EdgeMask: An Edge-based Privacy Preserving Service for Video Data Sharing

Samira Taghavi
Department of Computer Science
Wayne State University
Email: samira.taghavi@wayne.edu

Weisong Shi
Department of Computer Science
Wayne State University
Email: weisong@wayne.edu

***Abstract*—Preserving privacy in image and video data captured from public environments is essential for any research group that leverages, publishes, or shares such data. Although there are several research efforts attempting to resolve the privacy issues, they had quality and efficiency limitations. In this work, we proposed EdgeMask as a privacy preserving service that leverages edge computing and deep learning models to propose a real-time object segmentation approach and analyze the input data using parallel computing and speed up the object removal. Our experimental results indicate that EdgeMask reduces the computational time considerably.**

I. Introduction

Advanced driver-assistance systems (ADAS), Simultaneous localization and mapping (SLAM), and smart cities [11] are required to process the images and videos that are often captured from public environments. Captured images and videos may contain sensitive personal identifying data (i.e., individual's faces or license plates) that could violate the privacy of those individuals. Any using or sharing these raw image or video datasets can have legal implications.

Despite some efforts in recent years, privacy protection of video and image data captured from public places still remains a challenging problem and the limitations of existing approaches in terms of quality and efficiency prevent them to be applicable in practical applications. Several research efforts tried to address the privacy protection problem for only street-view data [3]. However, there are still some problems regarding these approaches.

Firstly, some of approaches tried to blur human faces [3], [6] that can't guarantee the privacy of individuals. Since, additional human characteristics including cloths, hair-style, location, other objects that are moving by person leads to the person identification.

Secondly, Uittenbogaard et al. [15] tried to remove pedestrians and vehicles based on Generative Adversarial Networks (GAN) that achieves good results in the image domain but is not applicable in video domain due to the lack of temporal constraints modeling [17]. Finally, the overall response time of existing approaches is slow (i.e. in the rage of seconds) that is not suitable for real-time image or video processing. Leveraging instance segmentation such as Mask R-CNN [4] or Generative Adversarial Networks [15] to process each frame degrades the efficiency significantly and increases the processing time to at least 200 ms per frame.

Instead of applying Mask R-CNN on each frame, we used optical flow [8] to extract the segmentation of vehicles and pedestrians. Mask R-CNN is used whenever new static object is shown in the scene or the direction of the camera pose is changed significantly. In addition, we feed the generated masks to an deeply inpainting approach [17] to remove undesired objects. In order to maintain the *Coherence* among different video frames and acquire original background information, we used a deep inpainting approach that processes the different video frames at once to propagate context information from *the known regions* to *the unknown regions* to improve the inpainting results [17].

Completely removing undesired objects not only can be used as an alternative for privacy preserving but also can be leveraged in visual SLAM solutions to improve the localization and loop closure [4] Also, previous approaches tried to remove unnecessary information from the images (e.g., all of the vehicles in the scene [15]). We believe removing many information from the image and video data is not necessary due to the existence of many instances of the same vehicle brand or model in the real world scenarios. We believe instead of removing all vehicles, only a particular vehicle should be removed per the user necessity.

Edge computing [14], on the other hand, is one of the outstanding solutions that emerged in recent years. Edge computing transfers the light-weight computing processes to the edge of the network (i.e., to do computation close to the data resource) and transfers computationally expensive tasks to the stronger computational servers. To the best of our knowledge, the comprehensive approach that leverages both blurring and object removal techniques for protecting the privacy of individuals in practical applications is understudied. To address the above-mentioned challenges we propose "EdgeMask". Our main contributions include:

- To the best of our knowledge, "EdgeMask" is the first work that proposes an edge-based privacy preserving framework that guarantees the real-time privacy preserving of individuals in public environments.
- We proposed a new efficient privacy technique that outperforms the existing systems by combining a top-down and bottom up segmentation solution that are used not only for efficient undesired object blurring but also for generating masks required for inpainting approach without requiring complex network architectures.

