Learning to Transfer Texture from Clothing Images to 3D Humans

Aymen Mir\textsuperscript{1}  Thiem Alldieck\textsuperscript{1,2}  Gerard Pons-Moll\textsuperscript{1}

\textsuperscript{1}Max Planck Institute for Informatics, Saarland Informatics Campus, Germany
\textsuperscript{2}Computer Graphics Lab, TU Braunschweig, Germany
{amir,gpons}@mpi-inf.mpg.de alldieck@cg.cs.tu-bs.de

Figure 1: Our model Pix2Surf allows to digitally map the texture of online retail store clothing images to the 3D surface of virtual garment items enabling 3D virtual try-on in real-time.

Abstract

In this paper, we present a simple yet effective method to automatically transfer textures of clothing images (front and back) to 3D garments worn on top SMPL \cite{42}, in real time. We first automatically compute training pairs of images with aligned 3D garments using a custom non-rigid 3D to 2D registration method, which is accurate but slow. Using these pairs, we learn a mapping from pixels to the 3D garment surface. Our idea is to learn dense correspondences from garment image silhouettes to a 2D-UV map of a 3D garment surface using shape information alone, completely ignoring texture, which allows us to generalize to the wide range of web images. Several experiments demonstrate that our model is more accurate than widely used baselines such as thin-plate-spline warping and image-to-image translation networks while being orders of magnitude faster. Our model opens the door for applications such as virtual try-on, and allows for generation of 3D humans with varied textures which is necessary for learning. Code will be available at https://virtualhumans.mpi-inf.mpg.de/pix2surf/.

1. Introduction

Our goal is to learn a model capable of transferring texture from two photographs (front and back) of a garment to the 3D geometry of a garment template, automatically and in real time. Such a model can be extremely useful for photo-realistic rendering of humans, varied generation of synthetic data for learning, virtual try-on, art and design. Despite the many applications, automatic transfer of clothing texture to 3D garments has received very little attention. The vast majority of recent methods work in image space directly. Most works focus on either image based person re-posing \cite{82,43,44,56,63,21,10,64}, or virtual try-on \cite{57,78,28,69,77,22,83}. Re-posing methods learn to synthesize image pixels to produce novel poses of a person, whereas virtual try-on methods learn to morph an image of a clothing item to drape it on a target person. The advantage of these methods is that they can be trained on large-scale datasets. The critical disadvantage is that they operate in pixel-space instead of 3D, they can not synthesize difficult poses, and struggle to produce temporally consistent results. Another line of work extracts texture by fitting 3D
human models (SMPL [42]) to images [15, 6, 4], but texture quality quickly deteriorates for complex poses. Other works map texture to 3D meshes [68, 24, 40, 53] by 3D scanning people, but the number and variety of clothing textures is limited because 3D scanning is time consuming. To break the lack of 3D data barrier, our idea is to learn a dense mapping from images of clothing items, which are ubiquitous on the internet, to the 3D surface of a parametric garment template directly [15]. However, accurate texture transfer is far from trivial: web images vary in texture, garment size and style, pose, and background. Nonetheless, in comparison to clothing worn by humans, web-store images have less variation, which we exploit for learning our model. Instead of manually annotating garment landmarks [85, 8], our idea is to collect training pairs by non-rigidly aligning a parametric 3D garment template to images. We leverage the parameterized garment models of MGN [15], non-rigidly fit their surface to image silhouettes. While our alignment usually produces good results, it is slow (5 – 15 minutes per image), and fails in 5% of the cases. Consequently, using only the successful fits, we learn a direct mapping from image pixels which runs in milliseconds, and is more robust than the optimization based approach. Our key hypothesis is that the mapping is determined by the image silhouette shape alone, and not by appearance. Hence, we train a CNN to predict correspondences from a UV map of the garment surface to pixel locations, given the silhouette shape alone as input. Since the model learns in-variances to shape and pose as opposed to appearance, it generalizes to a wide variety of garment images of varied textures. We refer to the correspondence predictor as Pix2Surf. Pix2Surf allows to digitally map the texture of a clothing image to the 3D surface of a garment in real time, see Fig. 1. Once the texture is mapped to 3D, we generalize to novel views, shapes and pose by applying 3D transformations to the 3D geometry [15] as illustrated in Fig. 2. Pix2Surf enables, for the first time, virtual try-on from images in 3D directly and in real time, enabling applications such as VR/AR, gaming, and 3D content production.

