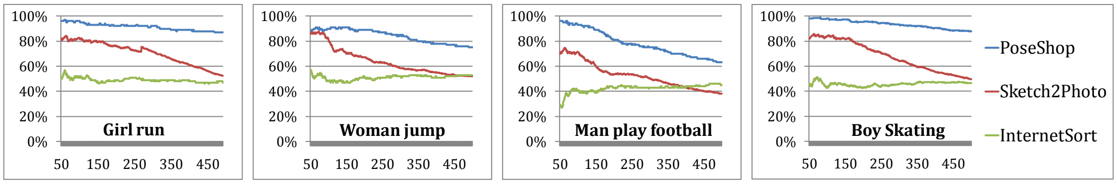


Fig. 6. Database quality vs. number of images. When more images are selected for each action, the true positive ratio will drop. The true positive ratio of the proposed method is consistently higher than the one of Sketch2Photo (see text for more details).

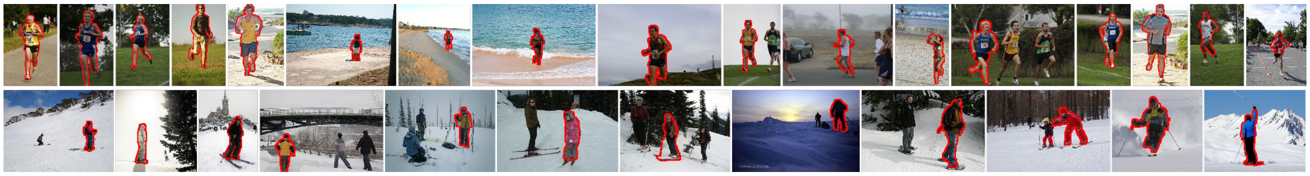


Fig. 7. Images obtained by querying 'a man runs in T-shirts and short' (first row) and 'a man skis in long-sleeves and pants' (second row).

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

with 2 × Quad Core i7 920 CPU and 12G 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. 8. On average, the clustering decrease true positives by 5% in the top 100 returned results. Still, most queries have high true positive ratio of above 70%.

6 Applications

Our character image database opens the door to many applications. There are mainly two possible types of applications: for analysis and for synthesis purposes. 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. For synthesis, our database and querying tools could be used with existing image composition systems like *Photo Clip Art* [4] or *Sketch2Photo* [5]. The additional attributes like actions and clothing type could make the compositions more controllable.

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

6.1 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 our database. We identify correlated *actions* based on their poses. 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. 9 shows an example of the correlation graph of 32 selected actions. According to the graph, most correlations we found are consistent to the semantic meaning of the action keywords. It took 140 minutes to calculate this graph on a 2 × Quad Core i7 920 CPU and 12G 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 learn only from the visual information associated with them.

Such action correlation analysis could also be used to improve the results of querying the database. If **B** is found to be related to **A**, some images of **B** can also be associated with the action label of **A**. We use this scheme to enhance our database query results. We compare the images of action **B** with the representative poses of **A**. An image with similarity

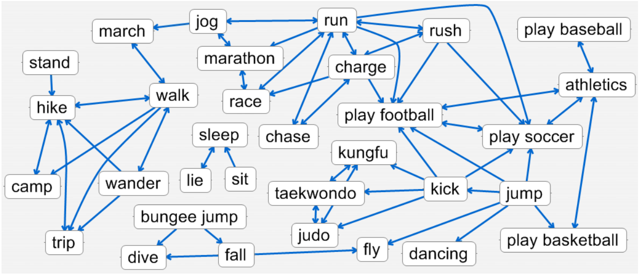


Fig. 9. The correlation graph of different actions. We use an arrow **A**→**B** to indicate action **B** is related to action **A** (according to the definition in Section 6.1). Note that this correlation is not symmetric, i.e. **A**→**B** does not imply **B**→**A**.

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 versatility and ability to find good matches in the database.