II. Reference Architecture

We proposed an edge-based architecture, EdgeMask, to address real-time privacy protection in street view data. Our

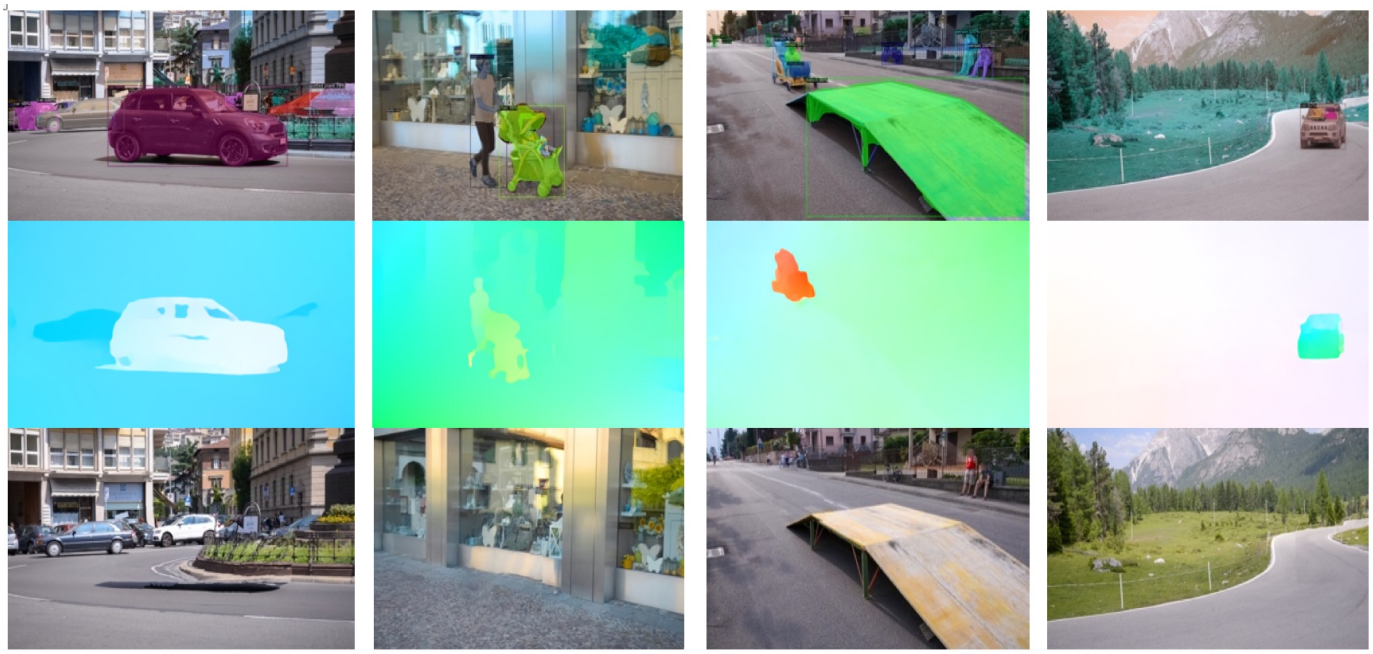

Fig. 1: Different steps of object masking and removal, the first row demonstrate the object mask and label, the second row is visualized optical flow, and third row is the final result for inpainting.

system automatically blurs and removes the pedestrians and vehicles in video and image data. As presented in figure 2, EdgeMask is a multi-layer architecture that consists of three different layers including: (1) the *Application Programming Interface* that manages the input and/or output and provides access to the final and/or intermediate results, (2) The *Edge-Intelligence* that implements the main analytical procedures and manages the different system modules, and (3) The *Security Layer* that prevents any unauthorized data access. The Edge-Intelligence consists of the two tiers including *Edge-Client* and *EdgeServer*. The EdgeClient blurs the sensitive data and is deployed at the edge of the network. The main idea for blurring the sensitive is based on light-weight object segmentation algorithm. Although, object segmentation is not accurate in this step, It guarantees that the sensitive data is completely masked. At the same time, It reduces the processing time. On the other hand, the EdgeServer is responsible for complex scenarios and the undesired object removal. At this step, everything is offline. We used more accurate object segmentation algorithm that is required to generate binary masks that are feed into object removal method.

A. Application Programming Interface

The *Application Programming Interface* controls the communication between the application's user and system and interprets the input commands in order to activate the system procedures. EdgeMask proposes the masking and removing as a service. Therefore, we introduced an standard API with different functionalities including "ObjectMask", "RemoveStaticObject" ,and "RemoveDynamicObject".

