Garment Modeling with a Depth Camera

Xiaowu Chen* Bin Zhou* Feixiang Lu Lin Wang Lang Bi
State Key Laboratory of Virtual Reality Technology & Systems, Beihang University

Ping Tan
Simon Fraser University

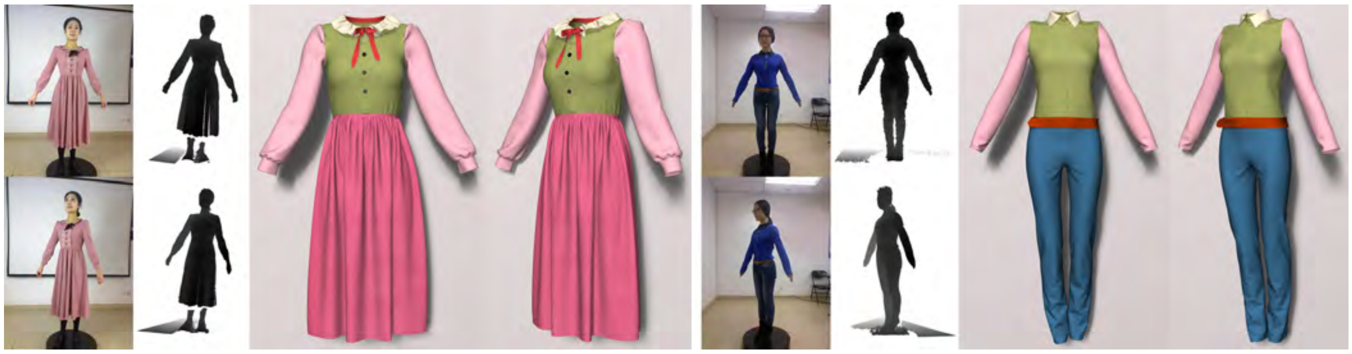
Figure 1: *Given an RGBD sequence of a dressed garment, our system produces a detailed high quality 3D garment model by detecting and assembling suitable components. Each color in the models denotes a different component.*

Abstract

Previous garment modeling techniques mainly focus on designing novel garments to dress up virtual characters. We study the modeling of real garments and develop a system that is intuitive to use even for novice users. Our system includes garment component detectors and design attribute classifiers learned from a manually labeled garment image database. In the modeling time, we scan the garment with a Kinect and build a rough shape by KinectFusion from the raw RGBD sequence. The detectors and classifiers will identify garment components (e.g. collar, sleeve, pockets, belt, and buttons) and their design attributes (e.g. *falbala collar* or *lapel collar*, *hubble-bubble sleeve* or *straight sleeve*) from the RGB images. Our system also contains a 3D deformable template database for garment components. Once the components and their designs are determined, we choose appropriate templates, stitch them together, and fit them to the initial garment mesh generated by KinectFusion. Experiments on various different garment styles consistently generate high quality results.

CR Categories: I.3.5 [Computing Methodologies]: Computer Graphics—Computational Geometry and Object Modeling I.3.7 [Computing Methodologies]: Computer Graphics—Three-Dimensional Graphics and Realism;

Keywords: Garment Modeling, Depth Camera, Garment Parsing, 3D Templates, Semantic Modeling

*corresponding authors: {chen, zhoubin}@buaa.edu.cn

1 Introduction

Realistic garment models are necessary to many applications. The majority of existing garment modeling works, such as [Umetani et al. 2011; Meng et al. 2012; Turquin et al. 2004; Robson et al. 2011], focus on creating 3D garment models to dress up virtual characters. Only a few works [Berthouzoz et al. 2013; Zhou et al. 2013] have studied the problem of modeling existing real garments, which has important applications such as virtual try-on [Cordier et al. 2003; Divivier et al. 2004] or prototyping [Volino and Magnenat-Thalmann 2005].

Berthouzoz et al. [2013] automatically convert existing sewing patterns into 3D garment models. This method generates sophisticated models, while the sewing pattern of a garment is not always available. Zhou et al. [2013] model garments from a single image. This method is simple and easy to use, but captures limited details.

We aim at a simple and intuitive solution for high quality modeling of real garments. Our method requires only a consumer-level depth camera like the Microsoft Kinect camera. This setup is low cost and easy to use for novice users with little experience in 3D modeling. A casual user can quickly generate detailed 3D garment models with our method, as shown in the examples in Figure 1.

Our key observation is that most of clothes consist of standard components, such as sleeve, body, collar, pocket, and button, and most components have standard designs. Therefore, we adopt the principles of 'assembly-based modeling' [Funkhouser et al. 2004; Chaudhuri et al. 2011] for rapid and intuitive garment modeling. While [Shao et al. 2012; Chen et al. 2014] apply similar principles for indoor scene modeling, garments have unique features and cannot be modeled by directly applying these methods. For example, garment components such as buttons and pockets are typically small and hard to segment from the 3D scans, design attributes like *lapel collar* and *tailor collar* are also difficult to tell from 3D scans. So our method consists of a sophisticated image analysis component to detect garment components and determine their design attributes, which is not exploited in earlier works.

We first survey garment design rules in [Ingham and Covey 1992], which allow us to represent the semantic structure of a garment by a parsing tree, as shown in Figure 2(c). It represents a garment by a set of predetermined components, e.g. collar, sleeve, pocket, etc.

Each component is further characterized by a few design attributes, e.g. size, shape, type, etc. Each attribute takes a few predetermined values, e.g. *lapel collar*, *stand collar*, *tailor collar*, etc., for collar type. This parsing tree provides strong structure priors to facilitate the modeling process.

In the modeling time, we capture the target garment dressed on a model in standing position with a Kinect. Our program automatically parses the front view RGB image by garment component detectors and design attribute classifiers trained from a manually labeled garment image database. These individually analyzed garment components are further jointly refined in a Bayesian network to exploit the semantic compatibility among components. For example, it will know *hubble-bubble sleeves* often appears with *falbala collars*. If needed (which is rarely the case), the user might interactively change the detection and classification results.

On the other hand, we have an expert-designed 3D deformable template database for garment components. Once the garment components and their designs are fixed, we select the best templates with the same design and stitch them together to form the final garment model. The template selection is guided by registering the template to the initial 3D garment mesh built by KinectFusion [Izadi et al. 2011]. As limited by the size of template database, the result model is not an exact 3D scan of the original garment. Instead, we aim at creating a plausible 3D garment model with the same design.

We claim the following contributions in this work. 1) we present a garment parsing tree summarizing garment modeling rules. Together with our 3D templates of garment components, it forms a compact rule-based garment modeling system, where a user can quickly generate high quality garment models by selecting garment components and their designs. 2) we build a labeled image database with 6,000 garment images, each of which is labeled with the skeleton pose, garment components, and their design attributes. This database and labeling tool should facilitate future research along this direction such as garment modeling, parsing, or garment image search. 3) we build garment component detectors and their design attribute classifiers and jointly analyze their results in a Bayesian network. The parsing result is further combined with 3D templates to build an end-to-end garment modeling system.

2 Related work

Many garment modeling works focus on the application of dressing up virtual characters in films and games. There are also garment retargeting techniques, such as [Guan et al. 2012; Brouet et al. 2012], that can transform garment designs from one character to another with large body shape or pose changes. Pattern-based methods [Fontana et al. 2005; Meng et al. 2012] simulate the real-life design and tailoring process to create garment models, including 2D pattern creation, physically-based simulation, and iteratively optimization of involved parameters, and can achieve high quality effects. Umetani et al. [2011] present an interactive tool, Sensitive Couture (SC), offering garment modeling and editing in pattern design and 3D drape. There are also some well known garment modeling tools, such as Marvelous Designer, vStitcher and Optitex PDS. Most of these systems are also labor intensive and require strong domain knowledge to use.

In comparison, sketch-based methods, such as [Wang et al. 2003; Turquin et al. 2004; Decaudin et al. 2006; Turquin et al. 2007; Mok et al. 2013], are relatively more intuitive. They allow the user to sketch directly over the mannequin figure to create 3D garment models. Robson et al. [2011] further incorporated contextual knowledge to improve loose clothes. While good results can be achieved, sketch-based methods still require strong domain expertise of garment design, and are difficult for non-experts.

Only a few works study the problem of digitizing real garments. Since real garments are often designed with sewing patterns, Berthouzoz et al. [2013] automatically converted existing sewing patterns into 3D garment models. This method can generate sophisticated models, while the sewing pattern of a garment is not always accessible. Zhou et al. [2013] created 3D garment models from a single image. This method is simple and easy to use, but captures limited details. We advocate garment modeling with a depth camera. Our method is intuitive to use and makes high quality modeling possible for novice users. Furthermore, many methods, such as [Scholz et al. 2005; White et al. 2007; Bradley et al. 2008; Popa et al. 2009], have been proposed to capture garment animation from videos. We focus on garment modeling and leave the animation problem for future study.

With the growth of public 3D model collections (e.g. the Google 3DWarehouse) and the increasing popularity of consumer-level depth cameras (e.g. [Li et al. 2013] and [Dou et al. 2015]), many data-driven modeling methods [Brouet et al. 2012; Zuffi and Black 2015] have been presented. These methods digitize real-word 3D scenes and objects by combining existing models. Fisher et al. [2012] used probabilistic models learned from a database of 3D scenes to synthesize new scenes. Chaudhuri and Koltun [2010] built a Bayesian network to assist interactive modeling [Chaudhuri et al. 2011] and shape synthesis [Kalogerakis et al. 2012] by suggesting model components according to the intermediate shape. Following this line, Shao et al. [2012] presented an interactive approach to semantic modeling of indoor scenes with a consumer-level RGBD camera. It retrieves the best 3D template model for each object from a database to reconstruct the final scene. Shen et al. [2012] built structured mesh models from a single scan of a Kinect camera with a mesh repository. Chen et al. [2014] presented a method for automatic semantic modeling of indoor scenes from low-quality RGB-D data, using contextual relationships learned from a 3D indoor scene database. However, most of these methods are designed for indoor scenes or man-made objects, such as chairs or bicycles. They cannot be directly applied to our garment modeling problem. For example, it will be very difficult to apply the techniques in [Shao et al. 2012; Chen et al. 2014] to segment the individual garment components in our examples.

