MagicCloth: Protect User Privacy in AR Streaming

Yuming Hu[1], Mingyu Zhu[2], Qiao Jin[1], Feng Qian[1], Bin Li[2]
[1]Department of Computer Science and Engineering, University of Minnesota
[2]Department of Electrical Engineering, Pennsylvania State University

ABSTRACT

With the growing number of users, Augmented Reality (AR) privacy issues have aroused researchers' concerns. Our work focuses on the protection of privacy-sensitive objects in indoor settings. We proposed a novel privacy protection mechanism MagicCloth for AR live streaming. When the user covers the privacy-sensitive object with a specially designed cloth, MagicCloth can accurately track it in real time, and replace it with a virtual object based on its shape and size seamlessly. By wisely integrating lightweight 0-1 segmentation, pattern detection, and AR plane tracking techniques, our MagicCloth system achieves real-time performance on Android smartphones. We extensively analyze the challenges involved in object replacement and have put forth corresponding solutions. In our future work, we aim to implement and refine these proposed solutions to effectively address the identified challenges and achieve seamless and privacy-preserving object replacement in AR scenarios.

CCS CONCEPTS

• **Security and privacy → Privacy protections**; • **Human-centered computing → Mobile devices**.

KEYWORDS

Augmented reality, visual privacy, object replacement

ACM Reference Format:
Yuming Hu, Mingyu Zhu, Qiao Jin, Feng Qian, Bin Li. 2023. MagicCloth: Protect User Privacy in AR Streaming. In *The 1st ACM Workshop on Mobile Immersive Computing, Networking, and Systems (ImmerCom '23), October 6, 2023, Madrid, Spain.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3615452.3617936

The current affiliation of Feng Qian is the University of Southern California.

ImmerCom '23, October 6, 2023, Madrid, Spain

ACM ISBN 979-8-4007-0339-3/23/10…$15.00
https://doi.org/10.1145/3615452.3617936

1 INTRODUCTION

Augmented reality (AR) privacy is an important aspect of AR technology that focuses on protecting users' privacy and mitigating potential privacy risks associated with AR experiences [21]. Several potential privacy problems can arise in AR live streaming. For instance, when sharing AR content, users may inadvertently display private items or sensitive information to the viewers [10]. The integration of virtual content (*e.g.,* mirror) with real-world objects can lead to unintended privacy violations, as virtual elements might interact with private spaces or personal belongings [26].

This paper focuses on the protection of privacy-sensitive objects (*e.g.,* a statue) in indoor scenarios, such as bedrooms. Furthermore, to avoid privacy leakage during data transmission (or on servers), we argue that the AR privacy protection system should completely run on the local smartphone. Note that we currently only consider privacy concerns for stationary objects. These factors bring us many challenges and requirements. 1) The user may move the smartphone around at any time, causing the camera to lose focus and produce blurry images. 2) Smartphones have limited computing resources. Live streaming typically requires 30 FPS, so heavy deep-learning models are impractical in this case. 3) We do not know what the privacy-sensitive object is or what its shape and size are. To provide a better user experience, a privacy-protection solution should better be out of the box. Protecting the privacy of individuals and objects within these controlled spaces requires careful consideration and effective privacy-preserving mechanisms.

Researchers have explored different approaches and techniques to safeguard the privacy of AR users. Hu *et al.* [10] proposed a fine-grained control framework, which splits the visual process and networking process. Users can choose whether to share their AR content with others at any time. However, in a real-world scenario, the user may not be aware that a private item appears on the smartphone screen. Zhu *et al.* [27] utilized smart LEDs for privacy protection. These LEDs flicker in a well-designed waveform, preventing unauthorized users from taking photos or videos. However, it cannot protect some individual privacy items alone. The most common solutions are vision-based (including neural network models) [11, 14, 15, 20, 24, 26], which are computationally heavy. Besides, we do not know what each user's privacy-sensitive objects are. Therefore, AR users might be required to collect images that contain the objects and retrain the models before using them. Diminished Reality (DR) [5] protects user privacy by removing the sensitive object. In addition to slow speed, DR often suffers from poor visual effects due to complex backgrounds and lighting conditions. Overall, these solutions are not suitable for AR live streaming, especially for large-scale daily use in indoor scenarios.

