MIRROR: Towards Generalizable On-Device Video Virtual Try-On for Mobile Shopping

DONG-SIG KANG, Seoul National University, Republic of Korea
EUNSU BAEK, Seoul National University, Republic of Korea
SUNGWOOK SON, Seoul National University, Republic of Korea
YOUNGKI LEE, Seoul National University, Republic of Korea
TAESIK GONG, Nokia Bell Labs, United Kingdom
HYUNG-SIN KIM, Seoul National University, Republic of Korea

We present *MIRROR*, an *on-device video virtual try-on* (VTO) system that provides realistic, private, and rapid experiences in mobile clothes shopping. Despite recent advancements in generative adversarial networks (GANs) for VTO, designing *MIRROR* involves two challenges: (1) data discrepancy due to restricted training data that miss various poses, body sizes, and backgrounds and (2) local computation overhead that uses up 24% of battery for converting only a single video. To alleviate the problems, we propose a generalizable VTO GAN that not only discerns intricate human body semantics but also captures domain-invariant features without requiring additional training data. In addition, we craft lightweight, reliable clothes/pose-tracking that generates refined pixel-wise warping flow without neural-net computation. As a holistic system, *MIRROR* integrates the new VTO GAN and tracking method with meticulous pre/post-processing, operating in two distinct phases (on/offline). Our results on Android smartphones and real-world user videos show that compared to a cutting-edge VTO GAN, *MIRROR* achieves 6.5× better accuracy with 20.1× faster video conversion and 16.9× less energy consumption.

1 INTRODUCTION

With booming e-commerce, clothing purchases are increasingly shifting online. Notably, *mobile phones* facilitate 76% of these transactions [11, 12]. However, online clothes shopping does not allow customers to evaluate if a piece of clothing *suits* them, which causes dissatisfaction and heightened returns. To step forward, there is a strong demand to recreate the offline clothing selection process *virtually* – enabling users to employ their smartphones to emulate trying on different outfits and poses before mirrors.

Multiple mobile services have offered a range of virtual try-on (VTO) experiences. TriMirror[1] generates a 3D avatar based on user inputs but falls short of realistic appearance. FXMirror[2] analyzes a customer's 3D image for more lifelike avatars but its scalability is limited by the need for an extra 3D depth camera. Furthermore, these avatar-based services require a significant burden to construct a 3D database, such as 3D clothes-modeling in a multi-view studio with manual point cloud manipulation [58]. In 2D image-based VTO, deep neural networks

[1] https://www.trimirror.com/
[2] http://www.fxmirror.net/en/main

*Hyung-Sin Kim is the corresponding author.
Authors' addresses: Dong-Sig Kang, Seoul National University, Seoul, Republic of Korea, silkyday@snu.ac.kr; Eunsu Baek, Seoul National University, Seoul, Republic of Korea, beshu9407@snu.ac.kr; Sungwook Son, Seoul National University, Seoul, Republic of Korea, sungwookson@snu.ac.kr; Youngki Lee, Seoul National University, Seoul, Republic of Korea, youngkilee@snu.ac.kr; Taesik Gong, Nokia Bell Labs, Cambridge, United Kingdom, taesik.gong@nokia-bell-labs.com; Hyung-Sin Kim, Seoul National University, Seoul, Republic of Korea, hyungkim@snu.ac.kr.

2474-9567/2023/12-ART163
https://doi.org/10.1145/3631420

Fig. 1. User demands in online clothes shopping, motivating generalizable, lightweight on-device video VTO.

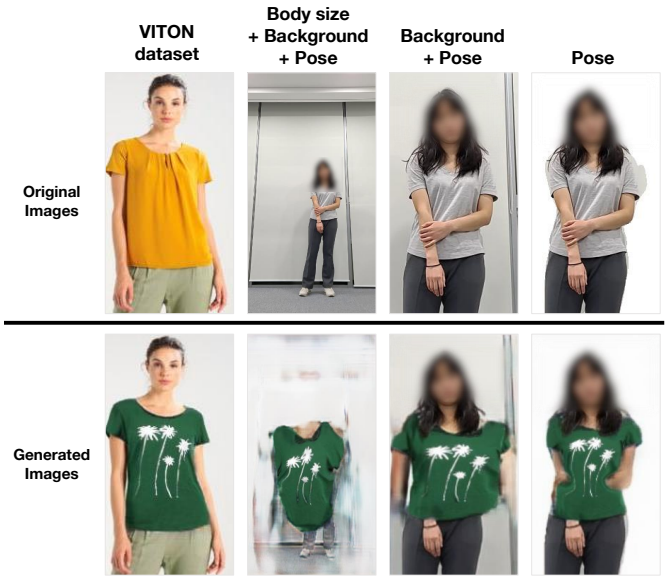


Fig. 2. Failures of a recent VTO GAN [20] in real-world videos due to various body sizes, backgrounds, and poses (i.e., domain shifts).

(DNNs), especially generative adversarial networks (GANs), put clothes directly on a person instead of an avatar [20, 24, 30, 47, 49]. Using these techniques, online shopping apps (e.g., Zeekit[3]) provide photo-based VTO. However, they merely support static images and require users to upload personal photos, risking privacy breaches.

This work systematically studies practical VTO for mobile clothes shopping, aiming to fill the gap between state-of-the-art GANs and real-world needs. Our study is grounded by a survey with >100 participants that reveals several user preferences in online clothes shopping (Section 2). Specifically, the survey shows that users want more vivid try-on experience using a video but without sharing their private information, such as body type and appearance. This leads to the design goals for our *MIRROR*: a mobile system that puts clothes adaptively on a *moving customer* in a recorded video, *100% on device* without information sharing (Figure 1).

Challenges. Despite the recent development of VTO GANs, two significant challenges emerge:

- **Data discrepancy:** VTO GANs [20, 24, 30, 47] are mostly trained on the VITON dataset [24] that contains 2D images capturing only *upper* bodies of *specific sizes* and *limited poses* against a *white background*. Since VTO is a delicate task that needs to understand detailed human semantics and warp/overlay clothes accordingly, naïve use of the existing GANs for real-world videos produces unfavorable outputs, due to *significant domain shift*. An erroneous example (Figure 2) is when a target image is taken in a natural indoor environment and has a user's full-body shot or their upper-body shot with the arms put on the body.

[3] https://zeekit.me/

Table 1. Latency and battery consumption of PF-AFN [20] on Samsung Galaxy S10 including Samsung Exynos M4 CPU and ARM Mali-G76 mobile GPU.

Task	Conversion latency	Battery consumption
960×540 frame	3.04 sec	-
900-frame video	57.3 min	805.8 mAh (23.7%)

- **Computation burden:** Even lightweight VTO GANs, such as PF-AFN [20] using knowledge distillation [27], remain unsuitable for a mobile device due to *many neural layers* and *lack of full mobile GPU support*. As shown in Table 1, PF-AFN requires ~3,041 ms to process a 960×540 image and consumes ~24% battery capacity to process a 900-frame video on Samsung Galaxy S10. Furthermore, typical optical flow-based feature tracking [1, 52], widely used for infrequent DNN excutions in on-device video analytics, cannot be applied to our case. This is because tracking VTO results needs *pixel-wise dense optical flow* that mandates another DNN execution [17, 31].

Approach. To tackle the problems, we design two novel components in *MIRROR*: (1) a generalizable VTO GAN, called DIVTON (Domain-Invariant VTO Network) and (2) a lightweight VTO tracking VITOFF (VIrtual-Try-OFF).

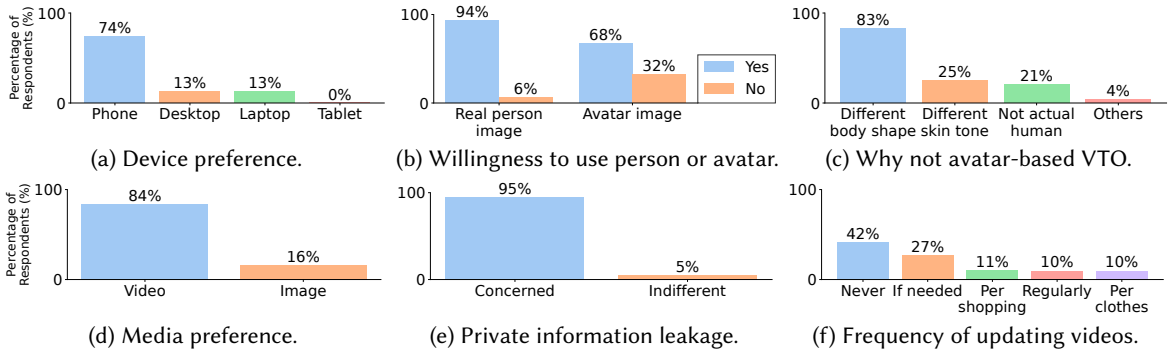
DIVTON aims to achieve a dual objective: capturing not only (1) intricate human body semantics for precise VTO but also (2) domain-invariant features for enhanced generalization. Prior efforts have focused on the former challenge. Initial VTO GANs leverage inputs from sophisticated human parsers to acquire detailed human body and clothing segments [24, 47, 49], making the generation module susceptible to parsing errors [30]. Conversely, the latest *appearance flow*-based networks (AFNs) embrace a parser-free architecture [2, 20, 26], yet they can still discern nuanced human semantics by distilling appearance flow from the human parsers during training. However, both categories of VTO GANs falter when faced with domain shifts. In contrast, the core concept of our DIVTON is harnessing the efficacy of *coarse-grained but relatively domain-invariant* human semantics for domain generalization. By designing a tailored model architecture and a distillation-based training method, we synergistically integrate the coarse-grained human parser with the AFN framework, providing generalizability without compromising the AFN's advantage to capture intricate human semantics devoid of paring errors.

Recognizing the computation burden of continuous DIVTON execution on mobile devices, VITOFF introduces *lightweight* pose/clothing tracking in synthesized frames to reduce the frequency of DIVTON execution. In contrast to the latest DNN-based optical flow techniques [29, 44, 45], VITOFF generates pixel-wise dense warping flow *without DNN computation*. The core assumption in VITOFF is that temporal pixel changes in the (unbreakable) body area exhibit high correlation. Since tracking individual pixels may be overkill in our context, VITOFF employs *selective* optical flow [39] that meticulously tracks only key features within the body area. Using the sparse feature flow as a high-quality *anchor*, we employ lightweight Thin-Plate Spline (TPS) transformation [4, 5] to generate dense warping flow for the current frame. Introducing two complementary metrics to identify diverse tracking errors, DIVTON is reactivated only when tracking outcomes prove unsatisfactory. Importantly, integrating VITOFF with DIVTON elevates video VTO quality over using DIVTON alone due to the comprehensive use of *spatiotemporal* information; errors exhibit temporal correlation, providing a consistent temporal view for users.

Lastly, given that customers want to see how the desirable clothes look on them before their final choices, a video should be synthesized with multiple candidate clothes. To utilize the application characteristic, we partition *MIRROR* operations into two phases: "Preview" and "Runtime." The preview phase analyzes a video and archives *reusable* information, fostering accelerated VTO for different clothes on the same video in the runtime phase.

Contributions. Our contributions are as follows:

- To our knowledge, *MIRROR* is the first on-device video VTO system for online clothes shopping. To ground our study, we conduct a survey that derives practical application scenarios and user requirements (Section 2).



(a) Device preference. (b) Willingness to use person or avatar. (c) Why not avatar-based VTO.

(d) Media preference. (e) Private information leakage. (f) Frequency of updating videos.

Fig. 3. Illustration of survey results including preferences for online shopping and VTO.

- We design two novel components: (1) DIVTON that captures detailed human body semantics as well as domain-invariant features from natural images and (2) VITOFF that minimizes neural-net computation while further improving video VTO quality via high-quality keypoint selection (Sections 4-6).
- Extensive experiments show *MIRROR*'s superiority on multiple off-the-shelf Android smartphones. Compared to PF-AFN, *MIRROR* achieves 6.5× better video VTO quality with 20.1× faster conversion and 16.9× less energy consumption (Section 8).

2 SYSTEM REQUIREMENTS AND APPLICATION SCENARIO

We first conduct a survey to ground our study, concretizing user requirements and an application scenario for *MIRROR*.

2.1 User Study and System Requirements

In order to investigate consumer requirements and objectives for virtual fitting services in the context of online clothing shopping, this study employs judgmental sampling. Participants were recruited by posting the survey link to the university's online bulletin board. This method resulted in an initial participation of 103 individuals. From this initial pool, a subset of 62 participants (ranging in age from 19 to 62, with a median age of 30; including 36 females) was chosen based on their alignment with the study's purposes. Specifically, they met the dual criteria of (1) engaging in online apparel shopping at least once a year and (2) expressing a discomfort stemming from the inability to physically try on clothes during their online apparel shopping experiences.