Recently, some garment image parsing methods [Yamaguchi et al. 2012; Liu et al. 2012; Yamaguchi et al. 2015; Yang et al. 2014] are proposed in the field of computer vision. They can recognize simple garment designs and the character pose from a single or multiple images despite view changes, pose variation and occlusion. Directly applying these methods cannot solve our problem because of their noisy results. We focus on garment parsing with known pose and garment segmentation, and generate much more accurate detection and classification results to facilitate modeling.

3 Overview

We aim at digitizing real garments and making high quality modeling intuitive for novice users. Specifically, we focus on creating a plausible 3D garment model with the same design attributes and size. Based on the observations that most of clothes consist of standard components and most components have standard designs, we survey garment design rules [Ingham and Covey 1992] and formulate a compact garment parsing hierarchy (Section 4.1), as shown in Figure 2(c). With respect to the parsing tree, our complete system includes a back-end and a front-end.

The back-end contains two garment databases. The first is a database of expert designed 3D garment component templates (Section 4.2), containing over 1000 various templates, as shown in Figure 3(b). These templates are designed on a standard 3D human

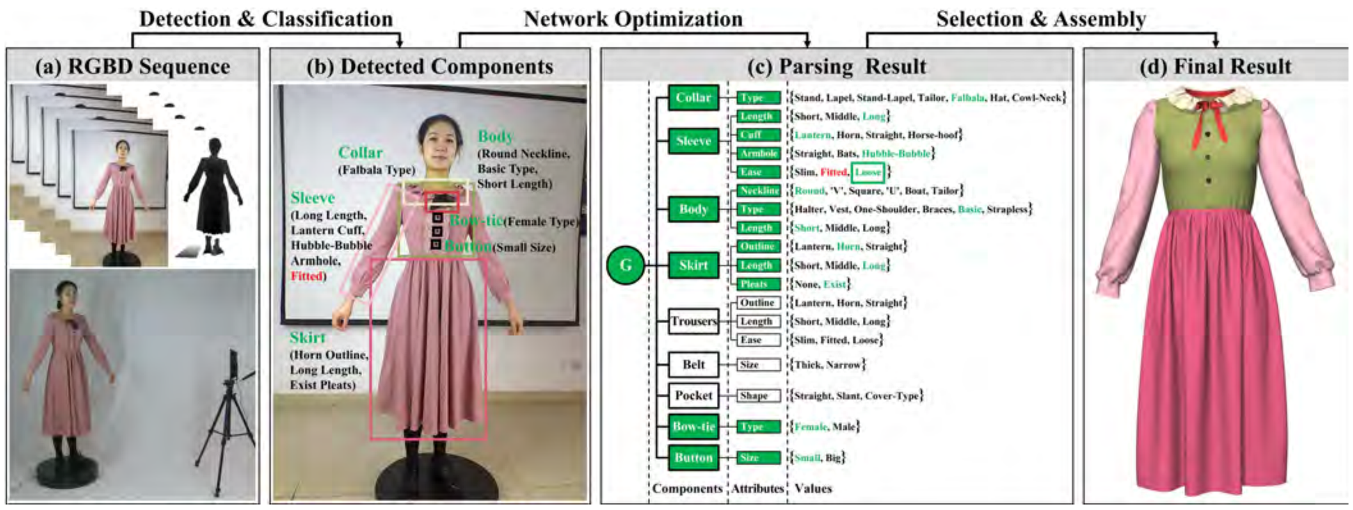

Figure 2: *Pipeline of our system. We first detect different garment components and identify their designs. These results are further jointly analyzed in a Bayesian network to enforce the contextual priors (Attribute value with red color in (b,c) is corrected by the Bayesian network to the right one with green color in (c)). Finally, deformable templates of detected components are stitched together to form final result.*

mannequin, which can be deformed according to the mannequin's skeleton pose to match that of the scanned model. Each template is labeled with its component name, various attributes, and 'slot seams' to assemble adjacent components. Together with the garment parsing tree, we forms an effective rule-based garment modeling system (Section 4.3), where a user can quickly generate high quality garment models by selecting garment components and their designs. The second is a manually labeled image database (Section 5.1), containing 6,000 garment images. Each image is labeled with skeleton pose, garment segmentation, and garment components including their design attributes and boundaries. Some sampled images are shown in Figure 4(a). We train garment component detectors (Section 5.2) and design attribute classifiers (Section 5.3) from these labeled 2D images by the DPM method [Felzenszwalb et al. 2010] and the random forest algorithm [Breiman 2001] respectively. We further learn a Bayesian network [Chaudhuri et al. 2011; Kalogerakis et al. 2012] capturing the semantic compatibility among garment components (Section 5.3).

The front-end is summarized in Figure 2. As shown in (a), we use a fixed Kinect camera to scan a model dressed in the target garment. The model stands on a rotating stage, which rotates roughly one round a minute. This setup is low-cost and intuitive to use even for novice users. As a pre-processing, we first use the KinectFusion system [Izadi et al. 2011] to build an initial mesh from the raw RGBD sequence. We then select a reference garment image where the model faces front. In this reference frame, the garment is segmented out by an interactive tool [Li et al. 2004], and the model's skeleton pose is also interactively labeled. The segmented garment is used to cut the initial mesh interactively, obtaining an initial garment mesh. We back-project the labeled 2D skeleton to the initial garment mesh to upgrade it to 3D, obtaining the 3D skeleton of the model. We apply the component detectors and attribute classifiers to this reference image. The results are shown in Figure 2(b). A Bayesian network jointly optimizes the individually identified garment components, as shown in Figure 2(c). Given the garment parsing result, our system selects the most appropriate component templates and automatically stitches them together to form the final 3D model (Section 6), as shown in Figure 2(d).

While our system includes several components, many of them are standard and common in 3D modeling works, such as Kinect scanning, fusion, segmentation from the background. Beyond those, our system mainly includes garment analysis (detectors, classifiers, and the Bayesian network) and garment synthesis (deforming and stitching templates). While a single image (or video) is easier in data capture, it is hard to achieve our high quality results because of the lack of 3D information.

4 Garment Modeling Rules and Templates

We observe several facts about costume design [Ingham and Covey 1992]. Firstly, garments consist of standard components such as sleeve, body, pocket, belt, and collar, etc. Secondly, each component follows standard design patterns. Typically, a component is characterized by a few design attributes, each of which takes a few predetermined values. For example, a sleeve is characterized by its length, cuff, armhole, and ease. The length attribute can be *short*, *middle*, or *long*, while the cuff attribute can be *lantern*, *horn*, *straight*, or *horse-hoof*. Thirdly, designers usually follow certain guidelines when choosing different garment components. So there is strong contextual constraint among different components. For example, *hubble-bubble sleeve* often appears with *falbala collar*, *straight sleeve* often appears with *stand collar*, and *tailor collar* often appears with *tailor body*. Finally, most components have fixed relative position to a human body and to the other components. For example, sleeves are wrapped around arms and connected to the garment body. These observations provide strong structure priors for garment modeling.

4.1 Garment Parsing Tree

Based on those observations, we define a hierarchical garment representation to facilitate modeling, as shown in Figure 2(c). Following the tree root, there is the layer formed by all garment components, $C = \{C_1, \cdots, C_M\}$. Empirically, we choose to include $M = 9$ different components, including '*Collar*', '*Sleeve*' (both left and right), '*Body*', '*Skirt*', '*Trousers*', '*Pocket*', '*Belt*', '*Bow-tie*' and '*Button*'. Each component is characterized by a handful of design attributes. So we define $C_i = (a_{i,1}, \cdots, a_{i,k}, \cdots, a_{i,N_i})$. For example, the component '*Sleeve*' (C_2) has four attributes, namely, '*Length*'($a_{2,1}$), '*Cuff*'($a_{2,2}$), '*Armhole*'($a_{2,3}$) and '*Ease*'($a_{2,4}$). All attributes from different

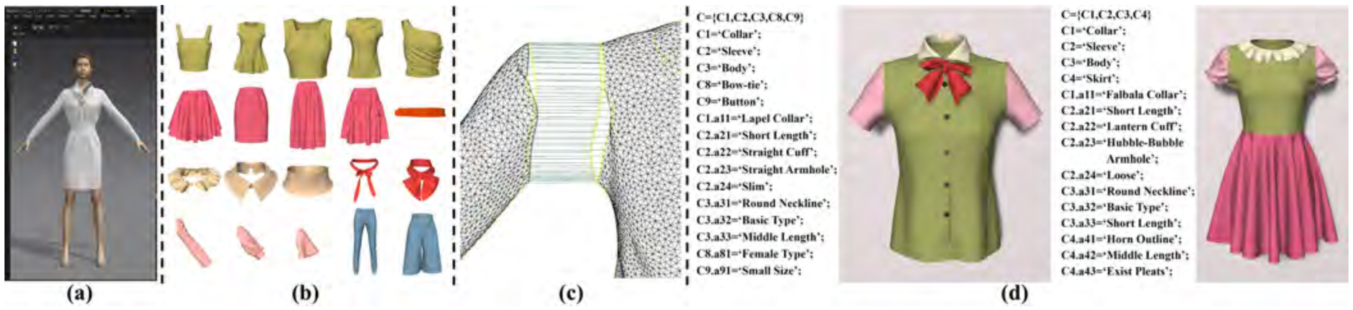

Figure 3: *Our 3D garment component template database. (a) Professional garment design on standard 3D mannequin; (b) Samples of component templates; (c) 'Slot seams' on a sleeve and the body; (d) Generated 3D garment models by manually specifying parsing trees.*