To stimulate further research in this direction, we will make Pix2Surf publicly available for research purposes. This will allow researchers and users to add texture to the SMPL+Garments model [15], generate synthetic humans, and visualize garment images in 3D directly.

2. Related Work

Synthesizing images of people in realistic clothing is a long-studied problem in the Computer Graphics community. Early works enable to represent and parametrize fabric virtually in 3D [72, 50], later realistic draping and animation has been achieved [12, 30]. While these works require artist-designed garments, careful parameter selection, and computational expensive physics-based simulation, fully-automated and faster methods have been introduced more recently. In contrast, these methods process sensor data such as images, depth maps, or 3D point clouds, and produce 3D reconstructions or photorealistic images with minimal interaction or even fully automatically.

2D image synthesis methods produce images of people holding a given pose or wearing a desired outfit. These works utilize recent advances in conditional image-to-image translation using generative adversarial networks (GANs) [25, 32]. E.g. [39] presents a method to produce full-body images of people in clothing from semantic segmentation of the body and clothing. However, the produced subject and outfit are not controllable. The method in [82] produces novel views of given fashion images. While the subject and outfit remain untouched, the pose can be constrained with a sparse set of camera angles. To gain more control over the output, a large number of works deal with synthesizing person images under a desired pose [43, 44, 56, 63, 21, 10, 64, 62]. For the same purpose, in [26], the authors learn warpings between images and the SMPL texture. Predicting correspondences instead of color values helps the network to generalize due to much lower variation in the warps than in the color-space. In Pix2Surf we also infer correspondences [61, 54, 8, 48] but focus on garments rather than on full bodies. In similar works, [48] utilizes DensePose [8] to warp image pixels into the SMPL texture, but inpainting is performed before warping into the target pose. More related to our work, recent methods focus on exchanging the subject’s outfit while preserving [57, 78] or changing [28, 69, 77, 22, 83] his or her pose. Additionally, special case methods for text-based fashion image manipulation [84] and image-based outfit improvement [31] have been presented. In contrast to our method, all these virtual try-on methods work in the image-space and thus perform no explicit reasoning about the underlying 3D
scene. This means they are not guaranteed to produce consistent output under varying poses and camera angles. This is approached in [23] via image warping and subsequent refinement of a previously generated image. While 2D warping improves the quality of synthesized videos, limitations of 2D methods are still present.

**3D reconstruction** methods focus on recovering the actual 3D shape of a captured garment alone, the body shape of a subject wearing the garment, or both simultaneously. Methods utilizing controlled RGB [35] and RGB-D images [17] have been presented, that select and refine 3D garment templates based on image observations. While [35] utilizes a tailor’s dummy, [17] expects the subject to hold a certain pose. Other methods focus on recovering the shape and detailed garment wrinkles of clothing item in less controlled settings [55, 20, 13, 36, 38]. While these methods can produce detailed geometry, none of these methods focuses on the appearance of the item. Another branch of research aims at 3D reconstructing the whole human including clothing. This can be achieved by optimization-based [6, 5, 73] or learning-based methods [4, 40] that utilize silhouettes or semantic segmentation of a short monocular video clip or recently even from single images [46, 7, 59, 24] and point-clouds [18]. Other methods utilize Kinect-fusion like approaches [33, 49] to scan people using RGB-D sensors [60, 41, 79, 19]. Having a 3D reconstruction of body and clothing, it can be used to non-rigidly track the subject [74, 27]. All these methods fuse body and clothing in a single template. Virtual try-on applications, however, often require separate meshes [29]. Therefore, methods that reconstruct the naked body shape or both body shape and clothing have been developed. The naked body shape alone has been estimated using several RGB images [11] or more accurately using a sequence of clothed scans [80]. Body shape and garments have been reconstructed simultaneously and fully-automatically from a series of scans [53], RGB-D images [66], and recently even from a small set of RGB images [15]. In [76] the authors present garment and body-shape recovery from a single image but heavily rely on physical priors and human interaction. In order to enable dynamic virtual try-on and clothing re-targeting, joint models of clothing and the human body have been developed [47, 75, 52, 45]. Again, all these works focus mainly or exclusively on the geometry of the garment, not on their appearance. Other works also learn to predict correspondences from depth maps to surfaces [67, 54, 71], image to surfaces [37, 8, 26], but they all address different problems.