The survey questions primarily focused on three aspects: (1) analyzing customers' usage patterns and characteristics during online shopping, (2) determining the requirements to meet users' aspirations to achieve via VTO services, and (3) identifying appropriate system design considerations accordingly. Specifically, the survey consists of the following questions: (Q1) Which shopping device do you use for online shopping? (Q2) Given a demo of either (i) a real-image-based or (ii) an avatar-based (TriMirror[1]) VTO platform, would you be willing to use this service? (Q3) If you indicated a lack of willingness to use an avatar-based VTO, what were your reasons? (Q4) Which media do you prefer for VTO, a video or an image? (Q5) How do you feel about sharing your personal appearances with the VTO service provider? (Q6) How often would you like to update your appearances within a VTO service? Our rationale behind the survey questions is to understand user preference and find out the most appropriate design for the try-on service in online shopping that aligns with user requirements. The survey results of each question are summarized in Figure 3.

In addition, our survey unveils four reasons why the participants responded that the inability to try on clothes during online shopping is inconvenient. This pertains to their inability to confirm the suitability of clothing design,

color, size, and texture. In this work, we concentrate on design and color compatibility, given their potential for validation through the synthesis of clothes onto users' images using VTO. The remaining two challenges will be discussed in Section 9.

In light of our survey, we suggest five system requirements for VTO services:

R1. *Preference for Mobile Devices:* Figure 3a illustrates that a significant 74% of the participants prefer using phones over desktops, laptops, and tablets when shopping online. This finding aligns with the trends observed in m-commerce preferences [11, 12], indicative of the widespread accessibility of smartphones. Inspired by this result, we focus on designing a VTO system for smartphones in this study.

R2. *Real-Video-Based VTO:* Figure 3b indicates that participants favor a real-person-based VTO system over an avatar-based alternative. The primary reason cited for this preference is the disparity between avatars and actual human bodies (Figure 3c). Furthermore, Figure 3d reveals that an impressive 84% of the participants prefer a video-based VTO system rather than an image-based one. Consequently, our focus lies in developing a real-person-based video VTO system.

R3. *Real-World Generalizability:* Insights from our survey also extend to users' expectations concerning video backgrounds. The results reveal that users expect 52% neat backgrounds, 35% white backgrounds, and 13% non-neat environments for VTO videos. The diverse preferences underscore the inherent challenges in achieving uniformity across user-generated videos. Furthermore, human factors like hairstyle, body shape, and pose introduce an additional layer of diversity in the videos. Given the intrinsic variations in user-generated data, the system is required to be generalizable to accommodate the real-world diversity.

R4. *Privacy Preservation:* Figure 3e clearly reveals the privacy concerns, showing that a substantial 95% of the participants are reluctant to send their personal appearance data to VTO service providers [25]. Additionally, among the participants who expressed their willingness to try real-person-based VTO in Figure 3b, 76% stated that they would be willing to use it if privacy is ensured (22% with no additional conditions and 2% contingent on UI/UX assurances). As the best form of securing privacy, a VTO system is desired to run on a smartphone without sending any data externally. As a result, we can guarantee that any user data for our service will never be compromised or shared with other applications or third parties, including shopping malls.

R5. *Reusability:* The survey reveals compelling user inclinations: 41% would record a video only once for VTO and 32% would re-record if it is necessary (Figure 3f). This insight underscores the potential burden of re-recording videos for most users. Therefore, a VTO system needs to reuse the same video for multiple clothing interactions to enhance user convenience.

Taking into account the requirements, we propose *MIRROR*, a VTO system tailored for online shopping scenarios. Our primary design principle is the on-device processing paradigm, ensuring that all operations occur within the user's device, without sending any sensitive data outside of *MIRROR* (R1, R4). The application focuses on generating VTO results using users' various real-world videos (R2, R3). Notably, we consider the computational cost due to limited resources of mobile devices (R1), while also leveraging the user's preference for reusing videos for VTO (R5).

2.2 Application Scenario

Figure 4 illustrates the usage scenario for *MIRROR*, which aims to enhance the user experience and achieve the goal of virtual fitting. In this scenario, users perform three main tasks within the *MIRROR* application, which operates independently from the online shopping application they use.

2.2.1 Preparation Phase. Users initiate by downloading the *MIRROR* app to their smartphone. Using their smartphones (R1), they are likely to record only a single video to save their time and labor (R5). The video can include various poses and backgrounds (R2, R3). *MIRROR* executes its "Preview" mode completely on the user's

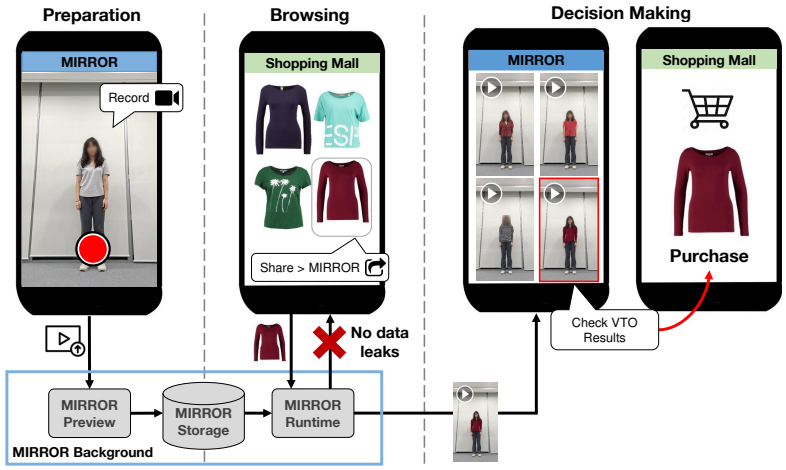


Fig. 4. Three-phase mobile clothes shopping scenario for *MIRROR*.

device, generating preliminary information from the video for later use. If the user wants to update the video, they can re-record the video and execute the "Preview" mode for the updated video. Importantly, for privacy preservation, the video never leaves the *MIRROR* app including external shopping applications (R4).

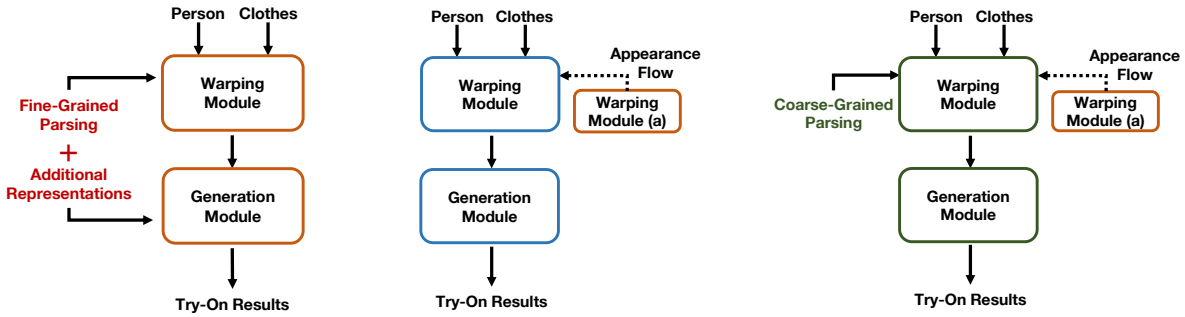
2.2.2 Browsing Phase. Users explore various clothing items in an online shopping application using their smartphones. By employing touch and hold press, users can share the images of desirable clothes with *MIRROR*. Then, these selected items will be added to the VTO list in *MIRROR* securely without any visibility to the user's screen, browser, or shopping applications (R4). Whenever a new item is added to the list, *MIRROR* enters its "Runtime" mode, converting the user's video into a VTO video featuring the selected item. The "Runtime" mode runs on the smartphone as a background process, enabling users to continue shopping using the shopping application. All processes are run in *MIRROR*, while the shopping application only provides clothes images without any access to user's video (R4). Given that the same video might be reused multiple times, the required information calculated by the preview mode is used in the conversion (R5).

2.2.3 Decision-Making Phase. Users access the *MIRROR* VTO list, where the selected items are virtually put on the user-provided video (R2, R3). Ideally, *MIRROR* should complete these video conversions before users conclude their browsing phase. These VTO videos are expected to provide users with at least a rough sense of how well the design and color of the items match their appearance, aiding in their purchasing decisions. Following the decision-making process in the *MIRROR* application, users can return to the online shopping application to finalize their purchase of the chosen clothing items. Notably, *MIRROR* does not share the VTO videos but only the selected items with the shopping application for safeguarding user privacy (R4).

3 RELATED WORK

3.1 Image- and Video-Based Virtual Try-On

VITON [24] is the pioneering work on image-based VTO, which opens a VTO dataset and proposes GAN-based image synthesis. To understand detailed human body semantics, early work on VTO GANs (depicted in Figure 5a) is usually trained using human-clothes image pairs sourced from the VITON dataset. Self-supervision is employed due to the absence of ground-truth labels for VTO outcomes [24, 47, 49]. In the training process, traditional VTO networks typically take the following steps: (1) utilize *external preprocessing networks* [8, 22, 35] to remove the clothing from an input image and generate multiple human representations, such as human parser, pose, and



(a) Traditional VTO using explicit fine-grained parsers

(b) Contemporary VTO distilling appearance flow to a parser-free model

(c) Appearance flow- and explicit coarse-grained parser-based VTO (Ours)

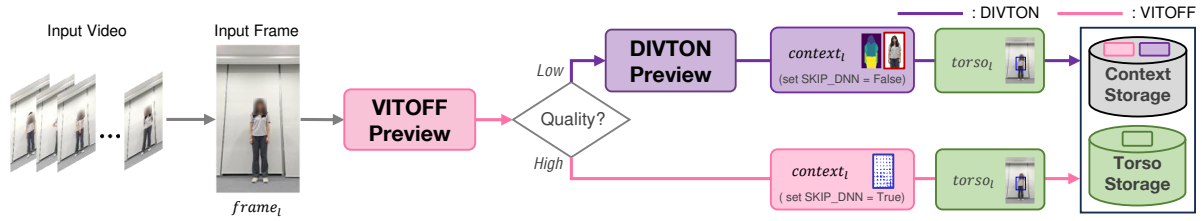
Fig. 5. Generations of VTO GANs. The dashed lines are activated only during training, whereas inference solely relies on the solid-line pipeline. In the case of DIVTON (Figure 5c), intricate human semantics are learned by distilling appearance flow from the warping module in Figure 5a during the training process. Furthermore, our model captures domain-invariant features through the integration of an external coarse-grained human parser.

densepose, and (2) try to restore the original image by reapplying the clothing, guided by these rich human representations. However, the computational demands of preprocessing networks are excessive for mobile devices. In addition, the explicit dependency on preprocessing network inputs for the generation module, as depicted in Figure 5a, makes the approach susceptible to parsing errors [30].

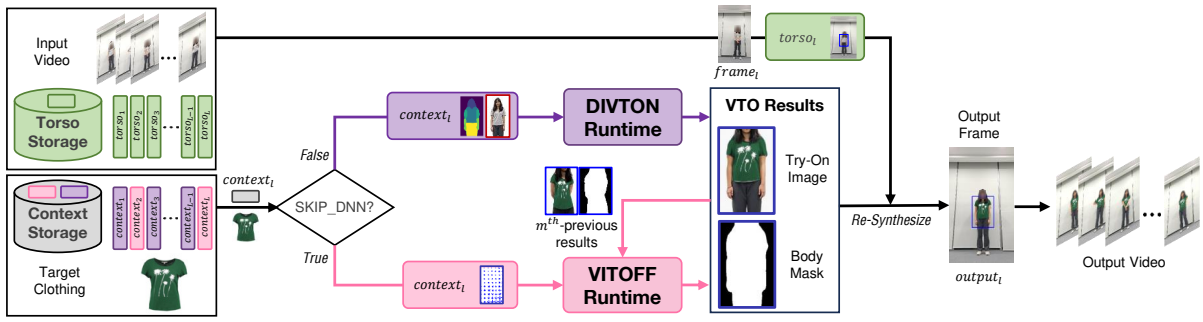
To alleviate the problems, PF-AFN [20] introduces a *parser-free* VTO GAN by eliminating the preprocessing steps, which captures intricate human body semantics directly from a person image, as depicted in Figure 5b. To this end, PF-AFN presents a distillation-based training method that includes a complex parser-based tutor DNN and a parser-free student DNN [27], where the student DNN learns *appearance flow* that is distilled from the tutor DNN's warping module. The tutor (Figure 5a) is trained using the aforementioned self-supervision to generate a synthetic image that virtually puts a target clothing on a person image, guided by its associated representations. Then the student DNN is trained using an original image together with its corresponding appearance flow and synthesized image driven by the tutor (i.e., distilled knowledge). Specifically, the student learns to restore the original image by overlaying the original clothing onto the synthesized image by using appearance flow, bypassing the need for explicit representations. Once training is complete, only the parser-free student DNN is utilized for inference. The latest methods embrace the parser-free paradigm due to its error-resilient nature [2, 26].