In this work, we proposed a new privacy protection mechanism MagicCloth for AR live streaming. The fundamental concept of MagicCloth involves three main steps. Firstly, the user covers a privacy-sensitive object with a cloth printed with unique patterns. Secondly, we leverage an ad hoc lightweight model to identify and monitor the movement of the cloth. Finally, we seamlessly replace the target with a virtual object, thereby ensuring privacy while providing a visually enhanced experience. The cloth itself can visually prevent viewers from seeing the privacy-sensitive object. Nevertheless, viewers might still know the geometric information of the object, which can leak private information. Our ultimate goal is to protect user privacy while maintaining a visual experience. Instead of adopting heavy DR solutions, we replace the target with a virtual object. In this way, we can obscure the target, hide its geometric information, and provide a better visual experience.

MagicCloth system mainly consists of three modules, including 0-1 segmentation, object replacement, and pattern detection. Note that we have only finished the segmentation and detection modules, and we will discuss our ideas about object replacement in §3. Different from traditional segmentation tasks [12], our 0-1 segmentation only needs to identify one type of target, *i.e.,* a piece of cloth with specially designed patterns. Therefore we can customize a lightweight neural network model to segment the cloth from the background. We then constructed a dataset containing everyday indoor scenes and trained our model based on it. Once obtaining the accurate location of the target, we can employ object replacement to generate a virtual object based on the target's size, shape and position. Unfortunately, our model is still time-consuming (~100 ms on smartphones) even though it is lightweight. So MagicCloth incorporates AR-specific tracking methods [8], allowing for accurate and consistent tracking of objects within the AR scene. It helps us update the location of the virtual object, but its tracking error accumulates over time and is sensitive to fast motion. Hence, we utilize pattern detection, which has a computation time of ~10 ms, to monitor the outcomes of plane tracking. Pattern detection is to locate the specially designed patterns on the cloth. If the locations of patterns are severely offset from that of the virtual object, we need to rerun the 0-1 segmentation model to regenerate the virtual object. By incorporating this pattern-based auxiliary detection tool, we enhance the robustness and reliability of our overall privacy preservation approach in indoor AR settings.

Our contributions are summarized as follows.

- We propose a novel framework MagicCloth for real-time privacy protection in AR live streaming.
- Lightweight segmentation model, pattern detection, and AR plane tracking are incorporated to achieve fast and accurate target tracking.
- We further discuss our idea of object replacement, which generates virtual items that are similar in shape and size to the target.

2 SYSTEM

In this part, we will talk about pattern detection, target tracking and 0-1 segmentation modules. The object replacement for visual privacy is discussed in §3.

2.1 Overview

To protect the privacy of users, we aim to replace the sensitive object with virtual objects. Before that, users need to cover the object with a cloth, on which the specially designed patterns are printed.

Next, we will talk about three parts of MagicCloth: 1) pattern detection, 2) target tracking, and 3) 0-1 segmentation. Since we cover the private object with cloth, we only need to detect the cloth rather than the object. A specific pattern (*i.e.,* annulus) is printed on the cloth, to accelerate the object detection, and details can be found in §2.2. As shown in Figure 1, our pattern detection will find the annulus on the cloth, further obtaining the target position and providing it to the AR module. It takes 10–30 ms (depending on the target number) to achieve pattern detection for Full HD images (*i.e.,* 1080P) on Android devices (*e.g.,* Samsung S21 Ultra), thus we leverage target tracking to reduce the latency. Our tracking method is based on MOSSE filter [3], and can locate the target in ~ 8 ms. The target location obtained through pattern detection and target tracking is only bounding-box level, which is insufficient for pixel-level object replacement or diminished reality (DR). Hence we utilize a transformer-based framework to achieve 0-1 segmentation, which distinguishes the target (*i.e.,* cloth) from the background. Besides, long-time AR plane tracking is unreliable, so we will regularly run the segmentation model to rectify the tracking result.