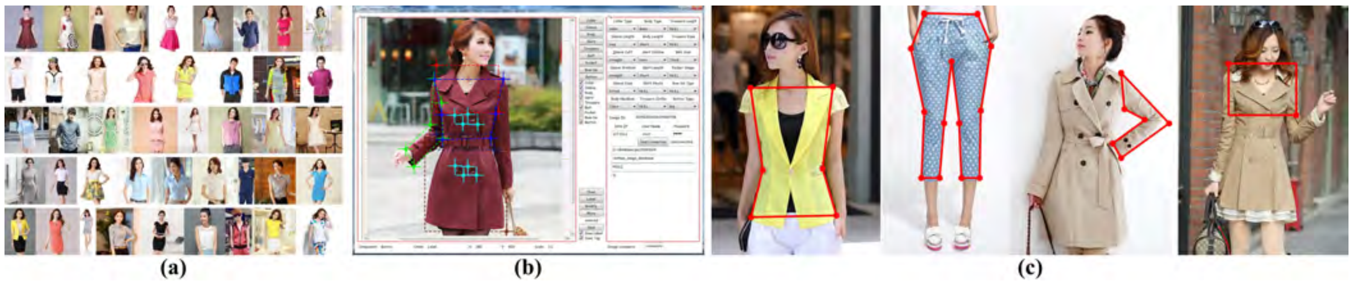

Figure 4: *Our garment image database. (a) Image Samples; (b) Screenshot of our annotation tool; (c) Sampled annotated images.*

components form the another level of the tree. Each attribute takes value from a small predefined set. For example, the '*Armhole*' attribute ($a_{2,3}$) can be '*Straight*', '*Bats*' and '*Hubble-Bubble*'. All values from different attributes form the leaf nodes of the tree. We do not encode position information in the tree, since many garment components have fixed relative positions.

Each garment can be parsed according to this tree. In other words, the structure of a garment is defined by the existence of all components and the attribute values of presented components. For example, the garment shown in Figure 2(b) is represented by the tree in Figure 2(c), where selected components and values are marked in green. Specifically, its parsing tree representation is $\{G = \{C_1, C_2, C_3, C_4, C_8, C_9\}, C_3 = \text{'}Body\text{'}, C_3.a_{3,1} = \text{'}Round\text{'}, \cdots\}$.

4.2 3D Template Database

We construct an expert designed 3D templates database for garment components. Each template in the database is designed by a professional garment designer with the commercial software, Marvelous Designer, as shown in Figure 3(a). The designer creates about 300 garments, including T-shirts, shorts, long-sleeved shirts, pants, skirts. During the creation of these garment models, their components and associated design attributes are all labeled. This gives us $175, 250, 330, 150, 25, 20, 20, 20, 10$ template models for the components '*Collar*', '*Sleeve*', '*Body*', '*Skirt*', '*Trousers*', '*Pocket*', '*Belt*', '*Bow-tie*', and '*Button*' respectively. Some of them are shown in Figure 3(b). Each template is labeled with its 'slot seams', which are control vertices used to seamlessly stitch two neighboring components during assembling. We use the re-mesh algorithm [Yu et al. 2004] to ensure corresponding 'slot seams' at neighboring components to have the same number of vertices. Figure 3(c) shows the 'slot seams' on a sleeve and the garment body.

All these garments are designed on a standard 3D mannequin and naturally rigged with a human skeleton. In this way, the database records the 'canonical position' of each template with respect to the human skeleton. Furthermore, all templates can be deformed according to a skeleton pose, which is useful since the scanned model might be in a different pose than the mannequin.

4.3 Rule-based Modeling

The garment parsing tree and 3D template database together define a rule-based garment modeling system, where a garment model can be created by manually specifying its components and attributes. We show some garment models generated in this way in Figure 3(d). Their parsing trees are shown nearby.

5 Automatic Garment Image Parsing

We build a manually labeled garment image database and train garment component detectors and design attribute classifiers from them. The reference RGB image is then parsed automatically by these detectors and classifiers to determine the semantic garment structure, which is used in Section 6 to create the garment model by assembling appropriate 3D templates.

5.1 Garment Image Database

We collect $260, 000$ garment images from online shopping websites such as Amazon and Taobao. To keep the labeling work manageable, we rank them according to their sales volumes and select the top 6000 images with a mannequin (or model) in standing pose. Some of those images are shown in Figure 4(a). We design an annotation tool to label the garment components. The user can mark various components by simply drawing a predefined polygon. Our interface then popups a dialog to let the user specify corresponding attributes. Meanwhile, our tool enables auto-saving and allows undo operations. A screenshot of this interface is shown in Figure 4(b). Some sampled annotated images are shown in Figure 4(c).

Furthermore, each image is labeled with a skeleton pose and garment segmentation.

5.2 Component Detection

We divide garment components as 'main components' and 'accessory components'. Main components include '*Sleeve*', '*Body*', '*Skirt*' and '*Trousers*'. Their positions are relatively fixed with respect to the skeleton, which enables naive detection with known skeleton pose. For example, *Sleeves* are detected as the image regions surrounding the arms, *Body* is detected as the rectangle formed by the joints at torso and arms. With garment segmentation and the main components' position prior, we can determine the existence of these main components and locate their coarse boundaries. An example with detected sleeve, body and skirt is shown in Figure 5(c).

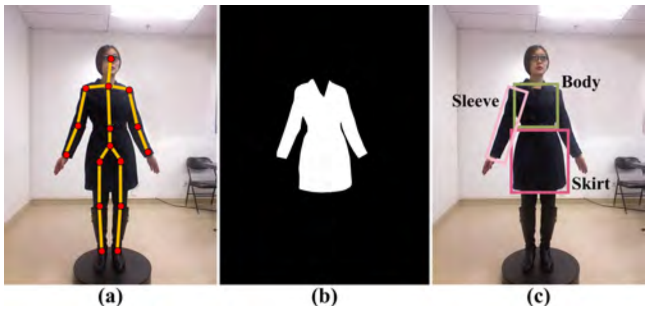

Figure 5: *Main Components Detection.(a) Reference image with model skeleton; (b) Result of garment segmentation; (c) Detected sleeve, body and skirt.*

The accessory components, including '*Pocket*', '*Belt*', '*Bow-tie*', '*Collar*' and '*Button*', require a more sophisticated detector, since their positions are more variant. We learn a deformable-part-model (DPM) [Felzenszwalb et al. 2010; Girshick et al. 2012] to detect accessory components. The DPM models are trained using a discriminative learning algorithm (latent SVM) that only requires bounding boxes of labeled components in training images.

We firstly collect a training set from the labeled images by choosing those annotated by the component's tag. We use the HOG feature [Dalal and Triggs 2005] to train a DPM detector, by fixing the number of models at 6. In testing time, given a reference garment image, the DPM model will generate multiple proposals $\{H_i(x_i, y_i, w_i, h_i), i = 1 \cdots m\}$. Here, m is the total number of the proposals, H_i denotes the i-th proposals at position (x_i, y_i), with width and height specified by w_i and h_i respectively. Such examples are shown in Figure 6(e,g), (m) and (o) for the components '*Pocket*', '*Collar*' and '*Belt*' respectively.

The raw proposals generated from DPM [Felzenszwalb et al. 2010; Girshick et al. 2012] are too noisy. We further combine various priors learned from the labeled database to enhance the detection. These priors include statistics on a component's position and size. To accumulate these priors from the labeled images of various sizes, we normalize all images by the shoulder width and torso length. We combine the position and size prior with the DPM detection score to define an energy as

$$E = f(H_i) f_p(x_i, y_i) f_a(w_i, h_i). \tag{1}$$

Here, f is the score returned by DPM; f_p is the position prior, encoding the most probable position of a component; f_a is the area prior, encoding the most probable size of a component. Both priors are represented as the corresponding histograms of the training