However, both categories of existing VTO methods focus solely on capturing intricate human semantics within the *confined source domain*, without considering the aspect of generalization to real-world scenarios. For instance, the VITON dataset has significantly restricted formats, such as upper body images of a fixed size and restricted poses on a (nearly) white background, as shown on the leftmost side in Figure 2. Our observations indicate that the VTO GANs trained on this dataset experience substantial performance degradation when handling real-world images (Figure 2). In contrast to the previous studies, we propose a tailored utilization of *coarse-grained* human semantics combined with appearance flow, as depicted in Figure 5c. This approach achieves *domain generalization* while simultaneously capturing intricate human semantics, improving performance with real-world images.

A few pieces of work have recently studied video VTO [17, 31, 34, 57]. The pioneering work [17] opens a video VTO (VVT) dataset and proposes FW-GAN that warps both a previously synthesized frame and a clothing image and fuses them to synthesize the current frame. FW-GAN executes multiple DNNs and *DNN-based dense optical flow* for every frame. The latest methods also utilize dense optical flow [34, 57] and/or even propose a transformer architecture [31]. However, they are extremely heavy to run on a mobile phone. Instead, this



(a) Preview phase: *MIRROR* operates DIVTON and VITOFF to extract reusable information from a user video and stores it.



(b) Runtime phase: *MIRROR* puts various clothes on the same video efficiently by using the context and torso storages.

Fig. 6. Overview of two-phase *MIRROR* operation that supports the mobile clothes shopping scenario.

work offers lightweight on-device video VTO capabilities through *infrequent execution* of a VTO GAN and the utilization of *non-DNN sparse optical flow* for warping synthesized frames.

3.2 On-Device Video Analytics

Given that running DNNs continuously to analyze every frame of a video on a mobile device is a huge burden, there have been a number of studies to enable on-device, lightweight video understanding. Some prior work utilizes both mobile devices and servers synergistically for fast and low-power video analytics [9, 50, 54], which sacrifices privacy by sharing user data with the server. Another group of studies provides 100% on-device video analytics by using lightweight video interpolation to execute a DNN infrequently. These studies offer application-specific solutions through meticulous exploration of target applications, including object detection [1, 48], human pose detection [52], and Android cursor detection [15]. While the prior work mostly focuses on detection-oriented applications, this work is the first to investigate VTO, a more sophisticated GAN-based application, for on-device video understanding and generation.

4 MIRROR OVERVIEW

This section provides an overview of *MIRROR*, a generalizable on-device video VTO system that aims to support the application scenario in Section 2. *MIRROR* has two novel components, DIVTON and VITOFF, to provide lightweight yet accurate video VTO. Specifically, DIVTON analyzes spatial information of an input frame for accurate VTO with domain generalization but requires heavy DNN computation. On the other hand, VITOFF provides lightweight two-step warping flow generation by analyzing spatiotemporal information of multiple input frames and previous VTO results. VITOFF is faster and more energy efficient than DIVTON but accumulates errors as executed for many frames continuously. For synergistic interplay of these two components, *MIRROR* utilizes VITOFF by default while selectively executing DIVTON to preserve accuracy.

To support the target scenario (i.e., mobile clothes shopping) efficiently, *MIRROR* divides DIVTON and VITOFF operations into two phases, as illustrated in Figure 6.

4.1 Preview Phase

The preview phase (Figure 6a) runs *only once* for each input video in the preparation phase, which utilizes the DIVTON and VITOFF previews to calculate reusable information and store the information in the context and torso storages for later use. This aligns with the need for video reusability, as explained in Section 2.2. When the *MIRROR*-preview receives a new l-th frame $frame_l$ (l > 1), it measures the quality of the VITOFF-preview results and, if necessary, optionally runs the DIVTON-preview for low-quality results. We employ a flag `SKIP_DNN` for each frame $frame_l$, which is set to `False` when the DIVTON-preview is executed, or `True` otherwise.

Upon DIVTON-preview execution, it generates a contextual information $context_l$ comprising a VITON-style person image along with its associated coarse-grained parsing result. These components serve as domain generalization for the appearance flow-based VTO GAN in the DIVTON-runtime. The underlying rationale is that extracting a VITON-style image from the target domain reduces format disparities compared to the source domain, while the coarse-grained parsing result effectively functions as a domain-invariant feature applicable to various target domains. Furthermore, the DIVTON-preview generates a torso box $torso_l$ to facilitate re-synthesis of the VTO outcome with the original image, which enhances the robustness of VTO across diverse domains.

In the case of the VITOFF-preview, it tracks intricate human body semantics for the current frame, starting from the m^{th}-previous frame. Here, the m^{th}-previous frame is defined as the most recent frame processed by the DIVTON-preview. Specifically, the VITOFF-preview tracks the current torso box $torso_l$ and generates a contextual information $context_l$ within this box that contains dense pixel-wise warping flow between the m^{th}-previous and current frames. Lastly, both types contextual features ($context_l$) originating from the DIVTON and VITOFF previews are stored in the context storage while the torso box ($torso_l$) is stored in the torso storage.

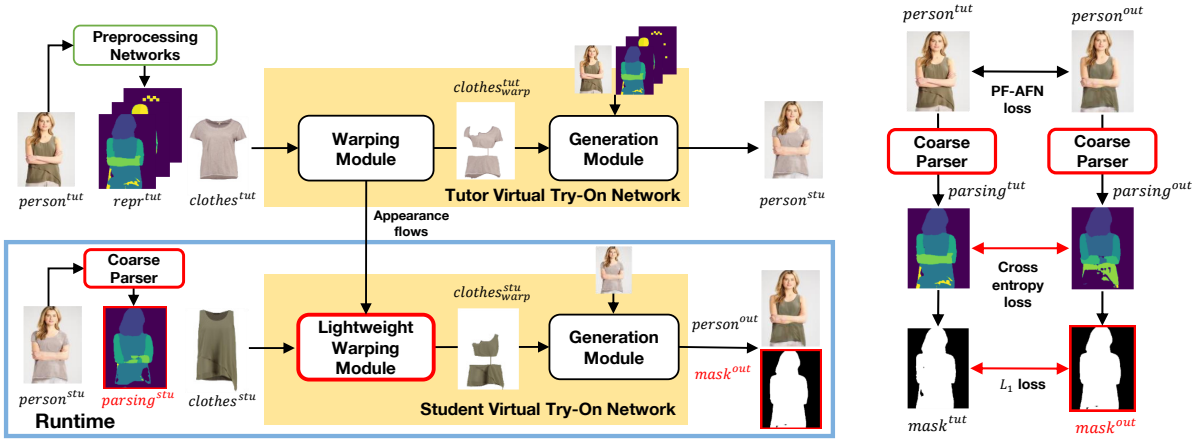
4.2 Runtime Phase

The runtime phase (Figure 6b) runs for each selected clothing item in the browsing phase, leveraging the DIVTON and VITOFF runtimes to virtually put the clothing on the user's video accurately and efficiently. Specifically, for an input frame $frame_l$, the *MIRROR*-runtime loads both the context information $context_l$ and the torso box $torso_l$ from the corresponding storages. Importantly, the pre-computation and caching of the essential VTO-related data significantly accelerates the *MIRROR*-runtime process. After examining the `SKIP_DNN` flag, it executes the DIVTON-runtime if the flag is `False`, or the VITOFF-runtime otherwise.

Both the DIVTON and VITOFF runtimes generate dual outputs for the input frame $frame_l$, a synthesized VTO image and a body mask. Notably, we add the body mask output for domain generalization due to its higher domain-invariance compared to intricate human semantics. To this end, while the DIVTON-runtime computes appearance flow-based VTO GAN on a person image and its associated parsing result in the loaded $context_l$, the VITOFF-runtime warps the m^{th}-previous VTO results for the current frame using the torso-bounded dense warping flow in the loaded $context_l$. The term m^{th}-previous VTO results signifies outcomes corresponding to the most recent frame processed by the DIVTON-runtime. Lastly, the *MIRROR*-runtime utilizes the loaded torso box $torso_l$ to overlay the synthesized image only exclusively onto the torso region of the original image. This amalgamation produces the ultimate synthesis result $output_l$.

5 DIVTON: ENABLING DOMAIN GENERALIZATION FOR VIRTUAL TRY-ON

This section presents details of DIVTON that is designed to resolve data discrepancy between the source and target domains by capturing both intricate human semantics and domain-agnostic features. To this end, we carefully integrate a coarse-grained human parser with the appearance flow-based network (AFN) [20]. Figure 7a



(a) Network (b) Loss

Fig. 7. DIVTON architecture. While leveraging the distillation-based training pipeline of the baseline, DIVTON has unique components that are marked in red: (1) lightweight warping module, (2) coarse parser, (3) additional parsing input $parsing^{stu}$ only for the warping module, (4) parsing constraint loss between $parsing^{tut}$ and $parsing^{out}$, and (5) additional mask output $mask^{out}$ and mask loss between $mask^{tut}$ and $mask^{out}$ for background re-synthesis. Note that all input and output images are in the form of the VITON dataset and only the blue region is executed at runtime.

depicts the DIVTON architecture, where only the parser-free student DNN in the blue box is executed during runtime while the parser-based tutor DNN is used to distill appearance flow during training.

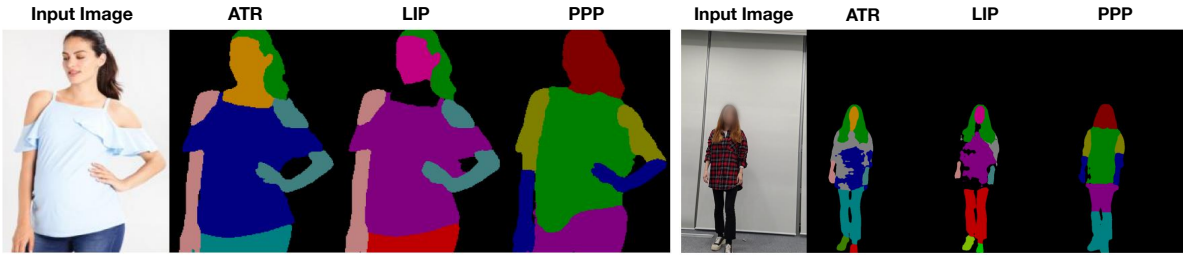
In contrast to the baseline AFN [20], DIVTON additionally utilizes a coarse-grained parsing result (i.e., domain-invariant feature) as an auxiliary input ($parsing^{stu}$) to the student DNN to foster domain generalization. The deployment of the coarse-grained parsing input is carefully engineered to mitigate the influence of parsing errors on VTO outputs: exclusively allocated to the warping module, exempting the generation module from its impact. In addition, DIVTON generates a domain-invariant body mask output ($mask^{out}$) to ensure smooth background re-synthesis. Lastly, as depicted in Figure 7b, DIVTON additionally employs parsing loss and mask loss during the distillation-based training to learn these domain-invariant features alongside with appearance flow. The newly incorporated features in DIVTON are marked in red in Figure 7.

Furthermore, while the core part of DIVTON in Figure 7a operates in the runtime phase, the whole two-phase pipeline of DIVTON includes careful preprocessing to polish a raw input frame in the preview phase and postprocessing of its output frame for background re-synthesis in the runtime phase, which further mitigates data discrepancy between the source and the target domains. We first describe the DIVTON core in Sections 5.1 to 5.2 and then the overall two-phase operation in Section 5.3.

5.1 Coarse-Grained Human Semantics: Domain-Invariant Feature

5.1.1 Problems. From the error cases on the VITON test data, we observe that PF-AFN is erroneous when the arms are overlapped on the body; the arms are falsely erased and covered with a target clothing, as the rightmost case in Figure 2.

5.1.2 Parsing Input. To get a hint for solving the problem, we also analyze other parser-based VTO GANs (e.g., ACGPN [49]) that utilize detailed human semantics as explicit inputs. Our finding is that the parser-based GANs better distinguish the overlapped arms from the body. However, naïvely adding the preprocessing networks to the AFN is not desirable since it has already been revealed that these heavy preprocessing networks are problematic in terms of computational burden and error propagation [30].