B. Edge-Intelligence

The *Edge-Intelligence* is a multi-component layer. All of the components run in parallel. Two main functionalities are the object blurring and the object removal. The object blurring is based on a bottom-up light-weight motion segmentation and top-down static object segmentation. The object removal is responsible for removing the undesired objects based on an inpainting approach [17].
Our functional flow includes six main parallel threads: parsing, loading, flow-extraction, segmentation, removal. The *parser* triggers the *EdgeClient* for object segmentation and blurring. The *EdgeClient* receives some clues from *EdgeServer* to mask the static objects. Simultaneously, the parser interprets the user commands and sends them to the *task planner*. The task planner redirects the received frames to the object segmentation [7] and the flow extraction [8] modules. The object segmentation generates the object segments associated with key-points, labels, and foreground scores. Also, It calculates a congestion value for each frame to find the scenes occupied with the undesired objects. The mask generator distinguishes objects and/or groups of objects based on the key-points and labels. The generator then produces the corresponding masks for the objects and/or group of objects. The batch organizer processes the video frames in this way: (1) It groups the frames and associated masks into the batch of frames, BOF, based on the foreground scores, (2) It analyzes the frames inside each BOF to observe if the camera has movements in both X and Y directions. If this is the case, then the BOF will be redirected to the task planner without applying the object removal. If no movement is present, then the inpainting module will be

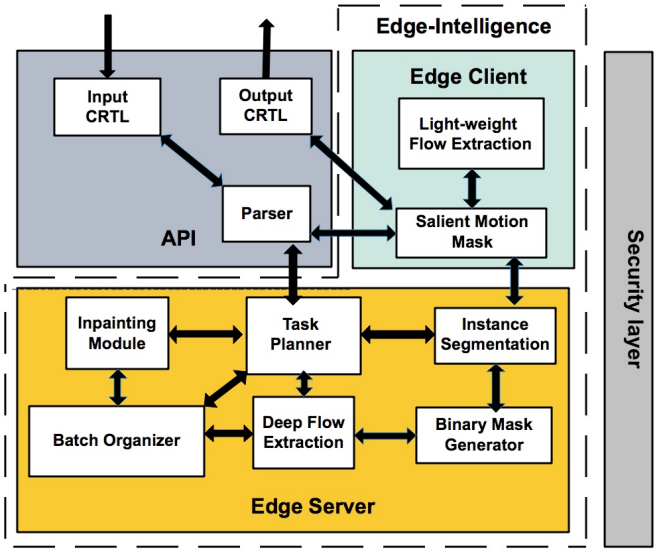

Fig. 2: EdgeMask System Architecture

triggered. Finally, the inpainting module [17] removes the undesired objects in two steps: (1) It identifies unknown regions and flows [8] and (2) It propagates the pixels from the known regions to the unknown regions based on the extracted optical flows. The propagation is applied in both the forward and backward directions. Then the successful inpainted frames are sent to the application programming interface as the system outputs.

C. Security Layer

Because of the network connection, privacy may be violated by any unauthorized third party agent that gets access to the data. Therefore, we used primitive and standard security techniques including the data encryption, the public-key cryptography, and the access control techniques to prevent any unauthorized access. The security layer is orthogonal to the other layers. This means that the security protocols should be applied on any network connection among the different system components. Therefore, each component establishes secure communication based on the *TLS/SSL* protocol. Also, any file transfer between different components should be based on the *SFTP* secure protocol. We didn't applied more advanced security methods [10] since It adds some overhead to the system and may increase the overall run time of the system.

III. CASE STUDY

The video and image captured from the real scenes may have different issues including: (1) The congested scene due to the presence of many undesired and sensitive objects. (2) The low video quality due to the rapid camera movement. (3) Any possible noise due to the weather condition (e.g., the snow or rain). The main goal of EdgeMask is to guarantee the privacy of individuals even if the video has such problems. In this section, we will explain the different analytical models and heuristics that are leveraged to manage different situations and speed up the overall performance.