6.2 Personalized comic-strips synthesis

Motivated by *Sketch2Photo* system [5] we wanted to generate personalized comic-strips. In our system, the user can generate a realistic style comic-strips of himself or his/her friends. The input to the system 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 characters 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 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.

Head swap People recognize characters by their faces, Bitouk et al. [2] 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 4 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 4 directions. We use the orientation of the input skeleton to decide which of these 4 images to use.

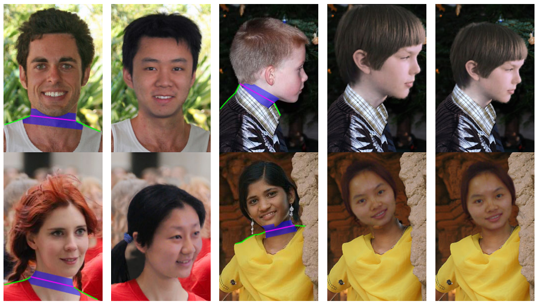


Fig. 10. Head swap result. For each example, from left to right, they are the original image, result by automatic swapping and interactively refined result. The left two examples do not need any interaction.

Once the target and source head images are decided, we apply the following 6 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. non-skin 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 segment above this line is 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 [35] within the face. The optimal transformation is the one that gives minimum set to set distance. [2] 6) Finally, we apply blending as proposed in [5] to paste the new head onto the target image.

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 our human database and further refined it by alpha matting [36] to estimate the fractional clothes region.

Reinhard et al. [37] 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

[2] The set to set distance between a set of points $p_i, 1 \leq i \leq N$ and $q_j, 1 \leq j \leq M$ 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.

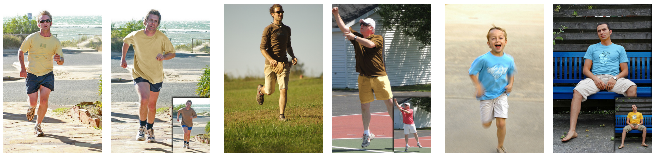


Fig. 11. clothes swapping results. In each example, the source image on the right is adjusted according to the target image on the left. The original source image before swapping is shown at the lower-right corner.

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 color swapping are shown in Fig. 11.

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%.

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.

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, S_i are the candidate background, human character and scene item components in the i^{th} frame respectively. The first term in the sum, $C_i(B_i, I_i, S_i)$ measures the local cost of the i^{th} frame using the hybrid blending cost of [5]. 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.

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 needs 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.

Note that when a user uses a specific image from the database and refines its segmentation, the results are recorded and stored in the database itself. Hence, this, and similar types of applications that utilize the database, can also be used to continually enhance the segmented database quality.

6.3 Example Results

We composed several comic-strips to demonstrate the application of our PoseShop database in personalized content