2.2 Pattern Detection

To improve detection accuracy, the pattern should be carefully chosen. A suitable pattern cannot be very large or small, and it needs to be simple. In this paper, we choose the annulus as our pattern (Figure 2), which is insensitive to rotation. Its dimension is set to be 10 cm × 10 cm.

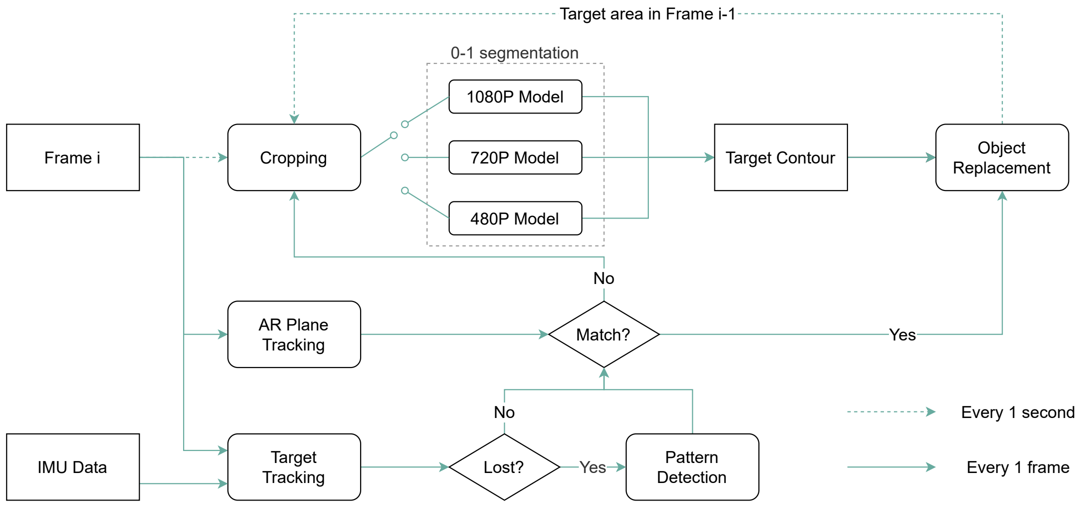

Figure 1: System overview.

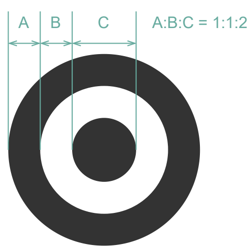

Figure 2: The annulus pattern.

During pattern detection, our target is to find any region in the image that is similar to our target pattern (*i.e.,* annulus). As shown in Figure 2, the length ratio of the black ring, white ring and central black ring is 1:1:2. For the middle row of the annulus pattern, there would be 5 consecutive black and white blocks, and their length ratio is 1:1:2:1:1 (denoted as *std_ratio*). For the middle column (*i.e.,* the vertical direction), there exists the same situation. Besides, colors (even in HSV space) can be greatly influenced by lighting conditions. Considering this factor, our pattern detection algorithm focuses on grayscale images, to enhance its robustness. The whole detection procedure contains 3 parts. 1) We scan each row to find the consecutive black and white blocks. 2) For the central black block, we then scan in the vertical direction. 3) If both the horizontal and vertical scan result shows this region is a candidate, we will further examine the similarity between this region and the template (*i.e.,* the annulus pattern). The detailed procedure is described as follows.

• Step ❶: In each row of the gray-scale image, the pattern detection algorithm searches for 5 consecutive blocks of black and white color (denoted as *row_blocks*). If the length ratio of *row_blocks* is similar to *std_ratio*, we then go to ❷.