A. Model Setup

In order to blur and remove undesired objects (i.e., the pedestrians and vehicles), we designed a pipeline of three different deep learning models with different computational costs. The first model in the pipeline is Flownet2.0 [8] that extracts the object flows in the real-time. The generated flows are clustered by a mathematical method to extract the dynamic regions of the scene. In order to extract the static object segments and required binary masks for the inpainting method, we used Mask R-CNN [7] proposed by Detectron2 [1] that extracts instance-based object segmentation.
For undesired objects removal, we compared the inpainting approaches based on the Peak Signal-to-Noise Ratio, PSNR, metric proposed by Liu et al. [9]. As presented in table II, PSNR for the approach proposed by Xu et al. [17] is 28.26 on DAVIS dataset [13] and is 27.49 for YouTube-VOS [16]. On the other hand, PSNR is 27.2 for the inpainting approach proposed by [15]. Therefore, we used the technique proposed by Xu et al. [17] to remove the undesired moving objects. The object removal [17] is based on optical flows extracted by FlowNet2.0 [8] to guarantee the coherence between different video frames. We used Res-Net101 as the backbone for the inpainting approach. In addition, for the static objects, we leveraged the image inpainting approach proposed by Yu et al. [18]. In table I, we reported the parameter setup used for different neural networks.

Model	Parameter	Value
Mask RCNN	Backbone Pretrained Weights	ResNet101 COCO Dataset
Inpainting	Backbone Flow Extraction Propagation Kernel Enlarged Mask Enlarged	ResNet101 FlowNet2 Bidirectional 50 or 70 YES
Flownet2.0	Backbone Model	ResNet101 FlowNet2.0

TABLE I: Deep Learning Model's Parameters

,

B. Hardware Setup

In order to build an edge-based solution, we used *Intel® Movidius™ Vision Processing Unit* to run object blurring at the edge of the network. We leveraged the *Intel Distribution of OpenVINO™ toolkit* [2] that includes optimized calls to *OpenCV*, *OpenCL™* kernels to run the object masking on *Intel Movidius™ Vision Processing Unit*. In this step, the objects are blurred based on a light-weight object segmentation

Approach	PSNR	SSIM	Coherence
Xu et al. [17]	28.26	76.22	YES
Uittenbogaard et al. [15]	27.20	68.00	NO

TABLE II: Comparison of inpainting approaches

Hardware	Model	Memory
GPU	NVIDIA GeForce GTX 1060	3G
VPU	Intel Neural Compute Stick 2	-

TABLE III: Hardware setup for EdgeMask

method using *OpenCV Gaussian Blur* operation. Also, we used a single GPU *NVIDIA GeForce GTX 1060 GPU* for the deep flow extraction and object removal.

C. Algorithm Design

In this section, we will elaborate the detailed algorithm that addresses the static and dynamic object blurring and removal in different scenarios. Our approach consists of the integration of a top-down and bottom-up approach.

1) Dynamic object segmentation: For the bottom-up solution, we applied the Flownet2 [8] on all video frames and extracted the optical flow between two consecutive video frames. We assumed that there isn't any sensor issue and data corruption. For the online video processing, the advantage of processing the video frames at the edge of the network, helps us to eliminate any possible network delay and the flow extraction will be quite rapid and falls in the range of 33 ms to 34 ms. Therefore for any dynamic object, we leveraged an statistical motion segmentation algorithm [19] to generate the object mask and make them anonymous via using blurring technique.

2) Light-weight motion segmentation: In order to estimate the dynamic regions, we used the approach proposed in [19] which is *salient motion mask* that extracts the dynamic regions by applying distance metrics on optical flow values estimated by Flownet2 [8] for each pixel. If we consider $F_t = \{F_1^t, F_2^t, ..., F_N^t\}$ as the optical flow field between two frames in which each element corresponds to the optical flow vector of each pixel in horizontal and vertical directions $F_t = \{u_i^t, v_i^t\}$, the global motion constrains is calculated as

$$S_i^t(F^t) = \sum_{\forall F_j^t \in F_t} d(F_i^t, F_j^t) \tag{1}$$

where $S_i^t \in [0, 1]$ and d$(\cdot)$ is the Minimum Barrier Distance (MBD) transform [19]. In the scenarios that camera is moving, an adaptive threshold method is applied and the pixels with low motion contrast are classified as background and removed from the segmentation. For the sake of efficiency, we did not further refine the optical flow for the scenarios with small number of undesired objects.