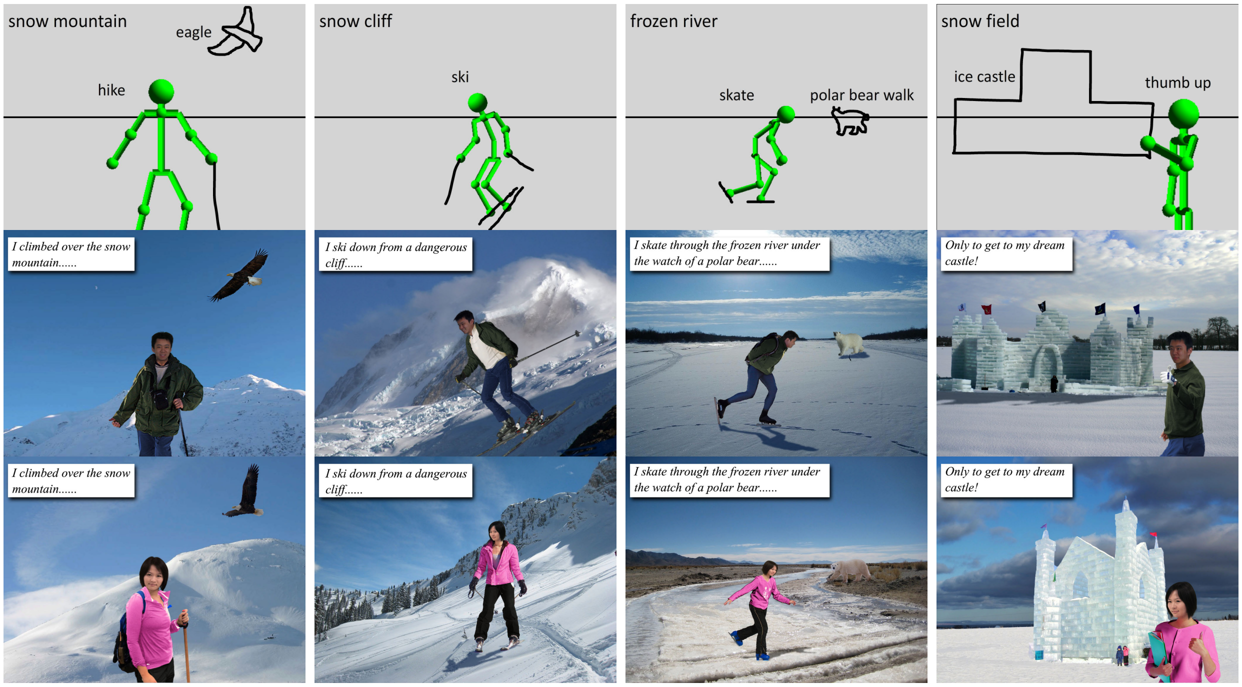

snow mountain
eagle
hike
snow cliff
ski
frozen river
skate
polar bear walk
snow field
ice castle
thumb up
I climbed over the snow mountain......
I ski down from a dangerous cliff......
I skate through the frozen river under the watch of a polar bear......
Only to get to my dream castle!
I climbed over the snow mountain......
I ski down from a dangerous cliff......
I skate through the frozen river under the watch of a polar bear......
Only to get to my dream castle!

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

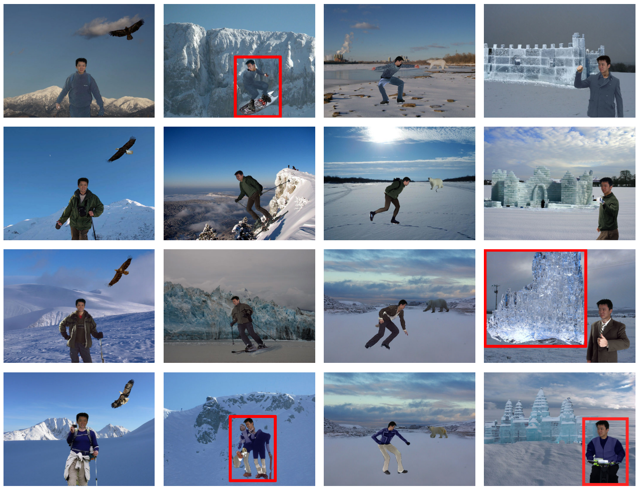


Fig. 12. Four automatic compositions created for ‘ski story’. Red frames indicate incorrect scene items.

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 five 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 ten minutes for each comic-strip (please refer to the supplementary video for more details). The most time consuming interaction is segmentation refinement of the characters and scene items, which usually takes four to five minutes, i.e. one 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 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). 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. 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’ vs. ‘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