(a) VITON dataset (source) (b) Real-world dataset (target)

Fig. 8. Example human segmentation results of three deep human parsers that are trained using coarse-grained (PPP) and fine-grained parsing labels (ATR and LIP), respectively.

To deal with the problem, we deeply investigate what kind of human body semantics to use and how to use the information for lightweight domain generalization without error propagation. First, among the three types of human representations given by the preprocessing networks, parsing, pose, and densepose, we decide to use only the parsing information (i.e., segmentation of each body part, $parsing^{stu}$ in Figure 7a). This is because generating densepose requires heavy computation while pose information might be too simple. Another reason is that the parsing information is also useful for other purposes in our scenario, such as image preprocessing (Section 5.3) and keypoint extraction for optical flow (Section 6.1).

Second, instead of active temple regression (ATR) [37] and look into person (LIP) [36] datasets commonly used for fine-grained human parsers, we use Pascal-Person-Part (PPP) dataset [10]. We found ATR and LIP datasets' rich labels for clothes are vulnerable to domain shifts (e.g., a clothing with unseen complex patterns or clothes on the background). In contrast, the PPP dataset provides simplified labels only for person's body, arms, face, and legs. We train the parsing network in [35] on the dataset to focus on *human* parsing, which is coarse-grained but robust to domain shifts. For example, Figure 8 shows human semantic segmentation results of deep human parsers (ResNet-101) that are trained on the three datasets, ATR, LIP and PPP, respectively. While all the three parsers provide fair segmentation results in the VITON (source) dataset (Figure 8a), the parsers trained on ATR and LIP suffer significant segmentation errors in a real-world (target) dataset since they are weak for domain shifts (Figure 8b).

Last but not least, in contrast to existing GANs [24, 47, 49] that input human representations to both the warping and generation modules, we determine to input the coarse-grained parsing results *only to the warping module*. The intuition is that while coarse-grained human semantics is important to distinguish the arms from the body when warping a target clothing in the target domain, once the warped clothing ($clothes^{stu}_{warp}$ in Figure 7a) is given, using the coarse-grained semantics further in the generation module can bother fine-grained synthesis.

5.1.3 Parsing Constraint Loss. We also utilize the coarse-grained parser to calculate parsing constraint loss [17] in training DIVTON. This is because each body part should be clearly distinguished in an output image, same as the corresponding input image except the clothing-related part; DIVTON should preserve coarse-grained human semantics. To this end, as shown in Figure 7b, we parse both the original image $person^{tut}$ and the output image $person^{out}$, generating two coarse-grained parsing results $parsing^{tut}$ and $parsing^{out}$. Given that $person^{tut}$ and $person^{out}$ wear the same clothing ($clothes^{tut}$), we make $parsing^{tut}$ supervise $parsing^{out}$ by calculating pixel-level cross entropy loss [55].

Thus, cross entropy loss enforces a constraint that $person^{tut}$ and $person^{out}$ have an identical parsing result [10] for each individual pixel. For example, if the synthesized image $person^{out}$ exhibits a larger body size than the original image $person^{tut}$, the parsing result of $person^{out}$ labels certain pixels as the body part, whereas the parsing result of $person^{tut}$ designates those same pixels as the background part, which increases the parsing constraint

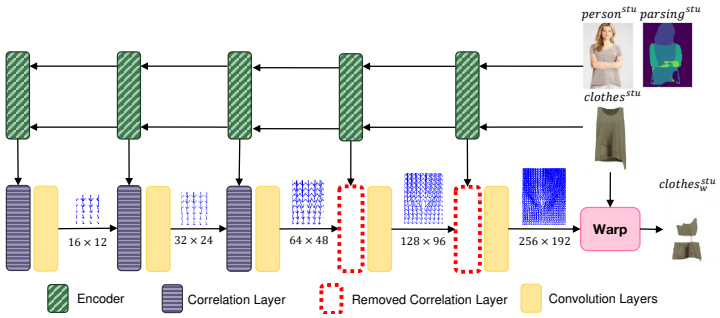


Fig. 9. The lightweight warping module in DIVTON. The last two correlation layers for processing high-resolution features are excluded.

loss. In addition, since human posture directly influences the parsing results, the parsing constraint loss plays a crucial role in distinguishing between arms and the torso where these body parts overlap. Importantly, our generation module does not take a parsing input but is trained by the parsing loss. The loose coupling enables the generation module to be guided by but not completely trust parsing results.

5.1.4 Body Mask Output and Mask Loss. For our target application, DIVTON's output ($person^{out}$) should be re-synthesized with the raw input frame (Figure 10), where human parsing takes the key role again. Given that the parsing result for the output image can be different from that for the input image ($person^{stu}$) due to different clothes, relying on the input parsing result is dangerous. However, executing the coarse-grained parser again for the output image at runtime significantly increases latency. To solve the problem, we design the generation module to produce an additional mask output ($mask^{out}$ in Figure 7) that distinguishes the human and the background in $person^{out}$. To this end, we generate a mask label $mask^{tut}$ from the parsing label $parsing^{tut}$ (i.e., body vs. non-body) and calculate supervised L_1 loss between $mask^{tut}$ and $mask^{out}$.

5.2 Lightweight Warping Module

Next, given that DIVTON aims to run on a mobile device, we analyze latency of PF-AFN and find room for relieving its computation burden without sacrificing try-on quality.

5.2.1 Problems. We observe that PF-AFN requires 3.04 seconds to process a single 960×540 image on Samsung Galaxy S10 (Table 1), too long for *video* VTO. Further analysis reveals that the warping module consumes most of the time (2.41 seconds) since Android does not enable it to run on GPU currently. Although only the specific `grid_sample` layer is not implemented for mobile acceleration, since it is executed twenty times in the warping module, running the `grid_sample` layer on CPU and other operations on GPU results in frequent CPU-GPU communication; running the warping module on both GPU and CPU is even slower than using only CPU.

5.2.2 Correlation Layer Removal. The warping module has a feature pyramid structure in Figure 9, which originally includes five *correlation layers* [18] to generate dense warping flow by reflecting the all pixel-to-pixel relationship between the person and clothes images. Since a correlation layer (purple color) requires 49× more computation than the next convolution layer (yellow color), our goal is to find and remove redundancy in them. The intuition is that since a person's body movement is restricted in a certain range, neighboring pixels in a clothing would have similar flow. Based on this intuition, we exclude the last two correlation layers, meaning that the human-clothes relationship at the level of all-to-all pixel correlation is considered only for low-resolution features while local correlation is still used for high-resolution features. This simple approach accelerates the warping module 1.84× on CPU, achieving 1309 ms latency on Samsung Galaxy S10.

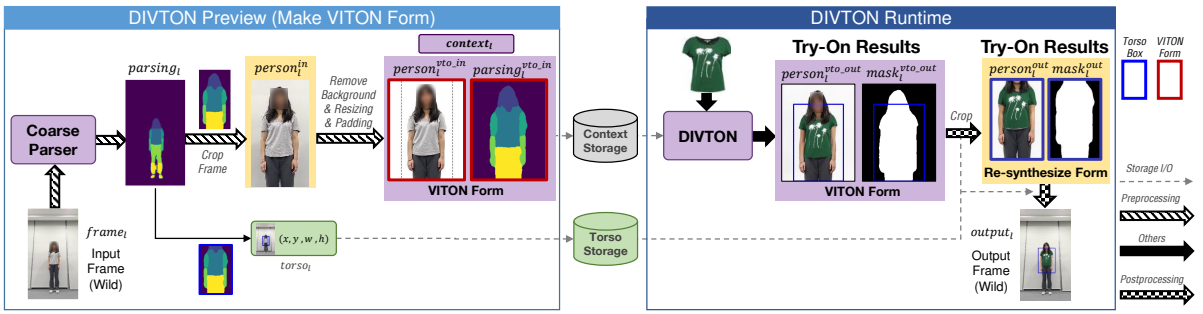


Fig. 10. Two-phase operation of DIVTON: (1) The preview phase utilizes the parsing result ($parsing_l$) for an original input frame $frame_l$, generating a torso box $torso_l$ and an input data $context_l$, including a VITON-style image $person_l^{vto\_in}$ and its parsing result $parsing_l^{vto\_in}$ ($person^{stu}$ and $parsing^{stu}$ in Figure 7, respectively). (2) The runtime phase postprocesses the VITON-style output $person_l^{vto\_out}$ and its human mask $mask_l^{vto\_out}$ to generate a re-synthesized final output frame $output_l$.

5.3 Deep Pre- and Post-processing for Domain Generalization

This section presents how DIVTON, trained on the confined VITON dataset, deals with various formats in target data via pre- and post-processing in the preview and runtime phases, respectively.

5.3.1 Problems. An image in the representative VITON dataset consists of the upper body and face with similar relative sizes on the white background, as the leftmost case in Figure 2. In contrast, naturally taken customer images are significantly different in that they have various poses, image sizes, and backgrounds (i.e., domain shift). Therefore inferencing these wild images directly results in disastrous outputs as the second left case in Figure 2. For generalizable VTO, pre-/post-processing is necessary to make a wild image as similar as possible to VITON data and synthesize an DIVTON output with the wild image again.

5.3.2 Preprocessing in the Preview Phase. As shown in Figure 10, DIVTON-preview preprocesses a raw input frame $frame_l$, generating VITON-style inputs ($person_l^{vto\_in}$ and $parsing_l^{vto\_in}$) for the DIVTON core and a torso box $torso_l$ for re-synthesizing VITON-style outputs with the background. To this end, we actively utilize the corase-grained human parsing result $parsing_l$.

To generate VITON-style inputs using the parsing result, we use the minimum and maximum locations of the upper body parts including face, arms, and torso, and add padding to reliably incorporate the whole upper body, resulting in a Region of Interest (RoI) represented as $person_l^{in}$. Then the RoI is fine-tuned to have a similar form of VITON data. First, since VITON data has the white background, we recognize $person_l^{in}$'s background using the parsing result and remove it. Next, given that VITON data has a fixed size (256,192), both the backgroundless person image and its parsing result are resized using cubic [32] and nearest interpolation, respectively. Importantly, the ratio of width and height remains the same to maintain the human shape [42] and any gap between the resized RoI and the (256,192) rectangular is filled in white to be recognized as the background. The final results, $person_l^{vto\_in}$ and $parsing_l^{vto\_in}$, are stored in the context storage and used by the DIVTON core at runtime.

In addition, we observe that the VTO network can distort the face image even though a target clothing is not put on the face. To preserve the face in the original image, we draw another (blue) bounding box excluding the facial part, represented as $torso_l$, and store it in the torso storage.

5.3.3 Postprocessing in the Runtime Phase. DIVTON-runtime re-synthesizes DIVTON's output image $person_l^{vto\_out}$ with the original image $frame_l$ by using the mask output $mask_l^{vto\_out}$ and the torso box $torso_l$. Importantly, all the pixels in the original image that are out of the torso box are maintained, including the *facial* part. To

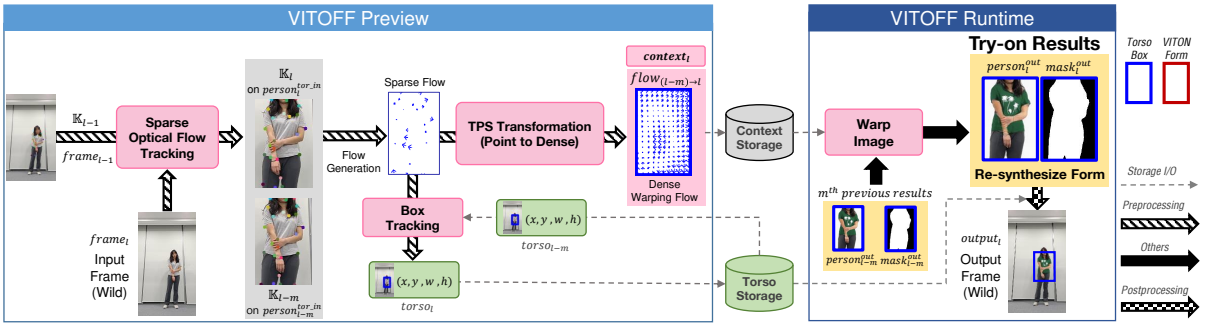


Fig. 11. Two-phase operation of VITOFF: In the preview phase, (1) sparse keypoints $\mathbb{K}_l$ are tracked using optical flow, (2) TPS transformation generates a dense warping flow $flow_{(l-m)\to l}$ based on the sparse flow, and (3) the torso box $torso_l$ is tracked from $torso_{l-m}$ by using the sparse flow. In the runtime phase, $person_l^{out}$ and $mask_l^{out}$ are generated by warping the previous results $person_{l-m}^{out}$ and $mask_{l-m}^{out}$ with $flow_{(l-m)\to l}$ and re-synthesized for the final output $output_l$.

re-synthesize only the torso area, we first crop and resize the two outputs, producing torso-bounded outputs $person_l^{out}$ and $mask_l^{out}$ that match the original torso size. The white background in $person_l^o$ is recognized by the mask output $mask_l^{out}$ and replaced by the original scene. With the recovered background, $person_l^o$ overwrites the torso area in the original image, resulting in the final output frame $output_l$.