• Step ❷: For the central black block of *row_blocks*, we will scan the column where its midpoint (denoted as *midpoint*) is located. In the vertical direction, we search for 5 consecutive blocks of black and white color (denoted as *col_blocks*). Note that *midpoint* is also located within the central black block of *col_blocks*. If the length ratio of *col_blocks* is similar to *std_ratio*, we then go to ❸.

• Step ❸: Now we can obtain the approximate bounding box (denoted as *bbox*) of the candidate region based on *row_blocks* and *col_blocks*. First, we crop the *bbox* region, and resize it to 32×32 pixels. Second, we leverage Pearson correlation [6] to calculate the similarity between this region and the template (*i.e.,* an annulus pattern), which is also 32×32 pixels in size. Note that the computation of correlation is relatively time-expensive, so we only compare the candidate region with our template if it passes the test in ❶ and ❷.

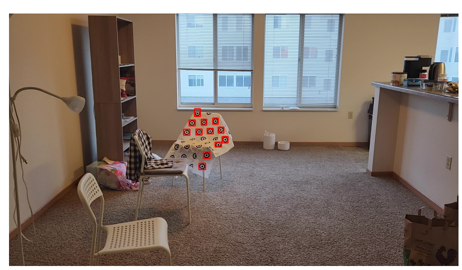

Figure 3: An example of pattern detection.

When the smartphone is not being moved, we only perform the pattern detection every 1 second to save energy. The detection frequency will be increased when the user moves faster. Moreover, in our algorithm, the "black" and "white" color is not defined by its pixel value, but the relative color difference between pixels. During the scanning, if the value of a pixel is *delta* smaller than the previous one, we will regard this pixel as black, and the previous one as white.

2.3 Target Tracking

Intuitively, the difference between two consecutive frames in a video is oftentimes relatively small. Especially, in the indoor AR streaming scenario, the user may even not move his/her smartphone. As a result, there is no need to conduct pattern detection in every frame. Instead, once we successfully detect the targets in a frame, we can choose to track them in the next few frames.

In this work, the target tracking is built upon the MOSSE filter [3]. Even though the tracker based on the MOSSE filter can achieve very high performance, there still remains a real-world challenge. Note that the speed of the tracker depends on its filter size. We can achieve almost 300 FPS in pattern tracking with a filter of 128 x 128 pixels. It is enough for tracking an object about the size of a chair at a distance of 5 m. However, when the user is close to the object (*e.g.,* < 1 m), the cloth can take up half of a frame. In this case, the tracking delay may exceed 1 second, which is obviously unacceptable. Our solution is based on the image pyramid, and the whole process is introduced next.

- Step ❶: For each frame in the video, we first convert it to grayscale, and then construct an image pyramid [1]. The pyramid contains a series of down-sampled images of the frame, including the 1080P, 540P, and 270P ones.
- Step ❷: If the bounding box of the target is ≥ 1/4 of the frame, we will track this target in the 270P image. When the bounding box is < 1/16 of the frame, the original 1080P image will be used for object tracking. In other cases, the 540P images are adopted during tracking the target.
- Step ❸: We leverage the MOSSE filter to track the target (*i.e.,* a cloth with a printed pattern), of which the initial position is obtained through the method described in §2.2.
- Step ❹: When the tracker shows we have lost our target, we will run the pattern detection immediately.

Although our tracking algorithm can quickly localize the targets, there still remain two main issues. First, the target size in each image will change when the user moves toward or backward. For instance, a user is 5 m away from the target (*i.e.,* cloth) at the beginning, and its bounding box is around 256 × 256 pixels in size. In this case, the initial MOSSE filter is set to 256 × 256 pixels. When the user moves toward the target, its bounding box can even take up nearly 50% of the original image. Note that this might happen in as little as 5 seconds. To maintain low tracking latency, we need to adapt the image resolution. Additionally, the filter should be resized according to image resolution. When we change the image resolution from 1080P to 540P, we need to reduce the filter size from 256 × 256 pixels to 128 × 128 pixels, and down-sample the filter content correspondingly.