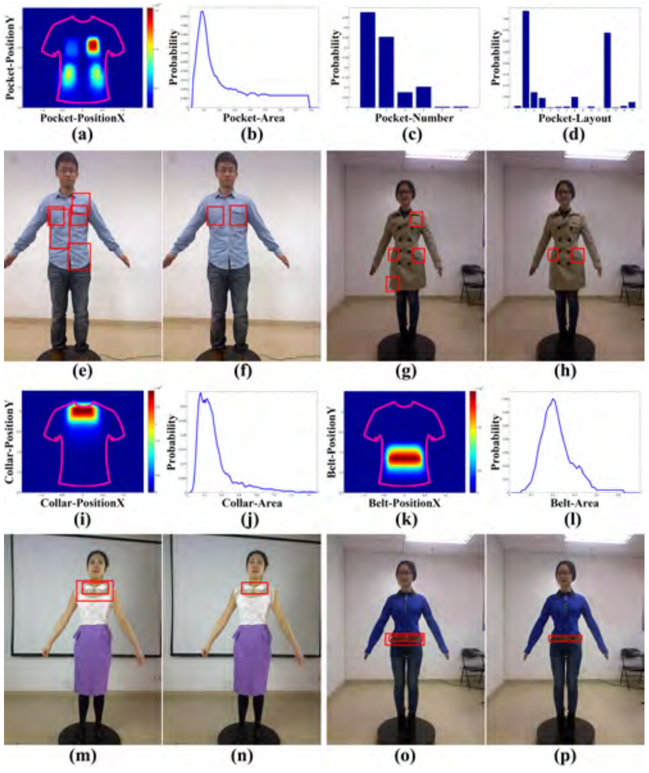

Figure 6: *Statistical priors for accessory components detection. (a), (i) and (k) are position priors for* 'Pocket'*,* 'Collar'*, and* 'Belt' *respectively; (b), (j) and (l) are area priors of* 'Pocket'*,* 'Collar'*, and 'Belt' respectively; (c) and (d) are the priors on the number of pockets and pocket layout; (e,g), (m) and (o) are the initial DPM detection of* 'Pocket'*,* 'Collar' *and* 'Belt' *respectively. (f), (h), (n) and (p) are the final detection when priors are enforced.*

image set. An example of learned f_p and f_a are shown in Figure 6(a)(b), (i)(j) and (k)(l) for the component '*Pocket*', '*Collar*' and '*Belt*' respectively. We compare the maximum value of Equation 1 with a predetermined threshold (0.001 in all our experiments) to detect accessory components. Figure 6(n) and (p) show the collar and belt detection results.

Additional Priors for '*Pocket*'. For the '*Pocket*' component, we further include its priors on layout and number of instances. The prior on number of instance is also a histogram, f_n, as shown in Figure 6(c). The layout prior is slightly more complicated. We assume a garment has no more than four pockets, one in each quad (though the histogram shows there are cases of 5-6 pockets). We define a layout code $L_1L_2L_3L_4$. Each of L_1, L_2, L_3 and L_4 is a binary code indicating if there is a pocket in the corresponding quad. So there are in total $2^4 = 16$ different possible layouts. We also compute a layout histogram, f_l, from the labeled images as shown in Figure 6(d). During detection, we detect a candidate pocket with highest energy (Equation 1) in each quad. We optimize the binary variables L_1, L_2, L_3 and L_4 by maximizing the following energy:

$$f_l(L_1L_2L_3L_4) f_n(L_1 + L_2 + L_3 + L_4) \frac{\sum_{i=1}^{4} L_i E_i}{\sum_{i=1}^{4} L_i}. \tag{2}$$

Here, E_1, E_2, E_3 and E_4 are the energies of the four candidates. This optimization is solved by brute force enumerating. The refined pocket detection results are shown in Figure 6(f) and (h). Each detected instance is enclosed by a polygonal boundary.

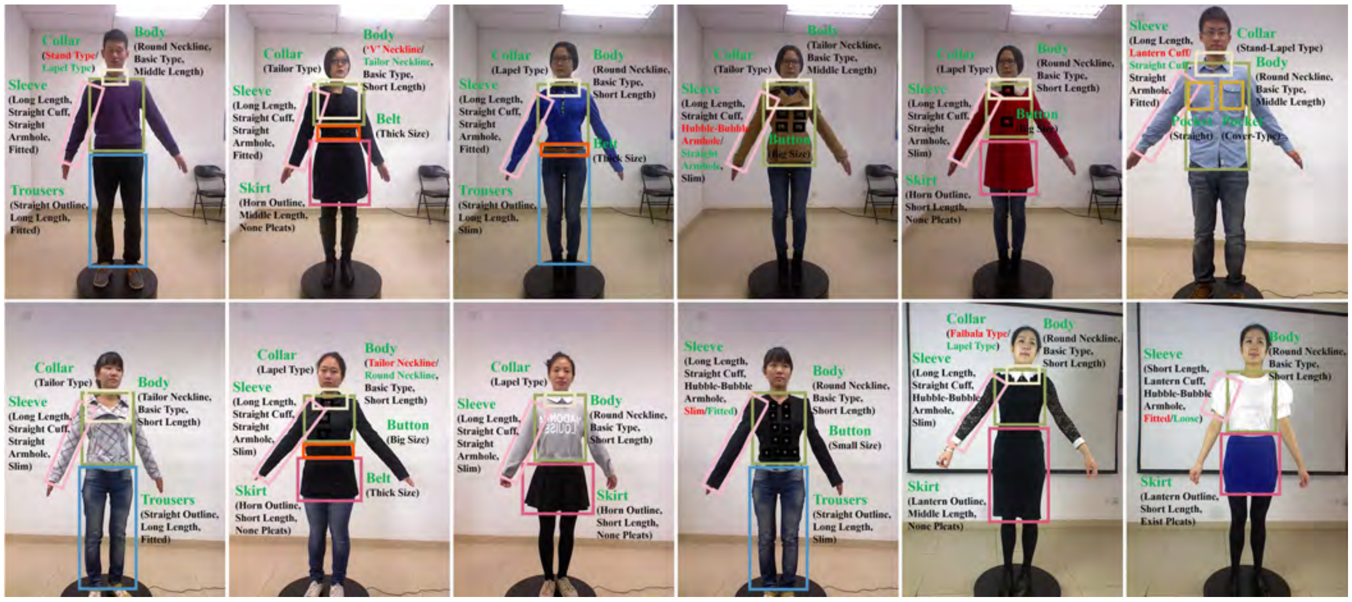

Figure 7: *The results of component detection and attribute classification on our scanned garment. The attribute value with red color is corrected by our Bayesian network to the right one with green color.*

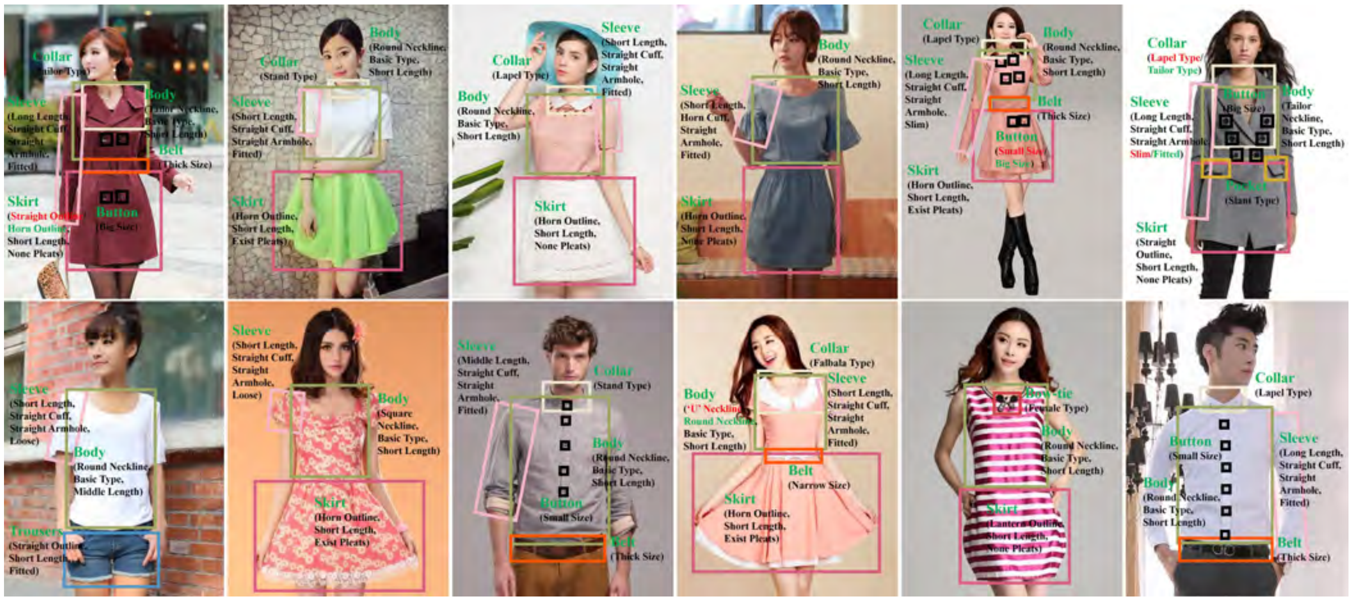

Figure 8: *The results of component detection and attribute classification on our internet images (with manually labeled garment segmentation and skeleton pose). The attribute value with red color is corrected by our Bayesian network to the right one with green color.*

5.3 Attribute Classification

We further estimate the attributes of each detected garment component. For example, the component '*Collar*' has a '*Type*' attribute, which takes value from '*Stand-Collar*', '*Lapel-Collar*', '*Stand-Lapel-Collar*', '*Tailor-Collar*', '*Falbala-Collar*', '*Cowl-neck-Collar*', and '*Hat-Collar*'. We apply the random forest algorithm [Breiman 2001] to solve this classification problem. The random forest is trained from the labeled polygon regions in the training image database. Specifically, we normalize the size of each patch in the training set to 200×200, and extract HOG [Dalal and Triggs 2005] and LBP [Wang et al. 2009] features on each normalized patch to train the random forest classifier. If a '*Collar*' component is detected in the reference image, we use the trained random forest classifier to determine its '*Type*' attribute.

For some attributes, including '*Length*' of '*Sleeve*', '*Body*', '*Skirt*' or '*Trousers*', it is more accurate to recover them from the 3D scan directly instead of looking at the RGB image. We classify these attributes using their relative length with respect to the 3D skeleton of the model directly, as shown in Figure 2(b).