**Automatic texturing** of 3D models from photos has been presented too, but the shape has to be first aligned with the input image [51, 70]. This alignment is expensive and error-prone as silhouette and feature cues may be ambiguous. The most related work here is [16], which maps texture from clothing items to the 3D SCAPE [9] body model. Their focus is not photo-realistic mapping, but rather to generate synthetic training data with texture variation to learn discriminative detectors. Their core texture mapping is based on 2D image warping – unfortunately, the code for this specific part is not available, and therefore comparison is not possible. However, qualitatively, our results look significantly more realistic, and we compare to a very similar baseline based on shape context (SC) matching and Thin Plate Spline (TPS) warping. Furthermore, our approach runs in real time. In contrast to all previous work, our method creates textured 3D garments fully automatically, without requiring prior alignment at test-time, which allows virtual try-on and novel view synthesis in real-time.

### 3. Method

Our key idea is to learn a mapping from images to the UV map of the garment, without using texture information, but silhouette shape alone. We first explain the parametric 3D garment models (Sec. 3.1) we use to regularize an automatic mesh to image silhouette fitting procedure (Sec. 3.2). Since fitting is expensive and error-prone, we learn an efficient neural mapping (Pix2Surf), which transfers the image texture onto the mesh in real time (Sec. 3.3).

#### 3.1. Preliminary: Parametric Garment Model

We leverage publicly available 3D garments templates [15] parameterized as displacements from the SMPL body model [42]. For every garment category (T-shirt, short pants, long pants), we define a garment template \( T^G \in \mathbb{R}^{m \times 3} \) as a sub-mesh of the SMPL body template \( T \in \mathbb{R}^{n \times 3} \). An indicator matrix \( I \in \mathbb{Z}^{m \times n} \) evaluates to \( I_{i,j} = 1 \) if a garment vertex \( i \in \{1 \ldots m\} \) is associated with a body shape vertex \( j \in \{1 \ldots n\} \). This correspondence allows representing garments as displacements \( D \in \mathbb{R}^{m \times 3} \) from the unposed SMPL body. Given shape \( \beta \) and pose \( \theta \), we can articulate a garment using SMPL:

\[
T^G(\beta, \theta, D) = I T(\beta, \theta) + D \tag{1}
\]

\[
G(\beta, \theta, D) = W(T^G(\beta, \theta, D), J(\beta), \theta, W), \tag{2}
\]

with joints \( J(\beta) \) and linear blend-skinning \( W(\cdot) \) with weights \( W \). Since in this work we keep \( D \) fixed, we denote the garment model as \( G(\theta, \beta) \). After the texture has been transferred to the surface, the geometry can be still be changed with \( D \).

#### 3.2. Non-Rigid Garment Fitting to Retail Images

To find a correspondence map between retail images in the web and 3D garments, we could non-rigidly deform its 3D surface to fit the image foreground. This is, however, not robust enough as different retailers photograph garments in different poses, backgrounds, and clothing itself varies in
Figure 3. Overview of method: We build a large paired dataset of 3D garments and online retail store images by non-rigidly aligning 3D garment templates with automatically computed image segmentations (I). This allows us to train our model Pix2Surf (II) to predict correspon- dences from the UV map of the garment surfaces to image pixel locations.

shape. Hence, we first automatically segment images, and then leverage the parametric garment model $G(\theta, \beta)$ defined in Sec. 3.1 to regularize the fitting process.

3.2.1 Automatic segmentation

We use an automated version of GrabCut [58]. Since garments are typically photographed over simple backgrounds, we obtain an approximate foreground mask using thresholding. We then run a closing operation to fill the holes on this mask, and erode it to obtain a prior for “absolute foreground”. The difference region between the mask and its eroded version is marked as “probable foreground”. Analogously, we obtain “absolute background”, and “probable background” using dilation. Using these prior maps to initialize GrabCut, we obtain accurate silhouettes without manual annotation.