Second, it is unnecessary to run the tracker all the time. 1) While the user/smartphone remains motionless, there is no need to update our estimate of the target's location in the image. 2) If the user/smartphone moves significantly, our tracker will no longer be able to locate the target. In general, it is only necessary to run target tracking when the user is moving moderately. Note that we only take into consideration the stationary objects. In this paper, IMU data (*i.e.,* accelerometer and gyroscope data) are used to estimate the motion state of the camera. We then decide whether to run target tracking based on the motion state. Due to measurement errors and noises, it is difficult to calculate the accurate motion of the smartphone-based just on IMU data. Therefore, we run a Kalman filter [2] to combine the tracking results with the IMU data.

2.4 0-1 Segmentation

Since the privacy object is covered by a piece of cloth, there is no need to implement classical semantic segmentation or instance segmentation [12]. We are only required to conduct 0-1 segmentation, which detects just one type of target, namely the cloth with annulus patterns. Besides, the target oftentimes only takes up a small portion of the image. In this case, we do not need to use the whole frame/image as the input of our segmentation model. We crop the *region* in the current frame that contains the target in the previous frame, and use this *region* (instead of the entire image) for segmentation. This *region* is set to be at least 4× the bounding box of the target in size. 3) Additionally, this model runs on smartphones. Under the premise of ensuring accuracy, it should be lightweight enough to reduce computation latency.

Figure 4: An example of the 0-1 segmentation result.

In this work, we employ a pre-trained SegFormer model as the foundation of our semantic segmentation method. SegFormer provides a powerful and efficient framework for semantic segmentation tasks, combining transformers and lightweight MLP decoders. We leverage the pre-trained weights of the SegFormer model, which have been trained on large-scale dataset ADE20k at resolution 512×512, to initialize our network. Then we fine-tune the SegFormer model on our dataset, adjusting the network parameters to specialize in the nuances and characteristics of our target semantic segmentation task. Combining the strengths of the pre-trained SegFormer model and the fine-tuning process, we achieve a robust and tailored semantic segmentation method that can accurately delineate the cloth region in our dataset.

As shown in Figure 1, we build three segmentation models with different input sizes, which is 1080P, 720P, and 480P, respectively. When cropping the *region*, we set its size to one of the above three resolutions. Then we choose the model according to the *region* size for 0-1 segmentation. Note that if multiple objects are found in the pattern detection, we will not crop the image, but use the 1080P model directly. To save power, we run the segmentation model every 1 second, updating the contour of the target.

3 OBJECT REPLACEMENT

In our research, we aim to seamlessly replace real-world objects with virtual objects to safeguard user privacy. Object replacement (OR) in AR scenarios consists of multiple steps. We begin by performing cloth detection and segmentation, where our focus lies in accurately identifying and isolating cloth within the scene. Once the cloth is detected and segmented, we employ diminished reality (DR) techniques to remove the cloth from the captured video feed.

Diminished reality (DR) techniques can create the illusion that the removed objects are no longer present in the scene, thus altering the user's perception of reality. Various approaches have been proposed in the literature to achieve DR, ranging from computer vision-based methods [7, 13] to advanced image processing algorithms [22]. Some studies have explored the use of techniques such as image-based rendering [17], texture rendering [16, 18], and pixel replacement [9] to seamlessly fill in the removed areas with suitable content that blends naturally with the surrounding environment. Jan *et al.* [9] have also investigated different applications of DR, including video editing and live streaming. Although the field of diminished reality is still relatively nascent, ongoing advancements in computer vision, image processing, and real-time rendering continue to push the boundaries of this technology, paving the way for more realistic and immersive augmented experiences. In our case, we apply DR to remove the cloth from the scene, creating the illusion that the cloth is no longer present.