3) Static object segmentation: For the static object segmentation and masking, we can apply the Mask R-CNN to each frame and produce object masks but it can degrade the efficiency significantly, instead we assumed the video frames don't include any rapid object appearance. Therefore, as a top-down solution, for any static object, we used the Mask R-CNN [7] to generate the object mask for a single frame then propagate it to other frames by tracking camera motion. When the camera is static, we will track the object itself and remove the mask if object left it's relative location. It is obvious during the object movement, It is considered as dynamic object and will be masked by previously described dynamic object segmentation technique (see *Dynamic object segmentation* section).

4) Static object segmentation (moving camera): We feed the first frame into Mask R-CNN. If the instance masks generated by the Mask R-CNN for k different objects with "person" and "car" label [7] are denoted as $M_1, \ldots, M_k$ then the background mask will be $M_b = \overline{\{M_1, \ldots, M_k\}}$ [11]. Since the Mask R-CNN is trained on the COCO dataset [7] based on specific object labels, it is not capable of generating the segmentation of all types of moving objects such as stroller and soapbox in figure 1. Therefore, we used clues from optical flow [8]. The overlapping mask denoted as M_o is calculated by summing up the $\overline{M_b}$ with the mask generated by motion segmentation technique. Then, we used the optical value of pixels that have intersect with $\overline{M_o}$ to acquire the displacement and move the segmentation. We repeat this process when there is significant change in camera direction.

5) Congested scene object segmentation: The congested scene are determined based on the density of the scene. If the number of the instance masks generated by Mask R-CNN and labeled as "person" is more than 10, we classify the scene as congested scene. For the congested scenes, the optical flow generated by flownet2 [8] is not accurate. Therefore, we used the architecture proposed in [17] to refine the optical flow via using the coarse-to-fine refinement architecture to generate more accurate optical flows. Although the efficiency degrades, it helps to accurately masks undesired objects in complex scenarios.

6) Object removal: Different from blurring technique, object removal is performed in offline mode and the binary masks are generated by processing each frame and are the result of the intersection between $\overline{M_b}$ and optical flow based motion segmentation. These masks are feed into inpainting method [17] to remove undesired objects. In order to remove an specific object we extracted the object motion by the method described in [19] and object binary mask generated by the Mask R-CNN and restricted by object id and then feed the resulted mask into inpainting method [17].

IV. EXPERIMENTS

In this section, we presented the evaluation results of EdgeMask and compared it with previous approaches based on the quality and performance metrics.

A. Quality Analysis

We conducted our experiments on the Davis dataset [13] that is used by Xu et al. [17] to evaluate the video inpainting.