3.2.2 Garment fitting

We fit the garment surface to silhouettes in two stages. In the first stage, we minimize the following objective

$$E_1(\beta, \theta, t) = w_s E_s + w_\beta E_\beta + w_\theta E_\theta,$$  \hspace{1cm} (3)

w.r.t. garment pose, shape and camera translation $t \in \mathbb{R}^3$. The objective in Eq. (3) consists of a silhouette $E_s$, a shape regularization term $E_\beta$ and a pose prior term $E_\theta$, which we explain in the following. The different terms are balanced using weights $w_s$.

Silhouette term: It is defined as:

$$E_s(\beta, \theta, t) = \Phi(w_i \Psi(I_r(G(\theta, \beta), t)) + w_o \hat{\Psi} (1 - I_r(G(\theta, \beta), t))).$$  \hspace{1cm} (4)

Here, $\Psi$ and $\hat{\Psi}$ are the distance transform, and the inverse distance transform, respectively, of the silhouette image, $\Phi$ is a Gaussian pyramid function, and $I_r(G(\theta, \beta), t))$ is the binary garment silhouette image obtained with a differentiable renderer. Consequently, the objective in Eq. (4) maximizes overlap between the garment image and the rendered mesh, and penalizes model leackage into the background.

Shape regularization: In order to regularize the fitting process in Eq. 3, we use a Mahalanobis prior

$$E_\beta(\beta) = \beta^T \Sigma^{-1}_\beta \beta$$  \hspace{1cm} (5)

on the shape parameters, where $\Sigma^{-1}_\beta$ is the diagonal covariance matrix from the SMPL dataset.

Pose prior: The term penalizes deviations of the pose from an A-pose $\theta_A$

$$E_\theta(\theta) = \|\theta - \theta_A\|^2.$$  \hspace{1cm} (6)

To minimize Eq. (3), we initialize the pose $\theta$ with an A-pose, as this approximates the pose of most garment images on the web. Additionally, we use scheduling: for shirts we first optimize shape and translation holding pose fixed and optimize all variables jointly afterwards. For pants and shorts the scheduling order is reversed. Stage 1 provides us with a coarse match to the silhouette of the garment, but the final mesh is restricted by the parametric model $G(\theta, \beta, t)$. To perfectly match silhouette boundaries, we non-rigidly
deform free form vertices $G \in \mathbb{R}^{m \times 3}$ of a mesh initialized with the optimized garment result of the first stage $G(\theta, \beta)$. Specifically, we optimize a more sophisticated version of Eq. 3:

$$E_2(G, \theta, \beta) = w_r E'_r + w_c E'_c + w_s E'_s + w_i E'_i.$$  

(7)

$E'_s$ is the same as in Eq. (4), but now we optimize the free form vertices $G$ instead of the latent parameters of the model $G(\theta, \beta)$, and $E'_c$, $E'_i$, and $E'_s$ are coupling, laplacian, edge constraint, and boundary smoothing terms, which we explain next.

**Coupling term:** It penalizes deviations of the free form vertices $G$ from the parametric garment model $G(\theta, \beta)$:

$$E_c(G, \theta, \beta) = \|G - G(\theta, \beta)\|^2$$  

(8)

**Edge Constraint Term:** The belt or waistline in shorts and pants retail images forms an almost perfect horizontal line. We exploit this by matching the top ring (waistline) of pants and shorts 3D meshes to this horizontal line in image space. Let $G_i \in \mathcal{R}$ denote the set of top ring vertices, $\pi(G_i)_y$ denote the y coordinate of vertex $G_i$ after projection $\pi(\cdot)$ onto the image, and let $y_{max}$ denote the y coordinate of the horizontal line in the image. We incorporate the following penalty:

$$E_c(G) = \sum_{G_i \in \mathcal{R}} \|\pi(G_i)_y - y_{max}\|^2$$  

(9)