6 VITOFF: TRYING OFF COMPUTATION BURDEN FOR VIDEO VTO

Although DIVTON provides domain-invariant VTO, the use of a deep human parser (i.e., additional DNN) increases computation burden compared to the parser-free baseline PF-AFN. To reduce latency and battery consumption on mobile devices, *virtual try-off* (VITOFF) avoids the heavy use of DIVTON by selectively recycling its previous output frames for later frames. As shown in Figure 11, VITOFF tracks and uses the differences between the current frame $frame_l$ and the two previous frames, (1) the latest DIVTON-processed frame $frame_{l-m}$ and (2) the latest-tracked frame $frame_{l-1}$, to convert the previous VTO outputs ($torso_{l-m}$, $person_{l-m}^{out}$ and $mask_{l-m}^{out}$) into the current ones in a lightweight manner. VITOFF's tracking results for the current frame, torso box $torso_l$, VTO result $person_l^{out}$ and body mask $mask_l^{out}$, are post-processed to produce a final VTO output frame $output_l$, as described in Section 5.3. VITOFF quality is measured by two synergistic metrics and DIVTON is run if the quality is unsatisfactory. Lastly, VITOFF operation is also divided into two phases, which further reduces the computation burden at runtime.

6.1 Selective Keypoint-Based Warping Flow

VITOFF provides two-step warping flow generation in the preview phase: (1) optical flow with coarse-grained parsing-guided *selective* keypoints and (2) keypoint-based pixel-wise dense flow using Thin-Plate Spline (TPS) transformation [5].

(1) Sparse Optical Flow with Keypoint Selection. We exploit optical flow to predict the movement of pixels between two frames in a video. Although dense optical flow tracks every pixel and shows good performance [6, 7, 29, 44, 45], it is difficult to apply to mobile devices due to heavy computation. In addition, it may not be necessary to track every pixel separately since body pixels would move together as a group. In this case, TPS transformation [5] can approximate the movement of the entire frame based on specific anchor pixels, which greatly reduces the computational overhead. We explain the details of TPS later in this section. Therefore, we select up to only 30 important pixels that are necessary for VTO, i.e., *keypoints*, and generate *sparse optical flow* [39] by tracking only these keypoints.

Careful keypoint selection is important for lightweight but accurate VTO frame generation and to this end, we exploit coarse-grained human semantics again. To extract the <30 keypoints ($\mathbb{K}_{l-m}$) efficiently from a frame $frame_{l-m}$ processed by DIVTON-preview, we adopt Oriented FAST and Rotated BRIEF (ORB) feature extractor [43] whose latency is less than 10 ms, with some modifications as below. First, among the keypoints given by the ORB extractor, we ignore those outside of the upper body (i.e., body segment in $torso_{l-m}$). The parsing result used for DIVTON is used again here to recognize the upper body. In addition, a straightforward option for keypoint selection, i.e., the highest score first selection, often results in the keypoints that are densely located in a small area where important pixels are clustered. However, closely clustered keypoints can result in the omission of keypoints with slightly lower scores, which are still vital for representing the entire pixels as anchor points. To address the problem, we apply the non-max suppression (NMS) that is widely used for reducing the number of box proposals in object detection [21]: repetitively selecting the highest score keypoint while removing the neighboring keypoints within 20 pixels of every selected keypoint. Lastly, the current keypoints $\mathbb{K}_l$ are tracked from the latest-tracked keypoints $\mathbb{K}_{l-1}$.

(2) Dense Warping Flow. Given the two selective keypoint sets $\mathbb{K}_{l-m}$ and $\mathbb{K}_l$, we generate a dense warping flow for *entire pixels* from the m^{th}-previous torso-bounded frame $person_{l-m}^{tor\_in}$ to the current frame $person_l^{tor\_in}$, denoted as $flow_{(l-m)\to l}$. We use TPS transformation, a smooth interpolation method used for shape matching [4], to find the source pixels of the current frame $person_l^{tor\_in}$'s pixels in the m^{th}-previous frame $person_{l-m}^{tor\_in}$. The intuition is that the carefully selected keypoints can act as high quality anchors, good enough to generate dense warping flow via lightweight TPS transform.

Specifically, given a set of keypoints $\mathbb{K}$ and a pixel p in a frame, TPS transforms the input pixel location $p = (p_x, p_y)$ to another pixel $q = (q_x, q_y)$ as:

$$TPS(\mathbb{K}, p) = a_1 + a_2 \cdot p_x + a_3 \cdot p_y + \sum_{k \in \mathbb{K}} w_k \cdot u(k, p) \quad \text{where} \quad u(k, p) = ||k - p||_2^2 \cdot log(||k - p||_2). \tag{1}$$

We optimize the TPS parameters $a = \{a_1, a_2, a_3\}$ and $w = \{w_k | k \in \mathbb{K}\}$ to retrieve the source pixels of the current l-th frame's pixels in the m^{th}-previous frame, as below:

$$a_l^*, w_l^* = argmin_{a,w} \frac{1}{|\mathbb{K}_l|} \sum_{k \in \mathbb{K}_l} ||TPS(\mathbb{K}_l, k) - k'||_2, \tag{2}$$

where k' is the ground-truth source keypoint ($k' \in \mathbb{K}_{l-m}$) in the m^{th}-previous frame for each current keypoint k ($\in \mathbb{K}_l$); the TPS parameters are obtained based on the sparse keypoint flow. The obtained TPS parameters are then used to calculate the source pixel location in $person_{l-m}^{tor\_in}$ for every pixel in $person_l^{tor\_in}$. Finally, the dense warping flow $flow_{(l-m)\to l}$ is acquired by computing pixel-wise location differences between the source pixels and the current pixels. The warping flow is stored in the context storage as $context_l$.

6.2 Warping & Bounding Box Tracking

VITOFF-runtime generates the current VTO outputs ($person_l^{out}$ and $mask_l^{out}$) by warping the m^{th}-previous outputs ($person_{l-m}^{out}$ and $mask_{l-m}^{out}$) using the dense warping flow $flow_{(l-m)\to l}$, respectively. We obtain a pixel value in the current VTO image ($person_l^{out}$ or $mask_l^{out}$) by gathering its source pixels in the previous VTO output ($person_{l-m}^{out}$ or $mask_{l-m}^{out}$), with bilinear interpolation or nearest interpolation, respectively. Importantly, the runtime warping with an existing flow is much simpler than dense flow generation in the preview phase.

After warping, VITOFF-runtime re-synthesizes $person_l^{out}$ with the original image $frame_l$ using $mask_l^{out}$, as in Section 5.3. To this end, VITOFF-preview tracks the torso box and stores it in the torso storage (Figure 11). Given that box tracking does not need a pixel-wise dense flow, we calculate the average displacement of the keypoint locations between $\mathbb{K}_{l-m}$ and $\mathbb{K}_l$ and move $torso_{l-m}$ to $torso_l$ accordingly.

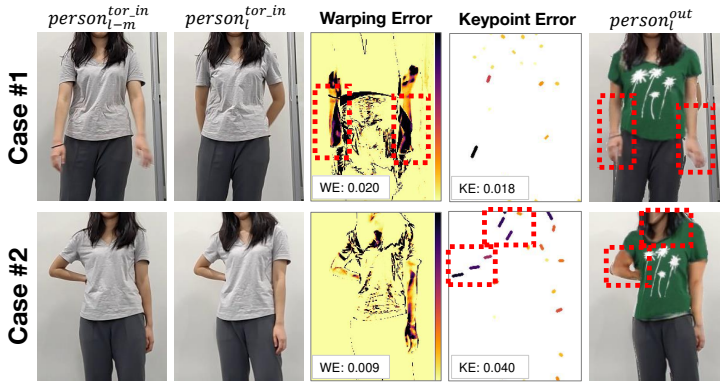


Fig. 12. Failure cases that could be detected by our quality metrics: Warping Error (WE) and Keypoint Error (KE). With different strengths, WE and KE capture different failure cases, Cases #1 and #2, respectively.

6.3 Synergistic Quality Metrics

As the gap between $frame_{l-m}$ and $frame_l$ increases, the quality of VITOFF-preview's warping flow decreases accordingly. Therefore it is necessary to evaluate the current warping flow and execute DIVTON-preview if its quality is not satisfactory, as shown in Figure 6a. Without any ground-truth frame to evaluate VITOFF results, we propose two synergistic evaluation metrics that capture different types of tracking errors.

6.3.1 Warping Error (WE). The first quality metric, named warping error (WE), evaluates whether the dense warping flow $flow_{(l-m)\to l}$ successfully transforms $person_{l-m}^{tor\_in}$ to $person_{l}^{tor\_in}$; WE is defined as the difference between the warped frame and the ground-truth frame $person_{l}^{tor\_in}$. Our intuition is that if the warping in the preview phase successfully tracks pose differences with the original clothing put on, runtime warping with a target clothing would also be successful.

For WE, we use normalized cross-correlation (NCC) [56] between two images, taking 1−NCC for WE. Importantly, naïve use of WE is vulnerable to domain shift, such as colors of the clothes and the background: when the clothes and the background have similar colors (i.e., high correlation), the WE can be low even though the actual transformation is wrong. To prevent this problem, we calculate a *domain-adaptive* WE threshold δ using the first frame of a video, which changes dynamically for each video by considering clothes and background. Specifically, we move the first frame by α pixels along the x- and y-axis, respectively, and compute NCC between the moved and the original frames. The averaged value of NCCs becomes $1-\delta$. When the background and clothing have a similar color, the artificial NCCs would increase, resulting in a smaller (tighter) WE threshold δ.

6.3.2 Keypoint Error (KE). Low WE does not necessarily mean that users are satisfied with the VITOFF quality. Despite small WE (average error), if the small errors are intensely located at a visually important region, users can recognize the VTO output as a failure. To capture this type of errors, we focus on the fact that keypoint locations are likely to get more attention from users. Specifically, we separately measure if the locations of source keypoints ($\mathbb{K}_{l-m}$) are estimated correctly from the current keypoints ($\mathbb{K}_l$), defining the estimation error as keypoint error (KE). The error distance for KE is the average L_2 distance between $\mathbb{K}_{l-m}$ and each of the source keypoints derived by TPS transformation with optimized parameters a* and w* from $\mathbb{K}_l$ as:

$$KE = \frac{1}{|\mathbb{K}_l|} \sum_{k \in \mathbb{K}_l} ||TPS(\mathbb{K}_l, k; a_l^*, w_l^*) - k'||_2 \tag{3}$$

6.3.3 Synergy between WE and KE. While severe errors can be captured by both WE and KE, it is important to note that WE and KE are complementary by capturing different types of failure cases, as shown in Figure 12 where errors are highlighted in red dotted boxes. Case #1 in Figure 12 illustrates a typical case that is captured by WE but not KE. In this case, the dense warping flow fails to track that the arms in $person_{l-m}^{tor\_in}$ become hidden in $person_{l}^{tor\_in}$, falsely displaying the arms in $person_{l}^{out}$. The keypoints located between the arms and the body in $person_{l-m}^{tor\_in}$ are still there in $person_{l}^{tor\_in}$ even though the arms disappear, resulting in low KE. However, the false arm display causes significant pixel differences, enabling WE to capture the failure. On the other hand, Case #2 in Figure 12 shows a failure case where KE is high but WE is low. In this case, the ground-truth and estimated frames are similar on average (i.e., low WE) but some important regions in $person_{l}^{out}$ look awkward (bent elbow and face). Given that the small important regions contain many keypoints, KE captures the failure.

We determine that a generated frame's quality is high only if WE $< \delta$ and KE < 0.10, and low otherwise. We investigate the impact of the hyperparameters in Section 8.4.