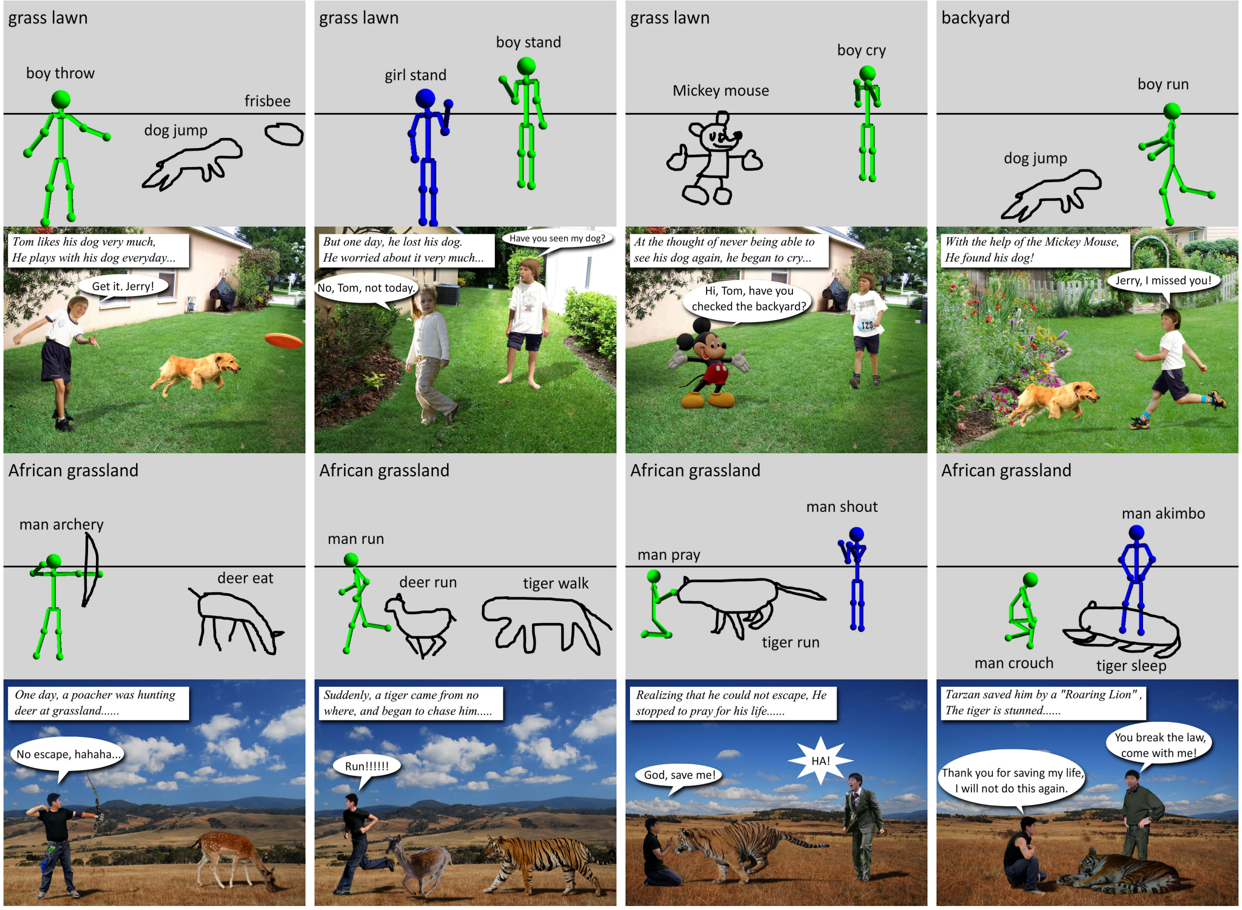

grass lawn
boy throw
frisbee
dog jump
Tom likes his dog very much, He plays with his dog everyday...
Get it. Jerry!
grass lawn
boy stand
girl stand
But one day, he lost his dog. He worried about it very much...
Have you seen my dog?
No, Tom, not today.
grass lawn
boy cry
Mickey mouse
At the thought of never being able to see his dog again, he began to cry...
Hi, Tom, have you checked the backyard?
backyard
boy run
dog jump
With the help of the Mickey Mouse, He found his dog!
Jerry, I missed you!
African grassland
man archery
deer eat
One day, a poacher was hunting deer at grassland......
No escape, hahaha...
African grassland
man run
deer run
tiger walk
Suddenly, a tiger came from no where, and began to chase him.....
Run!!!!!!
African grassland
man shout
man pray
tiger run
Realizing that he could not escape, He stopped to pray for his life......
God, save me!
HA!
African grassland
man akimbo
man crouch
tiger sleep
Tarzan saved him by a "Roaring Lion" , The tiger is stunned......
You break the law, come with me!
Thank you for saving my life, I will not do this again.