Network Optimization. The individually classified attributes might be incorrect. For example, in Figure 2(b), the attribute '*Ease*' of '*Sleeve*' is wrong. We further adopt the context constraint across multiple components to jointly optimize the classification results.

This is motivated by the observation that costume designers usually follow some guidelines when choosing a combination of different attributes. So we learn these contextual constraints from labeled images, and use them to jointly optimize all components.

We follow [Chaudhuri et al. 2011; Kalogerakis et al. 2012] to build a Bayesian network that captures the contextual prior from the labeled image database. Each vertex $A_i, 1 \leq i \leq 18$, in the network is a garment attribute. A directed edge from A_j to A_i encodes the conditional probability $p(A_i|A_j)$ between A_i and A_j. We use the labeled garment image database to learn the Bayesian network (including its structure and parameters) by the PNL (Probabilistic Networking Library) [Intel 2013]. Once the network is learned, we can evaluate the compatibility $p(A_i|\{A_{j_1}, A_{j_2}, \cdots\})$ between A_i and any set of attributes $\{A_{j_1}, A_{j_2}, \cdots\}$.

We then maximize the following energy function

$$\prod_{i=1}^{L} p(A_i = a_i)^{\lambda_1} p(A_i = a_i|\{A_j = a_j|j \neq i\})^{\lambda_2} \qquad (3)$$

to finalize the attribute values. Here, $L \leq 18$ is the total number of attributes of the detected components. $p(A_i = a_i)$ is the likelihood returned by the random forest. $p(A_i = a_i|\{A_j = a_j|j \neq i\})$ is the compatibility between A_i and all the other attributes. We set $\lambda_1 = 0.7, \lambda_2 = 0.3$ in all our experiments, and use the Genetic Algorithm [MIT and Wall 2007] to find the optimal attribute values $\{a_1^*, ...a_L^*\}$ by maximizing Equation 3. The solution is initialized according to the classification results. Some optimized results are shown in Figure 2(c), 7 and 8. The user can interactively refine the optimization result, though this is rarely needed.

6 Garment Model Assembling

We adopt the principles of 'assembly-based modeling' to stitch deformable templates of identified garment components to create the final 3D model. We use the rough garment mesh created by KinectFusion to guide this process.

Candidate Template Preparation. We first prepare a set of candidate templates $\{T_k^i; i = 1 \cdots m_k\}$ for each component C_k. The candidate templates in principle should have the same attributes as the parsing result. However, this will require a much larger 3D template database covering all kinds of designs. To alleviate this requirement, we form the candidate set by choosing templates with at least one consistent attribute as the parsing result.

All 3D garment templates are designed on a mannequin in standing pose, while the scanned model might be in a different casual pose. To facilitate 3D registration, we deform these templates by matching the skeleton pose of the mannequin to that of the scanned model. At the same time, we require the deformed templates to be close to the initial garment mesh generated by KinectFusion. The selected garment templates are deformed based on the methods in [Bouaziz and Pauly 2013; Brouet et al. 2012] and [Baran and Popović 2007; Choi and Ko 2000].

Template Selection. We maximize the following energy to select the optimal template T_k^* for the detected garment component C_k:

$$\prod_{k=1}^{K} \left[S_k^i + p(T_k^i|\{T_l^j|l \neq k\})^{\lambda} \right], \qquad (4)$$

where S_k^i is the geometric similarity between the deformed template T_k^i and the scanned garment mesh. This similarity is computed by the method in [Shen et al. 2012]. K is the number of detected garment components. λ is set to 0.4 in all our experiments. $p(T_k^i|\{T_l^j|l \neq k\})$ measures the compatibility among different templates, which is computed from the compatibility of their attributes. Specifically, $p(T_k^i|\{T_l^j|l \neq k\})$ is computed as

$$\prod_{n=1}^{N_k} p(T_k^i.a_{k,n}|\{T_l^j.a_{l,m}|m = 1...N_l, l \neq k\}). \qquad (5)$$

Here, N_k, N_l are the number of the attributes of templates T_k^i, T_l^j respectively. $T_k^i.a_{k,n}$ and $T_l^j.a_{l,m}$ are the n-th and m-th attributes of T_k^i and T_l^j respectively. $p(T_k^i.a_{k,n}|\{T_l^j.a_{l,m}|m = 1 \cdots N_l, l \neq k\})$ is the attributes compatibility computed by the Bayesian network (similar to the compatibility in Equation 3). We maximize Equation 4 by enumerating all the combination, obtaining a most suitable template T_k^* for each C_k.

Our system outputs the top 3 suitable templates $\{T_k^{*1}, T_k^{*2}, T_k^{*3}\}$ for each C_k for the user's interactive selection (which is rarely needed), as shown in Figure 9.

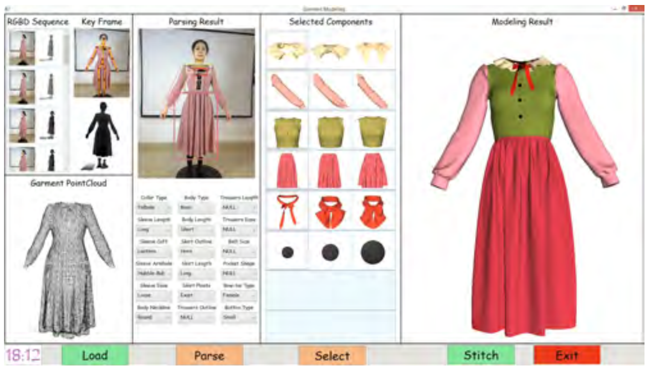

Figure 9: *The interface of our system.*

Template Assembly. We stitch the selected templates together by iteratively assembling garment components in the following order: '*Body*' (or '*Trousers*'), '*Skirt*', '*Sleeve*', '*Bow-tie*', '*Collar*', '*Pocket*', '*Button*'. The '*Belt*' is not assembled, since it is usually an isolated shape.

We put selected templates at their predefined canonical positions (with respect to a human skeleton), and merge the corresponding vertices in their 'slot seams' to seamlessly stitch these templates. The components '*Pocket*' and '*Button*' have no corresponding 'slot seams' on '*Body*'. So we merge their 'slot seam' vertices with their nearest vertices in '*Body*'. To facilitate stitching, the 3D templates are deformed by the method in [Sumner and Popović 2004] with additional constraints that minimizes the Euclidean distance between corresponding vertices in 'slot seams'. Some template assembly results are shown in Figure 2(d) and 9.

7 Results and Discussion

Figure 1 and 10 show the modeling results of our system. We evaluated our method on 44 different garments in total. As demonstrated by these examples, our system can deal with a wide variety of different garment styles including dress (e.g. Figure 1(left) and Figure 10(c,d)), coat (e.g. Figure 10(e,f)), shirt (e.g. Figure 10(g,h)), suits (e.g. Figure 10(i,j)), overcoat (e.g. Figure 10(a,b)), and pants (e.g. Figure 1(right) and Figure 10(k,l)). More examples are in shown in Figure 14 and our supplementary video.

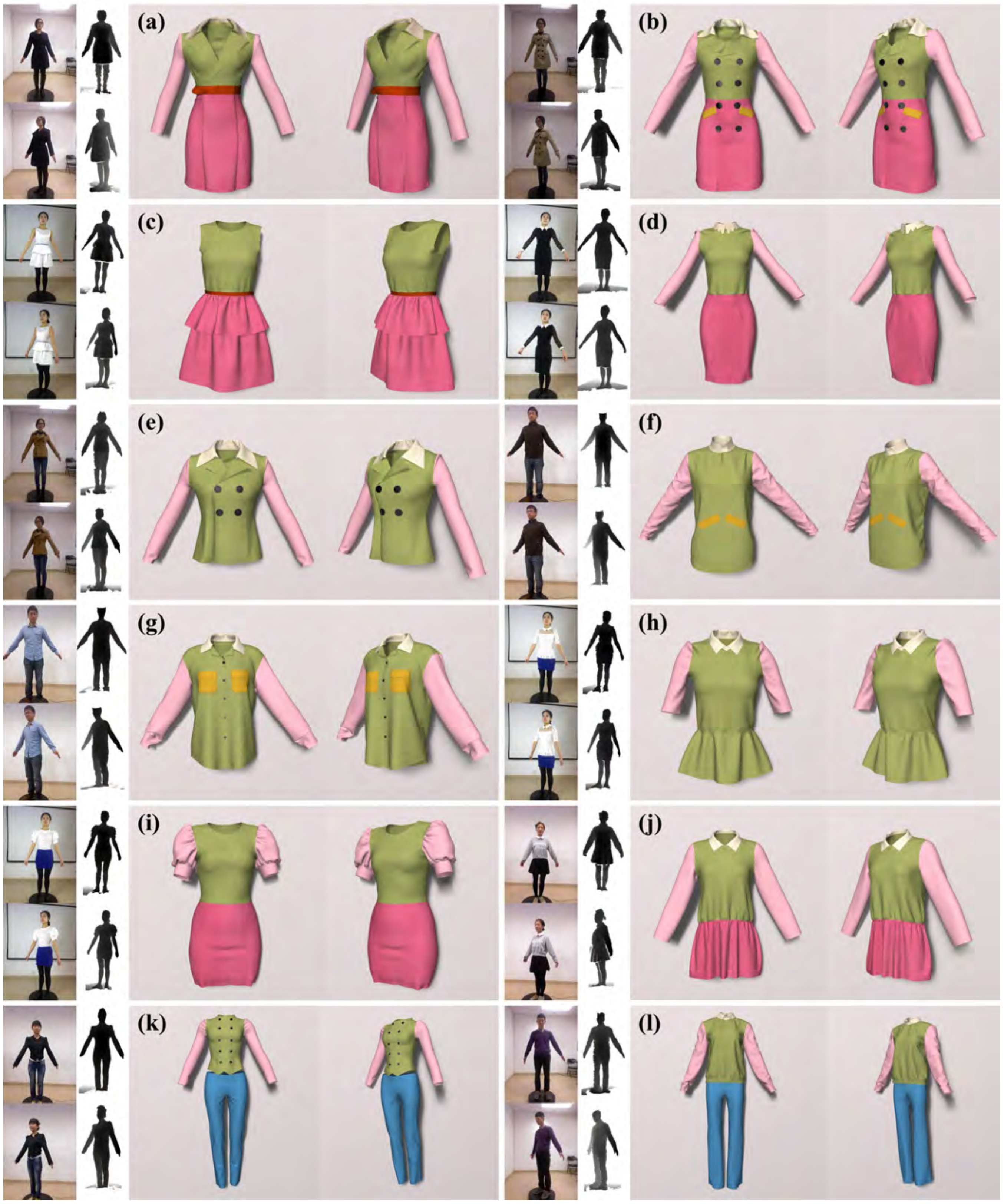

Figure 10: *High quality 3D garment models generated by our system. From left to right of each example: the input RGBD sequence, the two views of the generated 3D model. Each color in the models denotes a different component.*