7 IMPLEMENTATION

We implement *MIRROR* on multiple Android smartphones with Kotlin, considering an image resolution of (960,540) in a vertical environment.[4] DIVTON is trained on the VITON dataset [24] having (256×192) images, using 4 TITAN RTX GPUs and the PyTorch framework [41]. The parsing constraint loss is multiplied by 0.5 and added to the final loss. In the same environment, our deep human parser (ResNet-101) is trained on the Pascal-Person-Parts dataset [10] that provides coarse-grained body parts segmentation. For inference, we use ONNX Runtime [16] for model execution and NNAPI, a unified interface to CPU, GPU, and neural-net accelerators, for acceleration. While the warping module is executed with CPU due to the `grid_sample` operation, the rest of DIVTON are executed with GPUs. Image processing operations are implemented via C++ and OpenCV with NDK. We extract 300 keypoint candidates with ORB feature [43] and select up to 30 of them using the upper body segmentation and NMS.

8 EVALUATION

We extensively evaluate the VTO quality and computation burden of *MIRROR* on Android smartphones.

Test Datasets. Without an existing real-world VTO dataset, we collected test videos from 31 people, with 540×960 resolution and 30-second length at 30 fps, resulting in a total of 27,900 frames. We recruited among the target users derived from the survey in Section 2, those who agreed to use their videos for research purposes. Figure 13 shows thumbnails of our real-world videos, which have diverse genders, backgrounds, poses, types of clothes, hair lengths, heights, zoom rates, angles, etc. We asked the participants to film their videos as if they are in front of a mirror trying on clothes, which involves various postures and backgrounds. We randomly selected three different clothes for each video, resulting in a total of 93 different clothes used for evaluating the 31 videos. We also use the VITON test data [24] to inspect *MIRROR* on a curated dataset.

VTO Quality Metrics. For comprehensive evaluation, we measure the quality of the VTO results for both individual frames and entire videos. First, each VTO frame is evaluated using Learned Perceptual Image Patch Similarity (LPIPS) [53] that is widely used to evaluate synthesized images. However, two videos can have significantly different VTO quality even if their VTO frames have similar LPIPS; if the erroneous locations change frequently in different VTO frames, the video looks more noisy and is recognized more erroneous. To capture the time-domain smoothness, an entire VTO video is evaluated using Fréchet Video Distance (FVD) [17, 46]. The lower LPIPS and FVD, the better the VTO quality.

[4]The implemented code can be found at https://github.com/Ds-Kang/MIRROR.

Fig. 13. Thumbnails of our real-world video dataset collected from 31 participants.

Table 2. *MIRROR* performance on Samsung Galaxy S10 that shows each component's effectiveness: (1) Region of Interest (RoI), (2) background removal (BR), (3) mask output (MO), (4) parsing input (PI), (5) parsing loss (PL), (6) lightweight warping (LW), (7) lightweight interpolation (VITOFF), and (8) storages (Context+Torso). The lower the metrics, the better the performance.

Method			VITON	Real World (30-second Videos)			
VTO Network	**Lightweight Interpolation**	**Storage**	**LPIPS**	**LPIPS**	**FVD**	**Average Latency (ms)**	**Conversion Time (min)**
PF-AFN (Baseline)	N/A	N/A	0.250	0.682	26.59	3041±342	57.3
RoI	N/A	N/A	0.250	0.134	13.61	5204±308	121.6
RoI + BR	N/A	N/A	0.251	0.090	9.00	5175±358	119.2
RoI + BR + MO	N/A	N/A	0.249	0.093	8.08	5306±320	123.1
RoI + BR + MO + PI	N/A	N/A	0.247	0.087	6.87	5350±343	119.0
RoI + BR + MO + PI + PL	N/A	N/A	**0.243**	**0.086**	**6.57**	5343±284	122.8
RoI + BR + MO + PI + PL + LW (DIVTON)	N/A	N/A	0.244	0.088	6.96	**4367±302**	**100.3**
DIVTON	VITOFF (DOF [19])	N/A	N/A	0.120	11.00	1649.2	39.2
DIVTON	VITOFF (w/o NMS)	N/A	N/A	0.091	4.14	377.3	7.32
DIVTON	VITOFF	N/A	N/A	**0.090**	**4.11**	**375.0**	**7.20**
DIVTON	VITOFF	Context+Torso	N/A	**0.090**	**4.11**	**170.1**	**2.85**

8.1 Overall Quantitative Analysis

We sequentially add each component in *MIRROR* to the baseline PF-AFN to investigate its impact. We measure LPIPS and FVD under the VITON dataset (source domain) and our real-world dataset (target domain). We also measure average per-frame latency and the end-to-end video conversion time on Samsung Galaxy S10. Table 2 shows the results.

8.1.1 Effectiveness of DIVTON. First of all, the baseline performance is significantly degraded in the real-world dataset (2.73× worse LPIPS), confirming PF-AFN's vulnerability to domain shifts. In contrast, our coarse-grained human semantics-based deep preprocessing techniques (i.e., RoI and BR) remarkably improve VTO quality over the baseline in terms of both LPIPS and FVD in the real-world dataset. RoI crops the upper body and locates it at the center of an image, as a VITON image, which achieves 5.1× better LPIPS than vanilla PF-AFN. Whitening noisy backgrounds in BR further improves LPIPS 32% and FVD 34%. The results verify the effectiveness of DIVTON's deep preprocessing on resolving format disparities between wild images and VITON images. Next, both image and video qualities are improved when gradually adding the DIVTON components: mask output (MO), parsing input (PI), and parsing loss (PL). The improvement is higher in the real-world data than the VITON data, and higher in FVD than LPIPS. The results highlight that our methods are particularly helpful for domain generalization and smooth video VTO in the time domain. In addition, our lightweight warping (LW) module reduces latency by 18% without significantly sacrificing the quality.

On the other hand, the quality gain in DIVTON comes with additional computation due to the use of a DNN-based coarse-grained human parser, which nearly doubles the video conversion time compared to PF-AFN. This clearly shows the reason why VITOFF is needed for on-device video VTO.

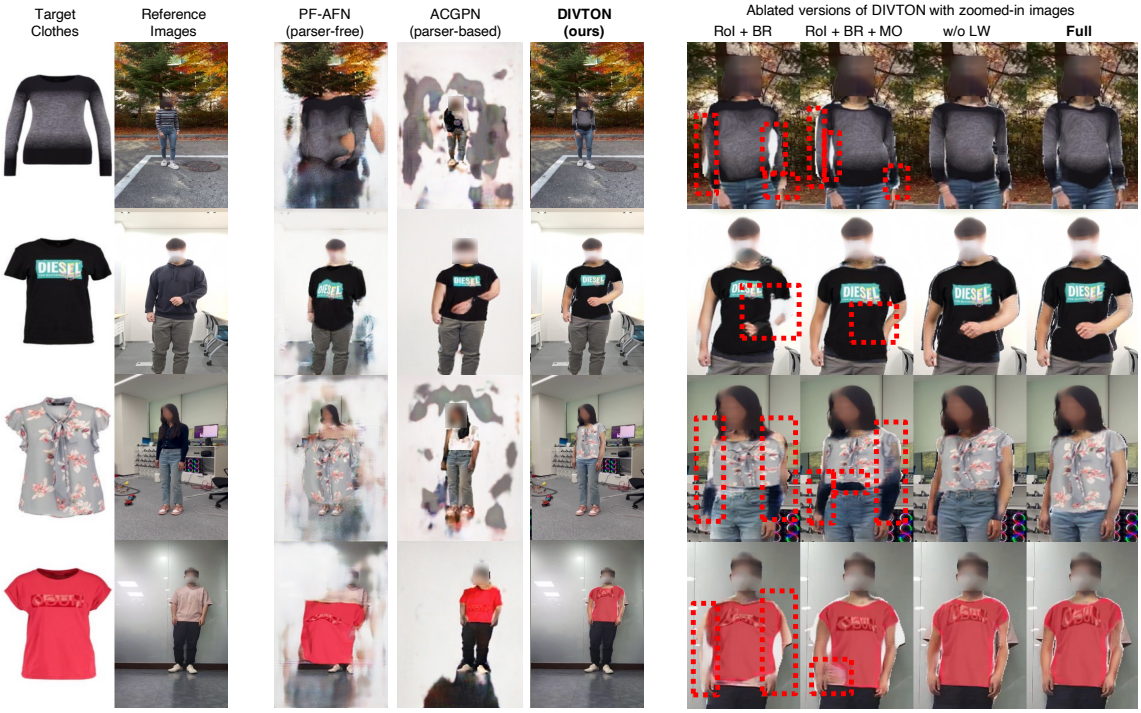
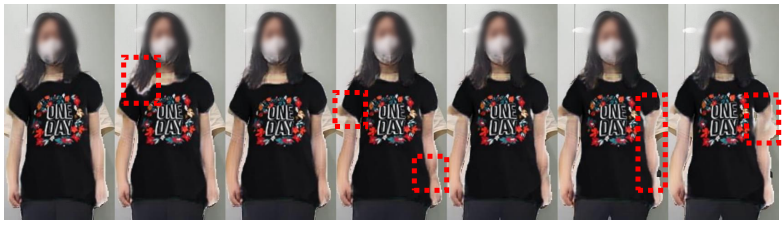


Fig. 14. Qualitative analysis comparing DIVTON (ours) with PF-AFN, ACGPN, and ablated DIVTON versions. The red boxes highlight identified errors, encompassing human body shape distortion, challenges in distinguishing the body from the background, and difficulties in discerning arms from other body parts.

8.1.2 Effectiveness of VITOFF and Storage. Interestingly, combining VITOFF with DIVTON remarkably reduces the number of DIVTON executions, making almost all frames in a video (93.6%) processed by DNN-less lightweight tracking. This verifies that our keypoint selection using coarse-grained human semantics results in good-quality keypoints, enough for robust VTO tracking without executing DIVTON. With much less DNN executions, VITOFF reduces the latency 11.6 times compared to the DIVTON-only case, requiring only several minutes for end-to-end video conversion. In addition, VITOFF even improves the VTO video quality (FVD) 41% over DIVTON by generating a smoother VTO video with careful use of spatiotemporal information.

In contrast, an ablated version of VITOFF with an existing non-DNN dense optical flow (DOF) technique [19], significantly degrades LPIPS and FVD compared to the proposed VITOFF with selective keypoint-based warping flow (and even worse than the DIVTON alone) and has less improvement in latency. This is because without keypoint extraction, VITOFF (DOF) utilizes only the WE error metric and fails to catch unsatisfactory tracking results in many cases. Another ablation VITOFF (w/o NMS) selects keypoints using coarse-grained human semantics but naïvely depending on high scores without using NMS. To ensure a fair comparison, we used 20 keypoints, which corresponds to the average number of keypoints selected when using NMS. VITOFF (w/o NMS) also underperforms our full VITOFF in LPIPS and FVD scores, primarily because the selected keypoints fail to represent the complete pixel content of the RoI in areas devoid of keypoints. The ablation study shows superiority of our tailored design for VITOFF.

Finally, utilization of our context and torso storages results in 2.5× further acceleration of *MIRROR*'s video conversion at runtime, verifying the effectiveness of the two-phase operation of *MIRROR*.



(a) DIVTON, which causes errors at different locations

(b) *MIRROR*, which causes correlated errors between frames.

Fig. 15. Qualitative results for consecutive frames.

8.2 Qualitative Analysis

To evaluate the qualitative results of DIVTON and VITOFF, we uploaded a video comparing the baseline (PF-AFN [20]), another existing parser-based GAN (ACGPN [49]), DIVTON-only, and *MIRROR* (DIVTON and VITOFF) in various cases.[5]

8.2.1 Effectiveness of DIVTON components. Figure 14 demonstrates the qualitative results of PF-AFN, ACGPN and our DIVTON using four examples: from top to bottom, (1) a relatively easy case, (2) an arm put on the body,(3) misalignment in the length of arm and the length of upper clothing, and (4) different width between original and target short sleeves. In all cases, the main challenges are to adapt to various body sizes and locations and distinguish the body from the background and the arms from the other parts.

As shown in Figure 14, vanilla PF-AFN completely fails in all cases since it naïvely assumes that the human body is located at the center of an image with a certain size. ACGPN better localizes the human body by using the parsing inputs but still fails due to parsing errors. In contrast, DIVTON produces remarkably improved VTO results in all cases. Taking a deeper look, with our human semantics-based deep preprocessing (RoI + BR), VTO results are significantly improved but the arms are still not distinguished correctly, which causes various errors (red dotted boxes). Adding MO prevents the background from invading the body part and adding PI and PL detects the arms most accurately. Lastly, LW does not significantly harm the VTO quality.

8.2.2 Effectiveness of VITOFF. Figure 15 compares the DIVTON-only case with *MIRROR* when converting consecutive frames. Using only DIVTON for all frames ignores temporal information, which causes erroneous locations (red boxes) to change frequently as time proceeds. In contrast, *MIRROR* generates correlated errors in the time domain, resulting in a smoother VTO video. This qualitative example explains why *MIRROR* provides similar LPIPS but significantly better FVD compared to using only DIVTON in Table 2.