**Laplacian Term:** In order enforce garment smoothness and minimize distortion, we include a Laplacian term [65]. Given a mesh with adjacency matrix $A \in \mathbb{R}^{m \times m}$, the graph Laplacian $L \in \mathbb{R}^{m \times m}$ is obtained as $L = I - K^{-1}A$ where $K$ is a diagonal matrix such that $K_{ii}$ stores the number of neighbors of vertex $i$. We minimize the mesh Laplacian:

$$E_L(G) = \|LG\|_p^2$$  

(10)

**Boundary Smoothing Term:** To ensure that the boundaries remain smooth, we penalize high second order derivatives along the boundary rings, similar to [53]. The output of the fitting are 3D garment vertices $G$, which together with their faces $F$ define a deformed mesh $\mathcal{G} = \{G, F\}$ accurately aligned with image silhouette.

3.3. Learning Automatic Texture Transfer

The fitting of the previous section is accurate, but slow, and fails sometimes. Hence, we run the fitting method on internet images, and manually remove the unsuccessful fits. From this data, we train an efficient neural model, referred to as Pix2Surf. Pix2Surf directly transfers texture from images to the 3D model surface, based on the silhouette shape alone. Next, we explain the key components of Pix2Surf, namely, input output representation, and losses used during training.

### 3.3.1 Pix2Surf: Input and Output Representation

The curated fits of Sec. 3.2 provide dense correspondences from image pixels $(i, j)$ to the 3D garment surface $G \subset \mathbb{R}^3$. Learning a mapping to the surface $G$ embedded in $\mathbb{R}^3$ is hard, and does not allow leveraging fully 2D convolutional neural networks. Hence, we compute a 2D UV-map parameterization (Sec. 3.3.3) of the garment surface, $u : \mathcal{B} \subset \mathbb{R}^2 \rightarrow G \subset \mathbb{R}^3$, where $u(\cdot)$ maps 2D points $(k, l)$ from the UV space to the surface $G$ embedded in $\mathbb{R}^3$. In this way, all our targets live in a UV-space, specifically from the fits, we generate: RGB texture maps $Y \in \mathbb{R}^{K \times L \times 3}$ using projective texturing, and UV correspondence maps $C \in C \subset \mathbb{R}^{K \times L \times 2}$, which store, at every pixel $(k, l)$ of the (front/back) UV map, the $(i, j)$ coordinates of the corresponding image pixel, that is $C_{k,l} = (i, j)$. The input representation for the garment images is a coordinate mask $X \in \mathcal{X} \subset \mathbb{R}^{M \times N \times 2}$, storing at every image pixel location its own coordinates if the pixel belongs to the foreground $\mathcal{F}$, and 0 otherwise, $X_{ij} = (i, j) \forall (i, j) \in \mathcal{F}$, $X_{ij} = (0, 0) \forall (i, j) \notin \mathcal{F}$. The foreground mask $\mathcal{F}$ is predicted at test time using a segmentation network trained using a standard cross-entropy loss—we compute segmentation labels for training using the automatic – but slow – GrabCut based method Sec. 3.2.1. For the front view of T-shirts, we additionally ask annotators to segment out the back portion of the shirt, which is visible in the front view image. The segmentation network learns to remove this portion ensuring that the back portion is not mapped to the UV-map. With this we collect a dataset $D$ consisting of inputs $X$, and targets $\{Y, C\}$. $D = \{X^i, \{Y^i, C^i\}\}_{i=1}^N$.

### 3.3.2 Pix2Surf: Learning

We could naively attempt to predict the texture maps $Y$ directly from images $I$ using image to image translation, but this is prone to overfit to image textures as we demonstrate in our experiments (Sec. 4.2). Instead, we follow a more geometric approach, and learn a mapping $f(X; w) : \mathcal{X} \rightarrow \mathcal{C}$ from coordinate masks $X$ to UV correspondence maps $C$, forcing the network to reason about the input shape. This effectively, learns to predict, for every UV map location $(k, l)$, the corresponding pixel coordinates $(i, j)$ in the image, $C_{k,l} = (i, j)$. Our key insight and assumption is that this smooth mapping depends only on silhouette shape $X$, and not on texture $I$. During training, we minimize the following loss.