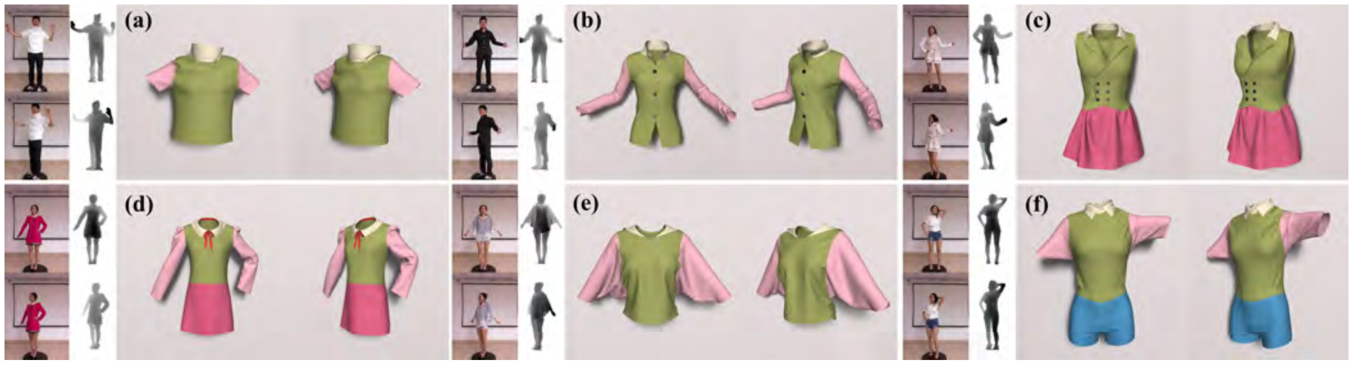

Figure 11: *Garment modeling results with models in different poses.*

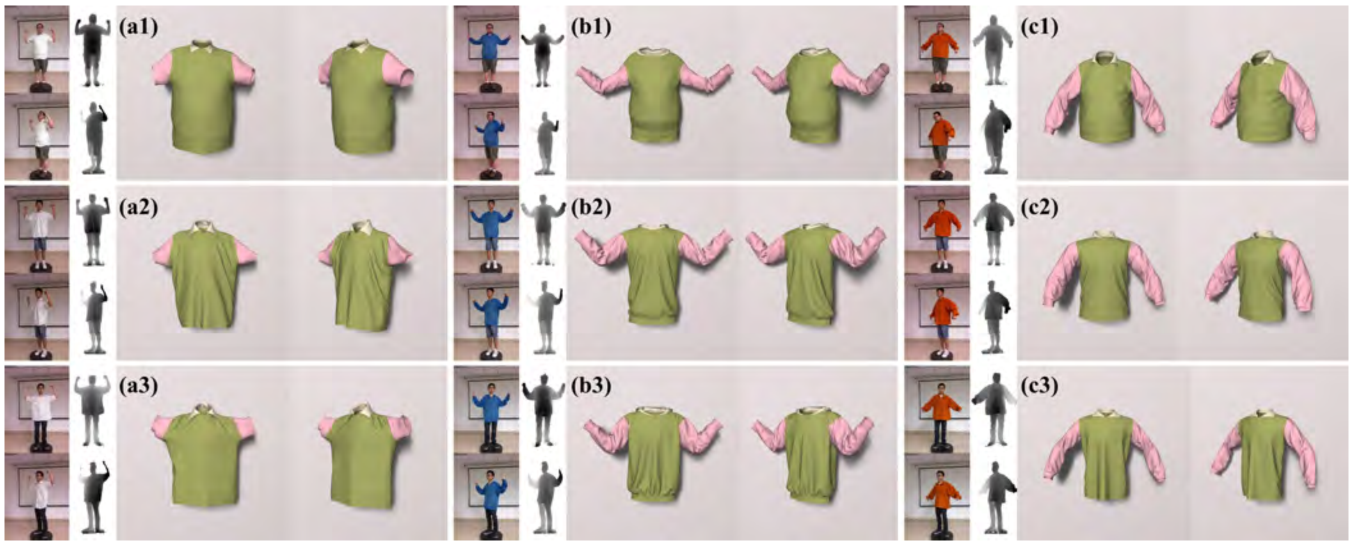

Figure 12: *Results of three different garments (white, blue and red) captured on models with different body shape.*

The generated 3D garment models are of high quality. As demonstrated in the various examples, our system can detect the accessory components correctly, such as '*Pocket*', '*Belt*', '*Bow*', '*Collar*' and '*Button*'. Also our system can classify different attributes correctly, such as *'Falbala Collar'*(e.g. Figure 1(left)), *'Tailor Collar'*(e.g. Figure 10(a,b,e)), *'Hubble-Bubble Sleeve'*(e.g. Figure 1(left), Figure 10(h,i,k)), *'Straight Shape Sleeve'* (e.g. Figure 1(right) and Figure 10(a)), *'Slant Pocket'* (e.g. Figure 10(b,f)), *'Horn Skirt'* (e.g. Figure 10(c,j)), and *'Lantern Skirt'* (e.g. Figure 10(d,i)).

Our system is robust to different poses as shown in Figure 11. The generated result also fits the model's body shape very well as shown in Figure 12, thanks to the template deformation. The template deformation and stitching steps also perform well. It deals garments of multiple layers (e.g. the layer between the collar and body, and the layer in the skirt in Figure 10(c)), which often requires collision detection to avoid self-intersection. Note for some suits (e.g. Figure 1(right), Figure 10(i,j,k,l), Figure 11(f) and Figure 14(g,i)), we create the top garment and the bottom garment independently, and manually place them together in order without merging.

Quantitative Analysis on Garment Parsing. We evaluate our garment component detection and design attribute classification with the labeled database. Table 1 and Table 2 show the accuracy of our garment component detection and design attribute classification respectively. The number of testing and training images are also indicated. The garment component detection is significantly improved by the various priors. Averagely speaking, the joint optimization of the Bayesian network improve the attribute classification accuracy slightly by 2.47%. We also manually check the number of incorrectly classified attributes in all the 44 garments scanned in this paper. Before the Bayesian network optimization, it makes one mistake on 14 garments, and two mistakes on 2 garments. The Bayesian network optimization reduces this to one mistake on 5 garments, and zero mistake on all others. Averagely speaking, our garments include 9 attributes. Some detection and classification results are visualized in Figure 7 and Figure 8 for the 44 captured garments and Internet images respectively.

Table 1: *The precision and recall of our garment component detection with priors and without priors.*

	Training	Testing	Precision		Recall	
			w/o	with	w/o	with
Collar	1800	1082	76.25%	91.04%	73.94%	87.25%
Belt	514	278	84.17%	96.04%	87.77%	88.49%
Pocket	937	599	50.92%	75.29%	58.93%	70.45%
Bow-tie	280	200	78.50%	85.50%	75.00%	81.00%
Button	537	395	63.29%	84.05%	86.84%	88.86%

Computation. All examples are done with a 64 bit desktop machine with a 3.6 GHz Intel Core (TM) i7 processor, 16.0 GB of

Table 2: *The accuracy of design attribute classification with and without Bayesian network optimization.*

Components	Attributes	Training	Testing	w/o opt.	with opt.
Collar	*Type*	2162	721	87.38%	92.23%
Sleeve	*Length*	3394	1132	96.64%	96.64%
	Cuff			81.80%	83.30%
	Armhole			82.69%	84.19%
	Ease			76.33%	79.33%
Body	*Neckline*	3853	1285	84.36%	90.19%
	Type			84.90%	85.91%
	Length			97.20%	97.20%
Skirt	*Outline*	1366	455	83.96%	85.71%
	Length			98.90%	98.90%
	Pleats			92.53%	93.63%
Trousers	*Outline*	862	288	86.46%	87.50%
	Length			98.96%	98.96%
	Ease			84.38%	85.07%
Belt	*Type*	594	198	90.91%	96.97%
Pocket	*Shpae*	1152	384	86.20%	92.45%
Bow-tie	*Type*	360	120	94.17%	96.67%
Button	*Size*	699	233	87.98%	95.28%

memory without special optimization such as GPU acceleration. For typical causal garments with about 17K triangles, our system takes about 13.1 minutes to generate the results. Among this 13.1 minutes, about 2.1 minutes are spent on pre-processing (including KinectFusion, garment image segmentation and skeleton labeling), 0.2 minutes on component detection and attribute classification, 10.8 minutes on template selection and stitching (specifically, 7.7 minutes on template deformation, 0.2 minutes on template selection, 2.9 minutes on template stitching).