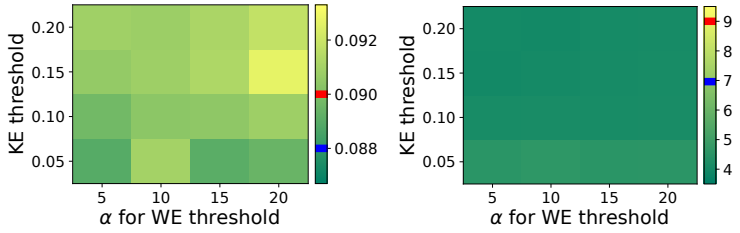
8.3 Impact of Coarse-Grained Human Parser

Table 3 evaluates VTO performance of DIVTON when fine- and coarse-grained parsing labels (PPP [10]) are used to train the deep human parser in DIVTON, respectively. For fine-grained parsing labels, we utilize LIP [36] that labels the VITON dataset and another widely used ATR [37]. The results in Table 3 show that DIVTON

[5] https://youtu.be/CrwxUW8cXQQ

Table 3. Impact of using using coarse-grained and fine-grained parsing labels on DIVTON.

Method	VITON	Real-world Dataset	
	LPIPS	LPIPS	FVD
Baseline (PF-AFN)	0.250	0.682	26.59
DIVTON with ATR [37] (fine-grained)	0.245	0.110	10.42
DIVTON with LIP [36] (fine-grained)	0.249	0.124	9.85
DIVTON with PPP [10] (coarse-grained, our choice)	**0.244**	**0.088**	**6.96**



(a) Image quality (LPIPS). (b) Video quality (FVD).

Fig. 16. Impact of the WE and KE parameters. In the color bars, red and blue lines are PF-AFN and DIVTON performance, respectively, both after preprocessing. The lower the metrics, the better the quality.

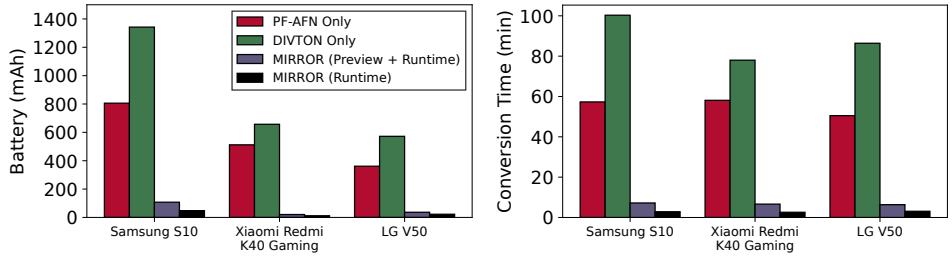
outperforms the baseline PF-AFN when using all the three label options, which verifies the effectiveness of using a deep human parser for domain generalization instead of direct image generation. On the other hand, both LIP and ATR perform comparably to PPP (our choice) in the VITON test dataset but significantly degrade DIVTON performance in terms of both LPIPS and FVD in the real-world dataset. This is because fine-grained human segmentation is vulnerable to domain shifts, causing nontrivial parsing errors in the real-world dataset. The results demonstrate that coarse-grained human semantics effectively serves as a domain-invariant feature for generalizable VTO.

8.4 Impact of Hyperparameters

Figure 16 analyzes the impact of VITOFF parameters in Section 6.3, the KE threshold and the artificial pixel movement α for WE threshold. First, *MIRROR*'s FVD is significantly better than PF-AFN with our deep preprocessing (red bar) and DIVTON (blue bar) regardless of these parameters since VITOFF utilizes temporal information effectively. On the other hand, *MIRROR*'s LPIPS is sensitive to the two parameters; using tighter thresholds increases VTO image quality at the expense of computation burden. Considering the trade-off, we aim to minimize the computation burden while providing LPIPS performance similar to PF-AFN with our human semantics-based preprocessing. To this end, we set the KE threshold to 0.10 and α to 10 pixels for the results in Table 2.

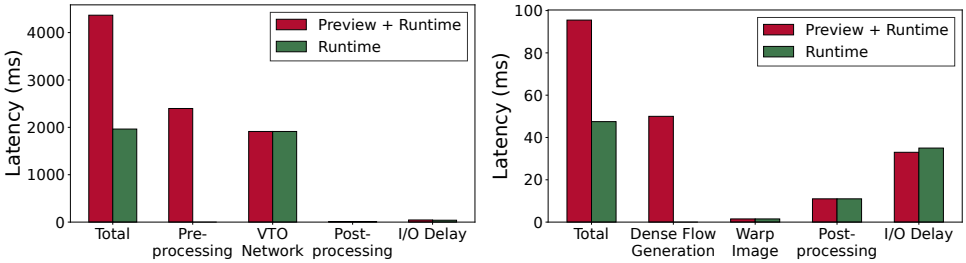
8.5 Computational Efficiency

Next, we evaluate *MIRROR*'s on-device computation overhead in terms of latency and battery consumption. In our target scenario, mobile devices have to perform VTO computations in the background while the user is shopping and present the VTO videos before the checkout when a decision is made. Since DIVTON (DNN only) consumes significant battery and conversion time (Figure 17), we prioritize computational efficiency in VITOFF to suit our scenario. The battery consumption is measured by the Android battery historian.



(a) Battery consumption. (b) Video conversion time.

Fig. 17. Computational overhead for video VTO with three smartphones.



(a) DIVTON operations. (b) VITOFF operations.

Fig. 18. Latency for each operation in *MIRROR*.

8.5.1 Analysis on End-to-end Video Conversion. Figure 17 shows the battery consumption and total video conversion time with three commodity smartphones (Samsung Galaxy S10, Xiaomi Redmi K40 Gaming, and LG V50) and a 30-second long video. We compare the four methods: PF-AFN only (i.e., baseline), DIVTON only, *MIRROR* (Preview + Runtime), and *MIRROR* (Runtime). The results are consistent in different smartphones as follows: First, running a VTO GAN for the entire video takes a huge computational overhead (PF-AFN), even more so with the use of a deep human parser (DIVTON). On the other hand, *MIRROR* (Preview + Runtime) remarkably reduces both battery consumption and conversion time with lightweight video interpolation in VITOFF. Excluding the preview phase from *MIRROR* by using the context and torso storages further reduces computation overhead. Overall, *MIRROR* (Runtime) results in an order of magnitude less computation on all three smartphones compared to the baseline, e.g., 16.9× less battery consumption and 20.1× faster video conversion in Samsung Galaxy S10.

8.5.2 Operation-Wise Analysis. We further analyze the impact of *MIRROR*'s two-phase design on each operation in DIVTON and VITOFF (Figure 18). The results show that introducing the preview phase eliminates the most costly operation in DIVTON and VITOFF at runtime: deep preprocessing and dense flow generation, respectively. The effective information recycling reduces the runtime latency of *MIRROR* by more than half.

8.5.3 Time-Series Analysis. An observation in Table 2 is that VITOFF reduces video conversion time more than average latency. For example, *MIRROR*-runtime reduces conversion time and latency 35.2× and 25.7× compared to using only DIVTON, and 2.5× and 2.2× compared to *MIRROR* (Preview + Runtime). This is because when relying on DIVTON on a mobile device, its temperature gets higher in continuous processing. As the device gets overheated, per-frame latency of DIVTON increases as well and gets much slower. In Table 2, average latency is measured in heating-free condition by cooling the device but video conversion time is measured with the device naturally heated.

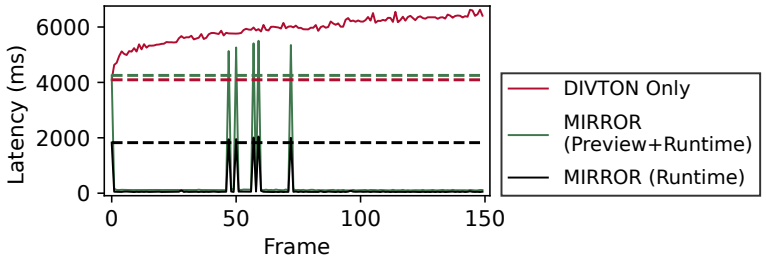


Fig. 19. Time-series operations when processing a 150-frame video. For each method, latency for the first frame is marked with a dashed line for comparison. In the *MIRROR* variants, the frames with peak latency are processed with DIVTON and the other are processed with VITOFF.

Table 4. Memory and storage usage for the baseline, *MIRROR* and its variants.

Method	Memory Cost	Storage Cost per Frame	Conversion Time
Baseline (PF-AFN)	2.1 GB	345.96 kB	57.3 min
DIVTON Only	1.2 GB	345.96 kB	100.3 min
MIRROR (w/o storage)	0.8~1.2 GB	**345.96 kB**	7.20 min
MIRROR (w/o VITOFF-preview)	0.8~1.2 GB	353.78 kB	3.78 min
MIRROR	0.8~1.2 GB	607.51 kB	**2.85 min**

To confirm the reason, Figure 19 shows per-frame latency on Samsung Galaxy S10 as time goes by. DIVTON-only worsens per-frame latency as time proceeds since the device gets *heated*. While both *MIRROR* variants significantly reduce per-frame latency of DIVTON by using VITOFF mostly instead of DIVTON, only *MIRROR*-runtime has stable per-frame latency of DIVTON, completely free from the heating problem with *zero execution* of the deep human parser at runtime, which causes additional gain on conversion time in Table 2.

8.5.4 System Overhead Analysis. Table 4 analyzes memory and storage usage of the baseline (PF-AFN), *MIRROR* and its variants. While the baseline consumes 2.1 GB of memory during video conversion, DIVTON occupies 43% less memory (1.2 GB) owing to its lightweight warping module. Although the deep human parser added to DIVTON increases memory consumption by 0.4 GB, the impact of lightweight warping module prevails, resulting in significant reduction in memory usage. In addition, VITOFF further reduces the memory usage to 0.8 GB most of the time since DIVTON is rarely executed.

In terms of storage cost, the full two-phase version of *MIRROR* stores the RoI locations, and human body semantics and dense warping flow in the RoI that are extracted by the preview phase. It occupies 1.76× more storage (607.51 kB/frame) than *MIRROR* (w/o storage) that stores only the input video (345.96 kB/frame). The additional storage usage in *MIRROR* results in 2.5× faster conversion compared to *MIRROR* (w/o storage), meaning that *MIRROR*'s computation reduction prevails additional read/write cost at storage. Importantly, since *MIRROR* processes most of the frames (93.6%) via lightweight tracking in VITOFF, the additional storage cost mainly comes from VITOFF-preview. To confirm the impact, we make another variation *MIRROR* (w/o VITOFF-preview) that stores only RoI locations and human semantics given by DIVTON-preview, which nearly nullifies additional storage cost without a significant increase in conversion time.

9 LIMITATIONS AND FUTURE WORK

While providing substantial gain in terms of video VTO quality and computation overhead, as the first approach towards generalizable on-device VTO system, *MIRROR* has room to be improved further.

Towards Real-time Video VTO. Although our target application (Section 2.2) does not require real-time VTO, providing real-time video VTO can combine VTO with augmented reality, which has the potential to diversify VTO applications. Currently, the major bottleneck is the lack of full GPU execution for a VTO GAN, as discussed in Section 5.2. We believe that future endeavors in supporting GPU operations and the advances of mobile AI chips [40] and model compression techniques [51] would solve this problem. Utilizing resourceful servers by sharing only de-identified information can also be considered in the future [28].

Towards Diverse Views and High Resolution Images. The current *MIRROR* implementation supports various poses and backgrounds with users' front-side views. While the front-side views are most important, users might want to check the VTO results on their side and back. The latest VTO DNNs [20, 26] do not support these views either due to lack of proper dataset, which can be resolved as VTO datasets are advanced. In addition, given recent efforts for high-resolution VTO [14], combining these DNNs with VITOFF would be interesting future work for on-device photo-realistic video VTO.

Towards Realistic Drape in VTO. From our survey in Section 2, we found that design, color, size, and texture are key factors causing inconvenience in online clothes shopping. While *MIRROR* focuses on design and color, it is still important to address proper sizing and realistic texture rendering to improve the user experience in VTO services. Previous work focusing on proper size fitting utilizes an avatar, which requires a 3D database of clothes [58] and an additional depth camera to approximate users' body size. However, these settings are not in typical online clothing shopping scenarios. Even after constructing the 3D database and measuring the user's body, 3D mesh-based cloth draping [3, 23] is computationally heavy on mobile environments. On the other hand, VTO capable of capturing the texture of garments requires computer graphics-based texture rendering techniques [33] for accurate results. However, employing such methods on mobile devices suffer from significant computational overhead.