$$L_{\text{total}} = \lambda_{\text{reg}} L_{\text{reg}} + \lambda_{\text{perc}} L_{\text{perc}} + \lambda_{\text{recon}} L_{\text{recon}}$$  

(11)

over the training set $D$, where each term in the loss is explained next. **Coordinate Regression Loss:** $L_{\text{reg}}$ evaluates an $L_2(\cdot)$ norm of the difference between the network output
$f(X^i; w)$ and the pseudo ground truths $C$ obtained using
the silhouette fitting algorithm of Sec. 3.2:

$$L_{reg} = \sum_{i=1}^{N} || f(X^i; w) - C^i ||^2_2$$  \quad (12)

Reconstruction Loss: To provide more supervision to the
network we use a differentiable sampling kernel [34] to
infer a texture map directly from the correspondence map.
We minimize a dense photometric loss between predicted
texture maps and target texture maps $Y$ obtained with pro-
jective texturing (Sec. 3.3.1):

$$L_{recon} = \sum_{i}^{N} \sum_{k,l} || I[f_{k,l}^1(X^i; w), f_{k,l}^2(X^i; w)] - Y_{k,l}^i ||_1$$  \quad (13)

where the original image $I$ is sampled (using a differentiable
kernel) at locations $(i,j) = (f_{k,l}^1(X^i; w), f_{k,l}^2(X^i; w))$ provided
by the predicted correspondence map.

Perceptual Loss: $L_{perc}$ is the perceptual loss as defined
in [81] between $I[f(X; w)] \in \mathbb{R}^{K \times L \times 3}$ (tensor notation)
and $Y$.

Once the network predicts a correspondence map for an in-
put image, we use it and the parallelizable kernel to generate
the final image by sampling points from the input image.

3.3.3 Implementation Details

We use Adam optimizer for training both networks. For the
segmentation network we use a UNet with instance normal-
ization and use color jittering in the input data to improve
performance. For Pix2Surf we use a six block ResNet. The
choice of normalization and activation functions is same as [32].

Custom UV Map: Since the artist designed SMPL UV map
cuts the garments into different islands (bad for learning a
continuous mapping), we use a custom UV map for each
garment category. We cut the garment surface into front and
back and compute the UV map using Blender This results
in two islands (front and back), which makes the image to
UV mapping continuous and hence easier to learn.

4. Experiments

Since we are solving a problem for which there is no
ground-truth data, we evaluate our proposed method quali-
tatively and with a user study. We show results for three dif-
ferent garment types (T-shirts, shorts and long-pants), see
Fig. 4. Notice that the texture is nicely mapped without
transferring background. Notice also how the 3D textures
are complete without holes and seams. We compare our
method against the popular Shape-context with Thin Plate
Spline (TPS) matching baseline, an image-to-image trans-
lation which operates on pixels directly, and image based
virtual try on methods.

4.1. Dataset

To train our models, we create datasets of garment im-
ages by scraping the websites of clothing stores – specifi-
cally the websites of Zalando [3], Tom Tailor [2] and Jack
and Jones [1]. The back view of clothing images is scarcely
available on the internet, so we were unable to create a
dataset of back view images large enough for training map-
ning networks. We leverage the fact that the distribution of
garment silhouettes is similar for both front and back views,
and either augment the dataset used for training the mapping
from the back view image to the back UV map by combi-
ning front view images with back view images or using only
front view images.

We create a datasets consisting of 2267 front images of T-
shirts. The networks pertaining to the back are trained using
a dataset of 2267 images of which 964 are back view images
and the rest are front images. The front shorts dataset has
2277 items. We use the same dataset to train the networks
pertaining to the back views. For pants we collect a dataset
of 3410 front views. We create a dataset for back views by
horizontally flipping the front view images and their corre-
spanding silhouettes. The back dataset has 3211 items.

The failure of the optimization based registration method
explains the discrepancy between the number of items in the
front and back views. Exploiting these front-back sym-
metries turns out to work well in practice, and allows to
compensate for the unavailability of back view images.