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.

jackets. The image components used for these compositions are provided in the supplementary files.

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 5 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. Additional results and details of this evaluation are provided in the supplementary file.

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. As a result, user interaction is necessary in our content synthesis systems mostly to refine 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. 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. 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 such as [38].

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 build a database of these images by indexing over the segmentation contour. This database can be queried using these attributes along with a skeleton that defines the pose and shape. We provide quantitative evaluation of the database by measuring labeling accuracy and per-pixel segmentation accuracy. We further present 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. The user provides a casual sketch with text labels and our system creates some automatic results which are further easily refined in an interactive manner. Comparison to an expert PhotoShop user created results suggests that our system is efficient and easy to use.

We plan to make the database and the query library publicly available to assist relevant 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 could also be improved such as automatic positioning and recomposition, better segmentation and possible enhancement with automatic text and story understanding.

References

[1] J. H. Hays and A. A. Efros, "Scene completion using millions of photographs," *ACM Trans. on Graph*, vol. 26, no. 3, pp. 4:1– 7, 2007.

[2] D. Bitouk, N. Kumar, S. Dhillon, P. Belhumeur, and S. K. Nayar, "Face swapping: automatically replacing faces in photographs," *ACM Trans. on Graph.*, vol. 27, no. 3, pp. 39: 1–8, 2008.

[3] L. Tao, L. Yuan, and J. Sun, "Skyfinder: attribute-based sky image search," *ACM Trans. on Graph.*, vol. 28, no. 3, pp. 68: 1–5, 2009.

[4] J.-F. Lalonde, D. Hoiem, A. A. Efros, C. Rother, J. Winn, and A. Criminisi, "Photo clip art," *ACM Trans. on Graph.*, vol. 26, no. 3, pp. 3:1–10, 2007.

[5] T. Chen, M.-M. Cheng, P. Tan, A. Shamir, and S.-M. Hu, "Sketch2photo: internet image montage," *ACM Trans. Graph*, vol. 28, no. 5, pp. 124: 1–10, 2009.

[6] M. Everingham, A. Zisserman, C. K. I. Williams, and L. Van Gool, "The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results," http://www.pascal-network.org/challenges/VOC/voc2006/results.pdf.

[7] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," *Technical Report*, 2007. [Online]. Available: http://resolver.caltech.edu/CaltechAUTHORS:CNS-TR-2007-001

[8] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "Labelme: A database and web-based tool for image annotation," *IJCV*, vol. 77, no. 1-3, pp. 157–173, 2008.

[9] N. Diakopoulos, I. Essa, and R. Jain, "Content based image synthesis," in *Proc. of CIVR*, 2004.

[10] 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.

[11] A. Georghiades, P. Belhumeur, and D. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE TPAMI*, vol. 23, no. 6, pp. 643–660, 2001.

[12] M. Enzweiler and D. M. Gavrila, "Monocular pedestrian detection: Survey and experiments," *IEEE TPAMI*, vol. 31, no. 12, pp. 2179–2195, 2008.

[13] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. of CVPR*, 2005, pp. 886–893.

[14] L. Sigal and M. J. Black, "Humaneva: Synchronized video and motion capture dataset for evaluation of articulated human motion," *Technical Report*, 2006.

[15] D. F. Alexander Sorokin, "Utility data annotation with amazon mechanical turk," in *Proc. of CVPRW*, June 2008, pp. 1 –8.