Although image-based VTO has relatively unexplored this regime, a few pieces of recent work have investigated proper person/clothing sizing [13] and texture representation [38]. With the initial attempts, we envision that the issues regarding size fitting and texture representation can be resolved gradually in the future by relavent datasets and advanced deep learning strategies. However, the existing methods consider neither domain generalization and computational burden, orthogonal to this work. Therefore, it would be valuable future work to combine the incoming advanced DNNs with the insights gained from our study, including how to address domain shifts in real-world datasets and reduce computational overhead.

10 CONCLUSION

The mobile online shopping market is growing rapidly. However, consumers still face the challenge of not being able to try on clothes before purchase. Although VTO GAN has emerged as a solution, offering consumers the VTO experience in their videos while securing privacy is challenging due to significant domain shifts and on-device computation burden. Therefore, we have investigated generalizable on-device video VTO for mobile clothes shopping. Based on a concrete scenario derived from a survey, we have proposed *MIRROR* that converts real-world videos fast and accurately, which comprises two-phase operations of a generalizable VTO GAN called DIVTON and a lightweight VTO tracking method VITOFF. With our tailored end-to-end design, *MIRROR* significantly improves the quality of video VTO with 20.1× faster conversion and 16.9× less energy consumption compared to the latest PF-AFN. While *MIRROR* achieves substantial performance improvement, a range of valuable yet unexplored challenges remains, such as full real-time capabilities, diversified viewpoints, high-resolution rendering, accurate size fitting, and realistic texture representation. As the first work on mobile-only domain-invariant video VTO, we hope that *MIRROR* can serve as a stepping stone to foster future advancements regarding these issues.

ACKNOWLEDGMENTS

This work was supported partly by the National Research Foundation (NRF) of Korea grant funded by the Korea government (MSIT) (No. RS-2023-00212780) and partly by the NRF of Korea grant funded by the Korea government (MSIT) (No. RS-2023-00222663).

REFERENCES

[1] Kittipat Apicharttrisorn, Xukan Ran, Jiasi Chen, Srikanth V Krishnamurthy, and Amit K Roy-Chowdhury. 2019. Frugal following: Power thrifty object detection and tracking for mobile augmented reality. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*. 96–109.

[2] Shuai Bai, Huiling Zhou, Zhikang Li, Chang Zhou, and Hongxia Yang. 2022. Single Stage Virtual Try-On Via Deformable Attention Flows. In *European Conference on Computer Vision*. Springer, 409–425.

[3] David Baraff and Andrew Witkin. 1998. Large steps in cloth simulation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*. 43–54.

[4] Serge Belongie, Jitendra Malik, and Jan Puzicha. 2002. Shape matching and object recognition using shape contexts. *IEEE transactions on pattern analysis and machine intelligence* 24, 4 (2002), 509–522.

[5] Fred L. Bookstein. 1989. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on pattern analysis and machine intelligence* 11, 6 (1989), 567–585.

[6] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. 2004. High accuracy optical flow estimation based on a theory for warping. In *European conference on computer vision*. Springer, 25–36.

[7] Thomas Brox and Jitendra Malik. 2010. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE transactions on pattern analysis and machine intelligence* 33, 3 (2010), 500–513.

[8] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2017. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7291–7299.

[9] Kaifei Chen, Tong Li, Hyung-Sin Kim, David E Culler, and Randy H Katz. 2018. Marvel: Enabling mobile augmented reality with low energy and low latency. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*. 292–304.

[10] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. 2014. Detect what you can: Detecting and representing objects using holistic models and body parts. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1971–1978.

[11] Stephanie Chevalier. 2021. Share of online traffic worldwide 2020, by device. https://www.statista.com/statistics/296695/preferred-mobile-payment-service-providers-mature-markets/

[12] Stephanie Chevalier. 2021. U.S. fashion e-commerce value 2015-2021. https://www.statista.com/statistics/736612/fashion-e-commerce-market-usa/

[13] Yunmin Cho, Lala Shakti Swarup Ray, Kundan Sai Prabhu Thota, Sungho Suh, and Paul Lukowicz. 2023. ClothFit: Cloth-Human-Attribute Guided Virtual Try-On Network Using 3D Simulated Dataset. *arXiv preprint arXiv:2306.13908* (2023).

[14] Seunghwan Choi, Sunghyun Park, Minsoo Lee, and Jaegul Choo. 2021. Viton-hd: High-resolution virtual try-on via misalignment-aware normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14131–14140.

[15] Yousung Choi, Ahreum Seo, and Hyung-Sin Kim. 2022. ScriptPainter: Vision-based, On-device Test Script Generation for Mobile Systems. In *2022 21st ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 477–490.

[16] ONNX Runtime developers. 2021. ONNX Runtime. https://onnxruntime.ai/. Version: 1.10.0.

[17] Haoye Dong, Xiaodan Liang, Xiaohui Shen, Bowen Wu, Bing-Cheng Chen, and Jian Yin. 2019. Fw-gan: Flow-navigated warping gan for video virtual try-on. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 1161–1170.

[18] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. 2015. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*. 2758–2766.

[19] Gunnar Farnebäck. 2003. Two-frame motion estimation based on polynomial expansion. In *Scandinavian conference on Image analysis*. Springer, 363–370.

[20] Yuying Ge, Yibing Song, Ruimao Zhang, Chongjian Ge, Wei Liu, and Ping Luo. 2021. Parser-free virtual try-on via distilling appearance flows. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8485–8493.

[21] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 580–587.

[22] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. 2018. Densepose: Dense human pose estimation in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7297–7306.

[23] Erhan Gundogdu, Victor Constantin, Amrollah Seifoddini, Minh Dang, Mathieu Salzmann, and Pascal Fua. 2019. Garnet: A two-stream network for fast and accurate 3d cloth draping. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 8739–8748.
[24] Xintong Han, Zuxuan Wu, Zhe Wu, Ruichi Yu, and Larry S Davis. 2018. Viton: An image-based virtual try-on network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7543–7552.
[25] Stefan Hauswiesner, Matthias Straka, and Gerhard Reitmayr. 2011. Free viewpoint virtual try-on with commodity depth cameras. In *Proceedings of the 10th International Conference on Virtual Reality Continuum and Its Applications in Industry*. 23–30.
[26] Sen He, Yi-Zhe Song, and Tao Xiang. 2022. Style-Based Global Appearance Flow for Virtual Try-On. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3470–3479.
[27] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* 2, 7 (2015).
[28] Sunwook Hwang, Youngseok Kim, Seongwon Kim, Saewoong Bahk, and Hyung-Sin Kim. 2023. UpCycling: Semi-supervised 3D Object Detection without Sharing Raw-level Unlabeled Scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 23351–23361.
[29] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. 2017. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2462–2470.
[30] Thibaut Issenhuth, Jérémie Mary, and Clément Calauzènes. 2020. Do not mask what you do not need to mask: a parser-free virtual try-on. In *European Conference on Computer Vision*. Springer, 619–635.
[31] Jianbin Jiang, Tan Wang, He Yan, and Junhui Liu. 2022. ClothFormer: Taming Video Virtual Try-on in All Module. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10799–10808.
[32] Robert Keys. 1981. Cubic convolution interpolation for digital image processing. *IEEE transactions on acoustics, speech, and signal processing* 29, 6 (1981), 1153–1160.
[33] Pramook Khungurn, Rundong Wu, James Noeckel, Steve Marschner, and Kavita Bala. 2017. Fast rendering of fabric micro-appearance models under directional and spherical gaussian lights. *ACM Transactions on Graphics (TOG)* 36, 6 (2017), 1–15.
[34] Gaurav Kuppa, Andrew Jong, Xin Liu, Ziwei Liu, and Teng-Sheng Moh. 2021. ShineOn: Illuminating Design Choices for Practical Video-based Virtual Clothing Try-on. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 191–200.
[35] Peike Li, Yunqiu Xu, Yunchao Wei, and Yi Yang. 2020. Self-correction for human parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020).
[36] Xiaodan Liang, Ke Gong, Xiaohui Shen, and Liang Lin. 2018. Look into person: Joint body parsing & pose estimation network and a new benchmark. *IEEE transactions on pattern analysis and machine intelligence* 41, 4 (2018), 871–885.
[37] Xiaodan Liang, Si Liu, Xiaohui Shen, Jianchao Yang, Luoqi Liu, Jian Dong, Liang Lin, and Shuicheng Yan. 2015. Deep human parsing with active template regression. *IEEE transactions on pattern analysis and machine intelligence* 37, 12 (2015), 2402–2414.
[38] Anran Lin, Nanxuan Zhao, Shuliang Ning, Yuda Qiu, Baoyuan Wang, and Xiaoguang Han. 2023. FashionTex: Controllable Virtual Try-on with Text and Texture. *arXiv preprint arXiv:2305.04451* (2023).
[39] Bruce D Lucas, Takeo Kanade, et al. 1981. An iterative image registration technique with an application to stereo vision. Vancouver.
[40] Keondo Park, You Rim Choi, Inhoe Lee, and Hyung-Sin Kim. 2023. PointSplit: Towards On-device 3D Object Detection with Heterogeneous Low-power Accelerators. In *Proceedings of the 22nd International Conference on Information Processing in Sensor Networks*. 67–81.
[41] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019).
[42] Joseph Redmon and Ali Farhadi. 2018. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767* (2018).
[43] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. 2011. ORB: An efficient alternative to SIFT or SURF. In *2011 International conference on computer vision*. Ieee, 2564–2571.
[44] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. 2019. Models matter, so does training: An empirical study of cnns for optical flow estimation. *IEEE transactions on pattern analysis and machine intelligence* 42, 6 (2019), 1408–1423.
[45] Zachary Teed and Jia Deng. 2020. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*. Springer, 402–419.
[46] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. 2018. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717* (2018).
[47] Bochao Wang, Huabin Zheng, Xiaodan Liang, Yimin Chen, Liang Lin, and Meng Yang. 2018. Toward characteristic-preserving image-based virtual try-on network. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 589–604.
[48] Ran Xu, Chen-lin Zhang, Pengcheng Wang, Jayoung Lee, Subrata Mitra, Somali Chaterji, Yin Li, and Saurabh Bagchi. 2020. ApproxDet: content and contention-aware approximate object detection for mobiles. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*. 449–462.
[49] Han Yang, Ruimao Zhang, Xiaobao Guo, Wei Liu, Wangmeng Zuo, and Ping Luo. 2020. Towards photo-realistic virtual try-on by adaptively generating-preserving image content. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

7850–7859.

[50] Juheon Yi, Sunghyun Choi, and Youngki Lee. 2020. EagleEye: Wearable camera-based person identification in crowded urban spaces. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*. 1–14.

[51] Jiseok Youn, Jaehun Song, Hyung-Sin Kim, and Saewoong Bahk. 2022. Bitwidth-Adaptive Quantization-Aware Neural Network Training: A Meta-Learning Approach. In *European Conference on Computer Vision*. Springer, 208–224.

[52] Jinrui Zhang, Deyu Zhang, Xiaohui Xu, Fucheng Jia, Yunxin Liu, Xuanzhe Liu, Ju Ren, and Yaoxue Zhang. 2020. MobiPose: Real-time multi-person pose estimation on mobile devices. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*. 136–149.

[53] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 586–595.

[54] Wuyang Zhang, Zhezhi He, Luyang Liu, Zhenhua Jia, Yunxin Liu, Marco Gruteser, Dipankar Raychaudhuri, and Yanyong Zhang. 2021. Elf: accelerate high-resolution mobile deep vision with content-aware parallel offloading. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*. 201–214.

[55] Zhilu Zhang and Mert Sabuncu. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems* 31 (2018).

[56] Feng Zhao, Qingming Huang, and Wen Gao. 2006. Image matching by normalized cross-correlation. In *2006 IEEE international conference on acoustics speech and signal processing proceedings*, Vol. 2. IEEE, II–II.

[57] Xiaojing Zhong, Zhonghua Wu, Taizhe Tan, Guosheng Lin, and Qingyao Wu. 2021. MV-TON: Memory-based Video Virtual Try-on network. In *Proceedings of the 29th ACM International Conference on Multimedia*. 908–916.

[58] Heming Zhu, Yu Cao, Hang Jin, Weikai Chen, Dong Du, Zhangye Wang, Shuguang Cui, and Xiaoguang Han. 2020. Deep Fashion3D: A dataset and benchmark for 3D garment reconstruction from single images. In *European Conference on Computer Vision*. Springer, 512–530.