4.2. Shape Context Matching and Image-to-Image
Translation Baselines

We compare our method with shape context (SC) match-
ing plus TPS and image-to-image translation baselines. We
implement the shape context matching baseline as follows.
We first render the garment mesh to obtain a silhouette. To
make the baseline as strong as possible, we first pose the
garment in the same pose as the image. We then warp the
garment image to match the rendered garment. We then
lift the texture from the warped image onto the mesh using
projective texturing. Our experiments demonstrate that SC
matching and TPS [14] is not powerful enough to precisely
match the contours of the two shapes so the final texture al-
ways has artifacts along the sides. See Fig. 5.

We also compare Pix2Surf to a Pix2Pix [32] baseline, which
learns to produce a texture map from an image. To train
Pix2Pix, we use the same texture maps we use to train
Pix2Surf. Unlike Pix2Surf, Pix2Pix produces blurry results
and does not preserve details of the input texture. See Fig. 5.

For further evaluation, we have conducted a user study with
30 participants. We created 20 videos containing images of
reference clothing textures and two rotating textured 3D
avatars “wearing” the reference clothing. One is textured
using a baseline method and the other using Pix2Surf. We
ask participants to choose the better looking avatar. In 100%
of all comparisons, the avatar textured with Pix2Surf was preferred over the one textured using the baselines.

4.3. Towards Photo-Realistic Rendering

Provided with the texture map of a 3D avatar without clothes, Pix2Surf allows to map the texture atop the avatar from potentially an infinite number of clothing items scraped from online websites, automatically and in real time. See Fig. 6 Using these photo-realistic renderings, we are able to compare our method with Image based virtual try-on method such as VITON [28] and CP-VTON [69]. Since our method requires a 3D avatar as input, to be able to compare, we first render a photo-realistic image using our method, and use VITON to change the upper garment of the rendering. Fig. 7 shows that VITON and CP-VTON work reasonably well for some poses, but fail badly for extreme poses while our method does not. We note that these methods are related but conceptually different compared to our method, so a direct fair comparison is not possible. These methods are also not explicitly trained to handle such poses, but the figure illustrates a general limitation of image based virtual try-on approaches to generalize to novel viewpoints and poses. In stark contrast, we only need to generalize to silhouette shape variation, and once the texture is mapped to the 3D avatar, novel poses and viewpoints are trivially generated applying transformation in 3D.

5. Conclusions and Future Work

We have presented a simple yet effective model that learns to transfer textures from web images of clothing to 3D garments worn by virtual humans. Our experiments show that our non-linear optimization method is accurate enough to compute a training set of clothing images aligned with 3D mesh projections, from which we learn a direct
mapping with a neural model (Pix2Surf). While the optimization method takes up ten minutes to converge, Pix2Surf runs in real time, which is crucial for many applications such as virtual try-on. Our key idea is to learn the correspondence map from image pixels to a 2D UV parameterization of the surface, based on silhouette shape alone instead of texture, which makes the model invariant to the highly varying textures of clothing and consequently generalize better. We show Pix2Surf performs significantly better than classical approaches such as 2D TPS warping (while being orders of magnitude faster), and direct image-to-image translation approaches.

We believe our model represents an important step towards learning a generative model of textures directly in 3D. We plan to address this in the future since it is lacking in current models like SMPL \cite{smpl}. We focused on texture, and assume garment geometry is given, but we think it should be possible to infer geometry from images. Since clothing is laid out on a flat surface on web photographs, geometry inference will require modelling how 3D garments deform when they are flattened out on a surface.

Acknowledgments: We thank Bharat Lal Bhatnagar and Garvita Tiwari for in-depth discussions and assistance in our data preparation. This work is partly funded by the Deutsche Forschungsgemeinschaft (DFG) - 409792180 (Emmy Noether Programme, project: Real Virtual Humans).
References

[29] Stefan Hauswiesner, Matthias Straka, and Gerhard Reitmayr. Free viewpoint virtual try-on with commodity depth cameras. In *Proceedings of the 10th International Conference on


[71] Lingyu Wei, Qixing Huang, Duygu Ceylan, Etienne Vouga, and Hao Li. Dense human body correspondences using convolutional networks. In Computer Vision and Pattern Recognition (CVPR), 2016.