[16] J. Sullivan and S. Carlsson, "Recognizing and tracking human action," in *Proc. of ECCV*, 2002, pp. 629–644.

[17] Y. Wang, H. Jiang, M. S. Drew, Z.-N. Li, and G. Mori, "Unsupervised discovery of action classes," in *Proc. of CVPR*, 2006, pp. 1654–1661.

[18] N. Ikizler-Cinbis, R. G. Cinbis, and S. Sclaroff, "Learning actions from the web," in *Proc. of ICCV*, 2009.

[19] M. Eitz, K. Hildebrand, T. Boubekeur, and M. Alexa, "Photosketch: A sketch based image query and compositing system," in *SIGGRAPH 2009 Talk Program*, 2009.

[20] D. Kurlander, T. Skelly, and D. Salesin, "Comic chat," in *Proc. of SIGGRAPH*, 1996, pp. 225–236.

[21] A. Shamir, M. Rubinstein, and T. Levinboim, "Generating comics from 3d interactive computer graphics," *IEEE Comput. Graph. Appl.*, vol. 26, no. 3, pp. 30–38, 2006.

[22] J. Assa, Y. Caspi, and D. Cohen-Or, "Action synopsis: Pose selection and illustration," *ACM Trans. on Graph.*, pp. 667–676, 2005.

[23] S. Zhou, H. Fu, L. Liu, D. Cohen-Or, and X. Han, "Parametric reshaping of human bodies in images," *ACM Trans. on Graph.*, vol. 29, no. 3, 2010.

[24] 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 database," *ACM Trans. Graph.*, vol. 30, pp. 32:1–32:10, August 2011. [Online]. Available: http://doi.acm.org/10.1145/2010324.1964927

[25] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *Proc. of CVPR*, June 2008, pp. 1–8.

[26] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *IJCV*, vol. 59, no. 2, pp. 167–181, 2004.

[27] M. J. Jones and J. M. Rehg, "Statistical color models with application to skin detection," *IJCV*, vol. 46, no. 1, pp. 81–96, 2002.

[28] C. Rother, V. Kolmogorov, and A. Blake, ""grabcut": interactive foreground extraction using iterated graph cuts," *ACM Trans. on Graph.*, vol. 23, no. 3, pp. 309–314, 2004.

[29] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, "Lazy snapping," *ACM Trans. on Graph.*, vol. 23, no. 3, pp. 303–308, 2004.

[30] A. Hernandez, M. Reyes, S. Escalera, and P. Radeva, "Spatio-temporal grabcut human segmentation for face and pose recovery," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, june 2010, pp. 33 –40.

[31] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE TPAMI*, vol. 24, no. 4, pp. 509–522, 2002.

[32] J. Ho, A. Peter, A. Rangarajan, and M.-H. Yang, "An algebraic approach to affine registration of point sets," *Proc. of ICCV*, pp. 1–8, 2009.

[33] P. F. Felzenszwalb and J. D. Schwartz, “Hierarchical matching of deformable shapes,” in *Proc. of CVPR*, 2007, pp. 1–8. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2007.383018

[34] B. J. Frey and D. Dueck, “Clustering by passing messages between data points,” *Science*, vol. 315, pp. 972–976, 2007. [Online]. Available: www.psi.toronto.edu/affinitypropagation

[35] C. Harris and M. Stephens, “A combined corner and edge detection,” *Proceedings of The Fourth Alvey Vision Conference*, pp. 147–151, 1988.

[36] A. Levin, D. Lischinski, and Y. Weiss, “A closed-form solution to natural image matting,” *IEEE TPAMI*, vol. 30, no. 2, pp. 228–242, 2008.

[37] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley, “Color transfer between images,” *IEEE Comput. Graph. Appl.*, vol. 21, no. 5, pp. 34–41, 2001.

[38] J.-F. Lalonde, A. A. Efros, and S. G. Narasimhan, “Estimating natural illumination from a single outdoor image,” in *Proc. of ICCV*, 2009.