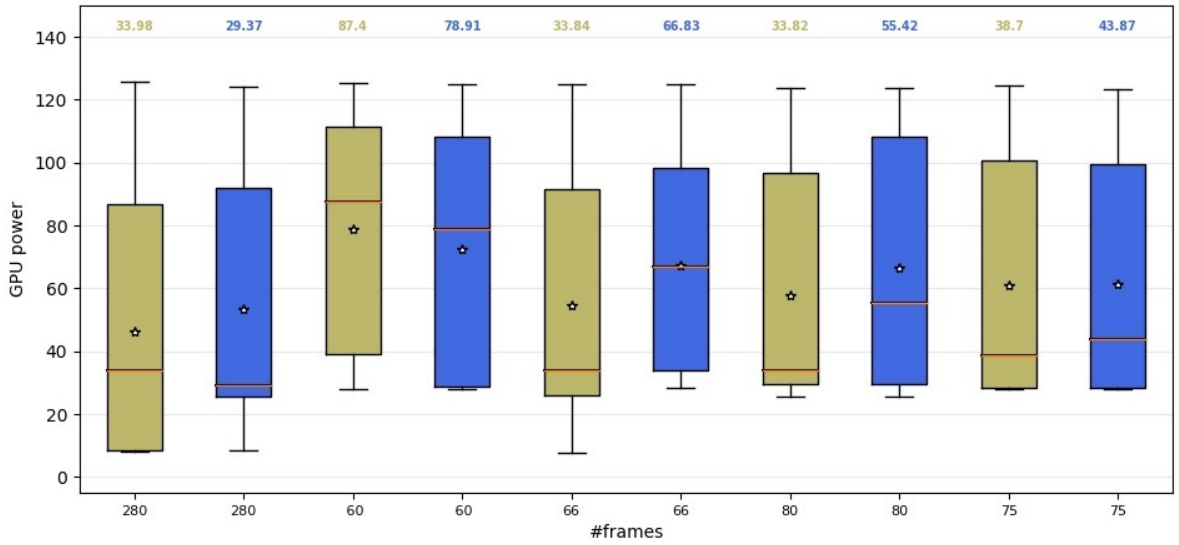

Fig. 3: The GPU power for parallel and serial computing units (#frames == 280). The blue box-plots present the distribution of the randomly sampled power rates and the green box-plots present the distribution of the original power rates.

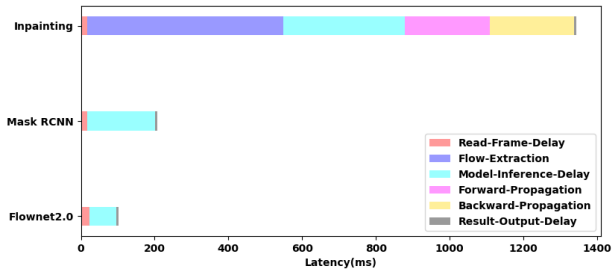

Fig. 4: The latency breakdown of the different DNNs

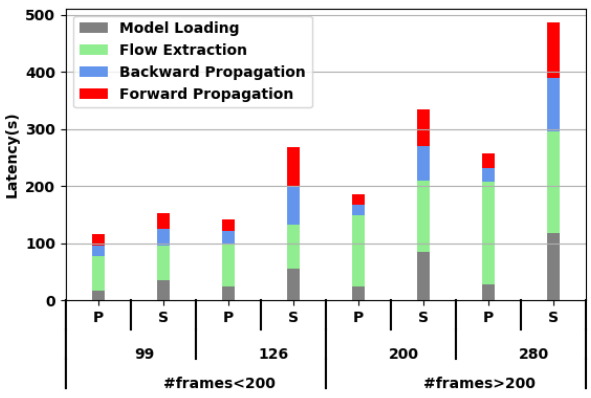

Fig. 5: The latency of parallel and serial computing units, P stands for parallel and S stands for serial.

Approach	Kernel 70	Kernel 50
Xu et al. [17]	27.838	28.011
EdgeMask	27.851	28.026

TABLE IV: Quality evaluation of EdgeMask. The kernel stands for the mask enlarge factor [17]

Approach	Object blurring	Object removal
Frome et al. [6]	10.1s	-
Flores et al. [5]	-	31.6s
Agrawal et al. [3]	12s	-
Nodari et al. [12]	-	21.4s
EdgeMask	80ms	1.16s

TABLE V: The computational time based on a single frame reported by each approach

The Davis dataset contains 150 different videos [17] in which the foreground objects are annotated manually. We compared the results of video inpainting for manually created annotations with the automatically generated annotations produced by EdgeMask. The evaluation results that are presented in table IV show that automatically generated masks fit to the inpainting technique [17].

B. Performance Analysis

We conducted the performance analysis using different baselines. (1) As presented in table V, EdgeMask reduces the computational time from multiple seconds to approximately one second for the object removal and to milliseconds for the object blurring in comparison to the different related approaches, (2) As presented in figure 5, EdgeMask improves the computational time of the leveraged video inpainting technique [17] by half due to the parallel computation, (3) The latency breakdown of three different Deep Neural Networks

presented in figure 4 shows that Flownet 2.0 [8] is fast enough to be considered as real-time (i.e., the maximum inference delay is 73 ms). In addition to the computational time, we reported the GPU power consumed by a single serial process in comparison to the all of parallel processes with the same input data that is presented in figure 3. The parallel processes have the maximum GPU power rates more than the serial process. One possible reason is due to simultaneously running on the GPU cluster. The overall maximum GPU power rate for all of the processes is close since all of them execute same computational unit. Also, the sum of the power rate for the parallel processes equals the serial process's power rate.