User Interaction. In the whole system, the user interaction includes scanning the garment by Kinect, garment segmentation from the reference image, and optional interactive selection of garment component.

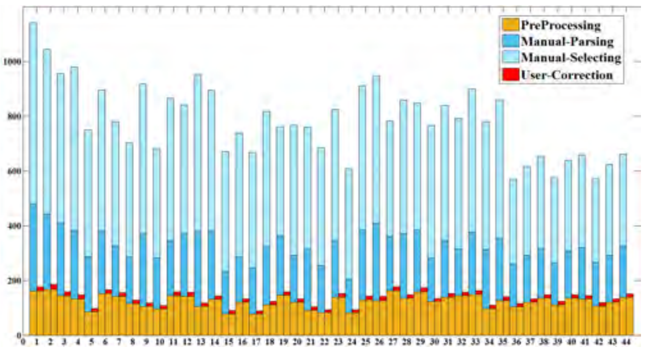

Figure 13: *User interaction time on all 44 garments for manual modeling and our automatic modeling.*

We invite 20 volunteers to evaluate the effectiveness of our garment modeling system. We first ask them to build garment models by manually parsing the reference garment image and selecting suitable templates. Note, we don't let the user directly select templates from the 3D database, since the database is big and direct selection is too tedious. (For example, there are 330 and 250 templates for *'Body'* and *'Sleeve'* respectively.) The manual parsing let the user select the garment attributes manually. The manual selecting let the user choose from a subset of templates with consistent attributes. For a comparison, they also use our system to automatically parse and select templates for modeling, and interactively correct parsing errors when necessary. Both manual and automatic modeling require pre-processing which includes KinectFusion, garment image segmentation and skeleton labeling. We record the user interaction time for manual modeling (*i.e.* pre-processing, manual parsing and manual template selection) and our automatic modeling (*i.e.* pre-processing and interactive correction of parsing errors). Figure 13 shows the comparison of user interaction time for manual and automatic modeling.

The average user interaction time for manual modeling is 12.8 minutes per garment. Specifically, for each garment, the user spent 2.1 minutes on pre-processing, 3.4 minutes on parsing, and 7.3 minutes on template selection. In comparison, the average user interaction time for our automatic modeling system is only 2.3 minutes, including 2.1 minutes in pre-processing and 0.2 minutes in correcting parsing errors. In addition, our automatic parsing achieves comparable accuracy with manual parsing conducted by professional volunteers with strong domain knowledge.

The scanned 3D points are used to accurately determine attributes related to length and size, such as the '*Length*' of '*Sleeve*' and '*Trousers*'. Also, the 3D points contribute a lot to the automatic deformation and selection of garment component templates We empirically found more than 80% computation is spent on processing the scanned 3D points, including KinectFusion, template selection and stitching. In the future, a GPU acceleration could be applied to make our system stronger.

8 Conclusion

We present an intuitive system for garment modeling with only a single depth camera. Given the RGBD sequence of a target garment, our system produces a high quality garment model by selecting and stitching the appropriate component templates from an expert-designed 3D database. We train component detectors and attribute classifiers from a large manually labeled garment image database. These detectors and classifiers build in strong prior knowledge of the garment structure to achieve good performance. Their results are jointly analyzed in a Bayesian network, and are used to guide the selection of templates. With our system, a casual user can quickly generate detailed 3D garment models.

Limitations. Our system has a number of limitations, which point out the direction of future study. Firstly, it does not produce a precise scanning of the target garment. Instead, it outputs a plausible garment model with similar semantic structure and shape. Our work assumes clear compositions of garment components, which is the case of many real garments. We cannot handle clothes of multi-layers, such as a shirt over an under-shirt, because of too much occlusions. Secondly, our current work focus on modeling the geometry shape and does not capture the cloth texture patterns. Thirdly, our work does not capture the cloth deformation and animation parameters.

Acknowledgements

We would like to thank the anonymous reviewers for their valuable comments and suggestions. We are also grateful to Qiang Fu, Faming Li and Yu Zhang for helpful discussions, Jiafan Li for image crawling, Canning Yang for 3D template design, and Xianning Meng and Qi Zhang for texture mapping and model rendering. Thanks also go to Marvelous Designer for their assistance, and Taobao for making their garment images available. This work was partially supported by NSFC (61325011&61502023&61532003), 863 Program (2013AA013801), and SRFDP (20131102130002). Ping Tan is supported by the Canada NSERC Discovery (611664) and Discovery Acceleration Supplement (611663).

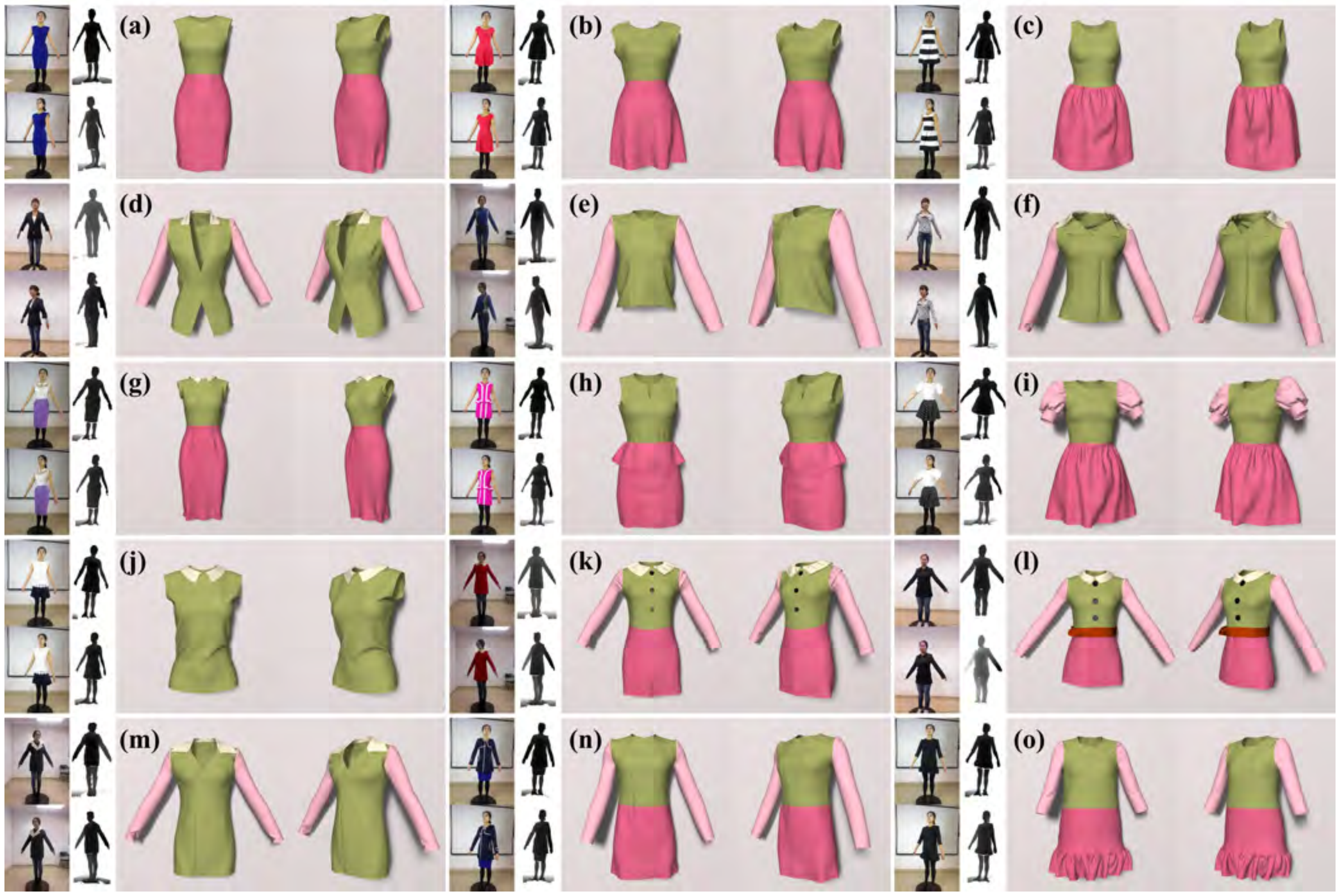

Figure 14: *More Results. From left to right of each example: the input RGBD sequence, the two views of the generated 3D model. Each color in the models denotes a different component.*

References

BARAN, I., AND POPOVIĆ, J. 2007. Automatic rigging and animation of 3d characters. *ACM Trans. Graph. 26*, 3, 72:1–72:8.

BERTHOUZOZ, F., GARG, A., KAUFMAN, D., GRINSPUN, E., AND AGRAWALA, M. 2013. Parsing sewing patterns into 3d garments. *ACM Trans. Graph. 32*, 4, 85:1–85:12.

BOUAZIZ, S., AND PAULY, M., 2013. Dynamic 2d/3d registration for the kinect. ACM SIGGRAPH Courses.

BRADLEY, D., POPA, T., SHEFFER, A., HEIDRICH, W., AND BOUBEKEUR, T. 2008. Markerless garment capture. *ACM Trans. Graph. 27*, 3, 99:1–99:9.