In addition, Glen *et al.* [20] demonstrated the potential of combining ARCore [8] with DR techniques, which is still relatively less explored compared to AR. Most AR applications tend to focus on enhancing or adding virtual content to the real world rather than removing or diminishing real-world elements. And the approach proposed in their work does not involve object detection or tracking, while relying on manually defined regions of interest (ROI). This limitation restricts its applicability, particularly in scenarios involving video streaming. In our research, we intend to apply diminished reality techniques in the context of video streaming, coupled with semantic segmentation methods, robust tracking methods and ARCore plane tracking methods.

With the cloth successfully removed using diminished reality, we can then seamlessly replace the void left by the cloth with a virtual object. This involves integrating the virtual object into the AR scene in a visually coherent manner, ensuring it aligns with the surrounding environment and appears natural to the viewer. However, the process of object replacement through diminished reality poses several challenges. Achieving accurate and robust cloth detection and segmentation is crucial for seamless object removal. The detection and segmentation algorithms must account for variations in cloth appearance, lighting conditions, and occlusions. Additionally, ensuring smooth and realistic integration of the virtual object into the scene requires precise alignment, scale adaptation, and lighting harmonization.

To facilitate the object replacement process, we propose the establishment of an indoor object database. This database would contain a diverse collection of virtual objects that match various shapes and sizes. By leveraging this database, we can dynamically select a suitable replacement object based on the shape and size of the cloth that was removed. This allows us to seamlessly integrate the virtual object into the scene while preserving the privacy of the original target.

4 EVALUATION

4.1 Implementation

For running speed reasons, we implemented the key modules mainly in C++, and ran them on Android via NDK. The device we conducted performance tests on is the Samsung S21 Ultra, which runs Android 13 and supports Google ARCore [8]. Pattern detection and target tracking are based on OpenCV [4]. The segmentation model is built upon Pytorch. After training the model on the PC, we converted it to an Android version by using Pytorch Mobile [19].

We used the Segformer-B0 model pre-trained on ADE20K dataset and then fine-tuned it on our own dataset with an NVIDIA T4 GPU. Some weights were not initialized from the pre-trained model and are newly initialized since we only did 0-1 segmentation. During training, we applied auto-orient and set up crop size to 640×640 on our dataset. We trained the models using AdamW optimizer for 12K iterations and used a batch size of 8. During the evaluation, we report semantic segmentation performance using mean Intersection over Union (mIoU).

Dataset. Our dataset incorporates various challenging factors to capture real-world scenarios, involving different camera angles, distances, lighting conditions, and smartphone movements. Besides, the dataset is collected in two scenarios, including the living room and bedroom. It consists of about 1000 images, which we split into 70% for training, 20% for validation and 10% for testing. In the future, we will enrich our dataset by considering more situations like foreground objects partially occluding the cloth. Note that these images are extracted from the 1080P videos, which are captured by the Samsung S21 Ultra.

4.2 Performance

Module latency. Table 1 demonstrates the latency (*i.e.,* computation time) of each module. For pattern detection, target tracking and segmentation, their average latencies are 12.7 ms, 8.1 ms and 68.6 ms, respectively. The pattern detection algorithm in §2.2 has different computation times (*i.e.,* module latency) for different images. There might be many candidate regions passing Step ❶ and ❷, and the subsequent correlation computation in ❸ is time-consuming. Compared to others, the 0-1 segmentation module takes the longest computation time. Among these three modules, only the segmentation is built based on the neural network, which requires a large amount of calculation for online inference.

Table 1: The latency of each module. The filter size in target tracking is 250 × 250 pixels, and the input of segmentation model is 720P.