V. Related Work

There are several approaches focused on the blurring and removing sensitive data. Frome et al. [6] applied a fast window box detector for faces and license plate detection to gain high recall and then used different deep neural networks to remove all of the false positives. Flores et al. [5] removed the pedestrians from the street view images but their work can not handle more than one pedestrian in the scene. Agrawal et al. [3] provided a high-level framework about main concerns and issues of blurring techniques as well as an approach for blurring the humans faces and bodies. Nodari et al. [12] tried to remove the person of interest and replace the person of interest with a similar and anonymous object. Uittenbogaard et al. [15] tried to remove pedestrians and vehicles based on Generative Adversarial Networks (GAN) that achieves good results on image data.

VI. Conclusions

In this paper, we proposed an edge-based secure architecture to guarantee the privacy of individuals in videos and images captured from the public environments. Based on our experimental results, the object blurring is fast enough to meet the privacy preserving performance requirements. On the other hand, the removal process is not fast enough. Therefore, the object removal can be as an alternative for privacy preserving in offline mode. We used three different neural networks with different computational costs and proposed a new algorithm to efficiently segment and mask undesired objects in online mode. Also, we used parallel processing to speed-up the inpainting technique. Our experimental results indicate that we reduced the computational time significantly.

References

[1] Detectron2. https://gilberttanner.com/blog/detectron-2-object-detection-with-pytorch, 2019.

[2] The intel distribution of openvino toolkit. https://software.intel.com/en-us/openvino-toolkit, 2019.

[3] Prachi Agrawal and PJ Narayanan. Person de-identification in videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(3):299–310, 2011.

[4] Berta Bescos, José M Fácil, Javier Civera, and José Neira. Dynaslam: Tracking, mapping, and inpainting in dynamic scenes. *IEEE Robotics and Automation Letters, 2018*.

[5] Arturo Flores and Serge J. Belongie. Removing pedestrians from google street view images. pages 53–58, 2010.

[6] Andrea Frome, German Cheung, Ahmad Abdulkader, Marco Zennaro, Bo Wu, Alessandro Bissacco, Hartwig Adam, Hartmut Neven, and Luc Vincent. Large-scale privacy protection in google street view. In *2009 IEEE 12th international conference on computer vision*, pages 2373–2380. IEEE, 2009.

[7] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[8] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470, 2017.

[9] Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 85–100, 2018.

[10] Saeid Mofrad, Ishtiaq Ahmed, Shiyong Lu, Ping Yang, Heming Cui, and Fengwei Zhang. Secdataview: a secure big data workflow management system for heterogeneous computing environments. In *Proceedings of the 35th Annual Computer Security Applications Conference*, pages 390–403, 2019.

[11] Sundaram Muthu, Ruwan Tennakoon, Tharindu Rathnayake, Reza Hoseinnezhad, David Suter, and Alireza Bab-Hadiashar. Motion segmentation of rgb-d sequences: Combining semantic and motion information using statistical inference. *IEEE Transactions on Image Processing*, 29:5557–5570, 2020.

[12] Angelo Nodari, Marco Vanetti, and Ignazio Gallo. Digital privacy: Replacing pedestrians from google street view images. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 2889–2893. IEEE, 2012.

[13] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 724–732, 2016.

[14] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge computing: Vision and challenges. *IEEE internet of things journal*, 3(5):637–646, 2016.

[15] Ries Uittenbogaard, Clint Sebastian, Julien Vijverberg, Bas Boom, Dariu M Gavrila, et al. Privacy protection in street-view panoramas using depth and multi-view imagery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10581–10590, 2019.

[16] Ning Xu, Linjie Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and Thomas Huang. Youtube-vos: A large-scale video object segmentation benchmark. 2018.

[17] Rui Xu, Xiaoxiao Li, Bolei Zhou, and Chen Change Loy. Deep flow-guided video inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3723–3732, 2019.

[18] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5505–5514, 2018.

[19] Tao Zhuo, Zhiyong Cheng, Peng Zhang, Yongkang Wong, and Mohan Kankanhalli. Unsupervised online video object segmentation with motion property understanding. *IEEE Transactions on Image Processing*, 29:237–249, 2019.