BREIMAN, L. 2001. Random forests. *Mach. Learn. 45*, 1, 5–32.

BROUET, R., SHEFFER, A., BOISSIEUX, L., AND CANI, M.-P. 2012. Design preserving garment transfer. *ACM Trans. Graph. 31*, 4, 36:1–36:11.

CHAUDHURI, S., AND KOLTUN, V. 2010. Data-driven suggestions for creativity support in 3d modeling. *ACM Trans. Graph. 29*, 6, 183:1–183:9.

CHAUDHURI, S., KALOGERAKIS, E., GUIBAS, L., AND KOLTUN, V. 2011. Probabilistic reasoning for assembly-based 3d modeling. *ACM Trans. Graph. 30*, 4, 35:1–35:9.

CHEN, K., LAI, Y.-K., WU, Y.-X., MARTIN, R., AND HU, S.-M. 2014. Automatic semantic modeling of indoor scenes from low-quality rgb-d data using contextual information. *ACM Trans. Graph. 33*, 6, 208:1–208:12.

CHOI, K.-J., AND KO, H.-S. 2000. Online motion retargetting. *The Journal of Visualization and Computer Animation (Special issue of Pacific Graphics '99) 11*, 5, 223–235.

CORDIER, F., SEO, H., AND MAGNENAT-THALMANN, N. 2003. Made-to-measure technologies for online clothing store. *Comput. Graph. Appl. 23*, 1.

DALAL, N., AND TRIGGS, B. 2005. Histograms of oriented gradients for human detection. In *Proc. CVPR*, 886–893.

DECAUDIN, P., JULIUS, D., WITHER, J., BOISSIEUX, L., SHEFFER, A., AND CANI, M.-P. 2006. Virtual garments: A fully geometric approach for clothing design. *Comput. Graph. Forum 25*, 3, 625–634.

DIVIVIER, A., TRIEB, R., EBERT, A., AND ETC. 2004. Virtual try-on: Topics in realistic, individualized dressing in virtual reality. In *Proc. Virtual and Augmented Reality Status Conf.*

DOU, M., TAYLOR, J., FUCHS, H., FITZGIBBON, A., AND IZADI, S. 2015. 3d scanning deformable objects with a single rgbd sensor. In *Proc. CVPR*, 493–501.

FELZENSZWALB, P. F., GIRSHICK, R. B., MCALLESTER, D., AND RAMANAN, D. 2010. Object detection with discriminatively trained part based models. *IEEE Trans. Pattern Anal. Mach. Intell. 32*, 9, 1627–1645.

FISHER, M., RITCHIE, D., SAVVA, M., FUNKHOUSER, T., AND HANRAHAN, P. 2012. Example-based synthesis of 3d object arrangements. *ACM Trans. Graph. 31*, 6, 135:1–135:11.

FONTANA, M., CARUBELLI, A., RIZZI, C., AND CUGINI, U. 2005. Clothassembler: a cad module for feature-based garment pattern assembly. *Computer-Aided Design and Applications 2*, 6, 795–804.

FUNKHOUSER, T., KAZHDAN, M., SHILANE, P., MIN, P., KIEFER, W., TAL, A., RUSINKIEWICZ, S., AND DOBKIN, D. 2004. Modeling by example. *ACM Trans. Graph. 23*, 3, 652–663.

GIRSHICK, R. B., FELZENSZWALB, P. F., AND MCALLESTER, D., 2012. Discriminatively trained deformable part models release 5. http://people.cs.uchicago.edu/ rbg/latent-release5/.

GUAN, P., REISS, L., HIRSHBERG, D. A., WEISS, A., AND BLACK, M. J. 2012. Drape: Dressing any person. *ACM Trans. Graph. 31*, 4, 35:1–35:10.

INGHAM, R., AND COVEY, L. 1992. *Costume Designer's Handbook: A Complete Guide for Amateur and Professional Costume Designers*. Prentice-Hall Inc.

INTEL, 2013. The open source probabilistic networks library. http://sourceforge.net/projects/openpnl/.

IZADI, S., KIM, D., HILLIGES, O., MOLYNEAUX, D., NEWCOMBE, R., KOHLI, P., SHOTTON, J., HODGES, S., FREEMAN, D., DAVISON, A., AND FITZGIBBON, A. 2011. Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. In *Proc. ACM Symposium on User Interface Software and Technology*, 559–568.

KALOGERAKIS, E., CHAUDHURI, S., KOLLER, D., AND KOLTUN, V. 2012. A probabilistic model for component-based shape synthesis. *ACM Trans. Graph. 31*, 4, 55:1–55:11.

LI, Y., SUN, J., TANG, C.-K., AND SHUM, H.-Y. 2004. Lazy snapping. *ACM Trans. Graph. 23*, 3, 303–308.

LI, H., VOUGA, E., GUDYM, A., LUO, L., BARRON, J. T., AND GUSEV, G. 2013. 3d self-portraits. *ACM Trans. Graph. 32*, 6, 187:1–187:9.

LIU, S., SONG, Z., LIU, G., XU, C., LU, H., AND YAN, S. 2012. Street-to-shop: Cross-scenario clothing retrieval via parts alignment and auxiliary set. In *Proc. CVPR*, 3330–3337.

MENG, Y., MOK, P., AND JIN, X. 2012. Computer aided clothing pattern design with 3d editing and pattern alteration. *Computer-Aided Design 44*, 8, 721–734.

MIT, AND WALL, M., 2007. Galib: Matthew's genetic algorithms library. http://lancet.mit.edu/ga/.

MOK, P., XU, J., WANG, X., FAN, J., KWOK, Y., AND XIN, J. H. 2013. An iga-based design support system for realistic and practical fashion designs. *Computer-Aided Design 45*, 11, 1442–1458.

POPA, T., ZHOU, Q., BRADLEY, D., KRAEVOY, V., FU, H., SHEFFER, A., AND HEIDRICH, W. 2009. Wrinkling captured garments using space-time data-driven deformation. *Comput. Graph. Forum 28*, 2, 427–435.

ROBSON, C., MAHARIK, R., SHEFFER, A., AND CARR, N. 2011. Context-aware garment modeling from sketches. *Comput. and Graph. 35*, 3, 604–613.

SCHOLZ, V., STICH, T., KECKEISEN, M., WACKER, M., AND MAGNOR, M. 2005. Garment motion capture using color-coded patterns. *Comput. Graph. Forum 24*, 3, 439–447.

SHAO, T., XU, W., ZHOU, K., WANG, J., LI, D., AND GUO, B. 2012. An interactive approach to semantic modeling of indoor scenes with an rgbd camera. *ACM Trans. Graph. 31*, 6, 136:1–136:11.

SHEN, C.-H., FU, H., CHEN, K., AND HU, S.-M. 2012. Structure recovery by part assembly. *ACM Trans. Graph. 31*, 4, 180:1–180:11.

SUMNER, R. W., AND POPOVIĆ, J. 2004. Deformation transfer for triangle meshes. *ACM Trans. Graph. 23*, 3, 399–405.

TURQUIN, E., CANI, M.-P., AND HUGHES, J. F. 2004. Sketching garments for virtual characters. In *Proc. Eurographics Workshop on Sketch-Based Interfaces and Modeling*, 175–182.

TURQUIN, E., WITHER, J., BOISSIEUX, L., CANI, M.-P., AND HUGHES, J. F. 2007. A sketch-based interface for clothing virtual characters. *Comput. Graph. Appl. 27*, 1, 72–81.

UMETANI, N., KAUFMAN, D. M., IGARASHI, T., AND GRINSPUN, E. 2011. Sensitive couture for interactive garment editing and modeling. *ACM Trans. Graph. 30*, 4, 90:1–90:11.

VOLINO, P., AND MAGNENAT-THALMANN, N. 2005. Accurate garment prototyping and simulation. *Computer-Aided Design and Applications 2*, 5, 645–654.

WANG, C. C., WANG, Y., AND YUEN, M. M. 2003. Feature based 3d garment design through 2d sketches. *Computer-Aided Design 35*, 7, 659–672.

WANG, X., HAN, T. X., AND YAN, S. 2009. An hog-lbp human detector with partial occlusion handling. In *Proc. ICCV*, 32–39.

WHITE, R., CRANE, K., AND FORSYTH, D. A. 2007. Capturing and animating occluded cloth. *ACM Trans. Graph. 26*, 3, 34:1–34:8.

YAMAGUCHI, K., KIAPOUR, M. H., ORTIZ, L. E., AND BERG, T. L. 2012. Parsing clothing in fashion photographs. In *Proc. CVPR*, 3570–3577.

YAMAGUCHI, K., KIAPOUR, M. H., ORTIZ, L. E., AND BERG, T. L. 2015. Retrieving similar styles to parse clothing. *IEEE Trans. Pattern Anal. Mach. Intell. 37*, 5, 1028–1040.

YANG, W., LUO, P., AND LIN, L. 2014. Clothing co-parsing by joint image segmentation and labeling. In *Proc. CVPR*, 3182–3189.

YU, Y., ZHOU, K., XU, D., SHI, X., BAO, H., GUO, B., AND EUNG SHUM, H.-Y. 2004. Mesh editing with poisson-based gradient field manipulation. *ACM Trans. Graph. 23*, 3, 644–651.

ZHOU, B., CHEN, X., FU, Q., GUO, K., AND TAN, P. 2013. Garment modeling from a single image. *Comput. Graph. Forum 32*, 7, 85–91.

ZUFFI, S., AND BLACK, M. J. 2015. The stitched puppet: A graphical model of 3d human shape and pose. In *Proc. CVPR*, 3537–3546.