Pattern Detection	Target Tracking	Segmentation
12.7 ± 5.3 ms	8.1 ± 4.6 ms	68.6 ± 1.9 ms

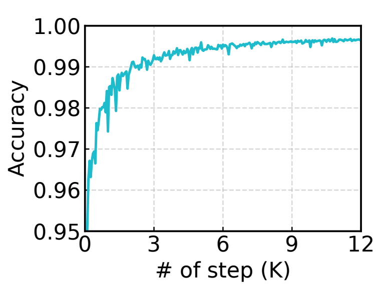

Figure 5: The accuracy of the 0-1 segmentation model.

Segmentation Accuracy. The pre-trained SegFormer was utilized in our segmentation module, to accelerate the offline training. Figure 5 plots the accuracy trend of our model during the training procedure. The accuracy can already achieve 95% at the 100th step, and finally reaches up to 99.7%. The testing results are shown in Table 2. Our model can obtain a mean IoU of 99.1%, while SegFormer can only achieve 37.4% IoU on average. Besides, the pixel-level accuracy of our model can even reach 99.6%. These results illustrate that after the adaptation in §2.4, our model can outperform the original SegFormer in terms of both accuracy and IoU when detecting the privacy object.

Table 2: The accuracy of segmentation results.

	0-1 Segmentation	SegFormer [25]
mIoU (%)	99.1	37.4

5 RELATED WORK AND CONCLUSION

Mixed Reality (MR). MR [23] can protect user privacy by replacing the target with some other objects. TransforMR [11] combines pose estimation, instance segmentation and video inpainting to achieve pose-aware object substitution. It builds a meaningful and interactive AR scene for users. However, TransformMR is compute-intensive and has poor real-time performance. Lindlbauer *et al.* [15] propose a Remixed Reality through live reconstruction of the 3D scene. To achieve that, they need to deploy multiple external depth cameras, limiting their daily usage.

Diminished Reality (DR). DR focuses on the removal or reduction of real-world objects or elements from a live video feed [5, 7, 13, 22]. Queguiner *et al.* [20] utilizes image inpainting to achieve DR, which does not need semantic segmentation. It requires that the clean 3D scene needs to be scanned beforehand, which is often impractical in real-world scenarios. VINet [14] leverages the deep neural network (DNN) to deal with video inpainting. Similar to other DNN models [24], it is difficult for VINet to realize real-time processing (*e.g.,* 30FPS) of 1080P video on Android smartphones.

Conclusion. In this work, we designed MagicCloth to preserve user privacy in AR streaming. It adopts a lightweight segmentation model to detect cloth with special patterns, achieving > 99% accuracy. The pattern detection algorithm helps rectify the target location. The cloth itself provides basic occlusion for objects. Combing these factors, MagicCloth effectively detects the target, and further leverages object replacement to protect/hide privacy-sensitive items.

ACKNOWLEDGMENTS

This work was supported in part by NSF grants: 1915122, 2106090, 2212298, 2152610, and 2152610.

REFERENCES

[1] Edward H Adelson, Charles H Anderson, James R Bergen, Peter J Burt, and Joan M Ogden. 1984. Pyramid methods in image processing. *RCA engineer* 29, 6 (1984), 33–41.

[2] Gary Bishop, Greg Welch, et al. 2001. An introduction to the kalman filter. *Proc of SIGGRAPH, Course* 8, 27599-23175 (2001), 41.

[3] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. 2010. Visual object tracking using adaptive correlation filters. In *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 2544–2550.

[4] Gary Bradski and Adrian Kaehler. 2008. *Learning OpenCV: Computer vision with the OpenCV library.* " O'Reilly Media, Inc.".

[5] Yi Fei Cheng, Hang Yin, Yukang Yan, Jan Gugenheimer, and David Lindlbauer. 2022. Towards understanding diminished reality. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. ACM, 1–16.

[6] Israel Cohen, Yiteng Huang, Jingdong Chen, Jacob Benesty, Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. 2009. Pearson correlation coefficient. *Noise reduction in speech processing* (2009), 1–4.

[7] Vasileios Gkitsas, Vladimiros Sterzentsenko, Nikolaos Zioulis, Georgios Albanis, and Dimitrios Zarpalas. 2021. Panodr: Spherical panorama diminished reality for indoor scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3716–3726.

[8] Google. 2023. AR Core. https://developers.google.com/ar.

[9] Jan Herling and Wolfgang Broll. 2012. Pixmix: A real-time approach to high-quality diminished reality. In *2012 ieee international symposium on mixed and augmented reality (ismar)*. IEEE, 141–150.

[10] Jinhan Hu, Andrei Iosifescu, and Robert LiKamWa. 2021. Lenscap: split-process framework for fine-grained visual privacy control for augmented reality apps. In *Proceedings of the 19th annual international conference on mobile systems, applications, and services*. 14–27.

[11] Mohamed Kari, Tobias Grosse-Puppendahl, Luis Falconeri Coelho, Andreas Rene Fender, David Bethge, Reinhard Schütte, and Christian Holz. 2021. Transformr: Pose-aware object substitution for composing alternate mixed realities. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 69–79.

[12] Anna Khoreva, Rodrigo Benenson, Jan Hosang, Matthias Hein, and Bernt Schiele. 2017. Simple does it: Weakly supervised instance and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 876–885.

[13] Daiki Kido, Tomohiro Fukuda, and Nobuyoshi Yabuki. 2020. Diminished reality system with real-time object detection using deep learning for onsite landscape simulation during redevelopment. *Environmental Modelling & Software* 131 (2020), 104759.

[14] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. 2019. Deep video inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5792–5801.

[15] David Lindlbauer and Andy D Wilson. 2018. Remixed reality: Manipulating space and time in augmented reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–13.

[16] Siim Meerits and Hideo Saito. 2015. Real-time diminished reality for dynamic scenes. In *2015 IEEE International Symposium on Mixed and Augmented Reality Workshops*. IEEE, 53–59.

[17] Shohei Mori, Momoko Maezawa, Naoto Ienaga, and Hideo Saito. 2016. Detour light field rendering for diminished reality using unstructured multiple views. In *2016 IEEE international symposium on mixed and augmented reality (ISMAR-Adjunct)*. IEEE, 292–293.

[18] Shohei Mori, Fumihisa Shibata, Asako Kimura, and Hideyuki Tamura. 2015. Efficient use of textured 3D model for pre-observation-based diminished reality. In *2015 IEEE international symposium on mixed and augmented Reality workshops*. IEEE, 32–39.

[19] PyTorch. 2023. PyTorch Mobile. https://pytorch.org/mobile/home.

[20] Glen Queguiner, Matthieu Fradet, and Mohammad Rouhani. 2018. Towards mobile diminished reality. In *2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE, 226–231.

[21] Franziska Roesner, Tadayoshi Kohno, and David Molnar. 2014. Security and privacy for augmented reality systems. *Commun. ACM* 57, 4 (2014), 88–96.

[22] Sanni Siltanen. 2017. Diminished reality for augmented reality interior design. *The Visual Computer* 33 (2017), 193–208.

[23] Maximilian Speicher, Brian D Hall, and Michael Nebeling. 2019. What is mixed reality?. In *Proceedings of the 2019 CHI conference on human factors in computing systems*. 1–15.

[24] Silvan Weder, Guillermo Garcia-Hernando, Aron Monszpart, Marc Pollefeys, Gabriel J Brostow, Michael Firman, and Sara Vicente. 2023. Removing objects from neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 16528–16538.

[25] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. 2021. SegFormer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems* 34 (2021), 12077–12090.

[26] Yiqin Zhao, Sheng Wei, and Tian Guo. 2022. Privacy-preserving Reflection Rendering for Augmented Reality. In *Proceedings of the 30th ACM International Conference on Multimedia*. 2909–2918.

[27] Shilin Zhu, Chi Zhang, and Xinyu Zhang. 2017. Automating visual privacy protection using a smart led. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*. 329–342.