Design of Privacy Preservation System in Augmented Reality

Yoonsang Kim, Saeed Boorboor, Amir Rahmati, Arie E. Kaufman

Stony Brook University

{yoonsakim,sboorboor, amir, ari}@cs.stonybrook.edu

Introduction

Augmented Reality (AR) capability to overlay virtual data on top of real-world objects and enable better understanding of visual inputs attract the attention of application developers and researchers alike. However, the privacy challenges associated with the use of AR systems is not sufficiently recognized. We present Erebus, a privacy-preserving framework designed for AR applications. Erebus allows the user to establish fine-grained control over the visual data accessible to AR applications. We explore the use cases of Erebus framework and how it can be applied to safeguard the privacy of the user's surroundings in AR environments. We further analyze the latency penalty imposed by Erebus to understand its effect on user experience.

Problem Statement

We suggest a scenario where a serious threat can be introduced. Any two or more clients with AR applications connected through network. An AR video chat application is a good example to illustrate the case. The application which heavily relies on real-time video data synchronization and communication.

Scenario: There is a possibility of a threat actor collecting data in between network-connected clients or eavesdropping data as a malicious remote client. To be specific, the malicious actor may obtain the private data of clients from their video camera inputs. However, we assume that the integrity of internal network security for the sender client is not compromised. In other words, the threat can only exist on two locations: The external network path of clients and at the receiving remote client-side

System Design & Implementation

Our base approach to solve the privacy exposure is by letting the sender clients process the privacy-sensitive information themselves. As illustrated in Figure 1, the client system is responsible for obscuring any privacy-sensitive data on its side before any data leaves the client's AR device. This approach can readily solve our threat scenario because the only data the attacker can acquire will be obscured data.

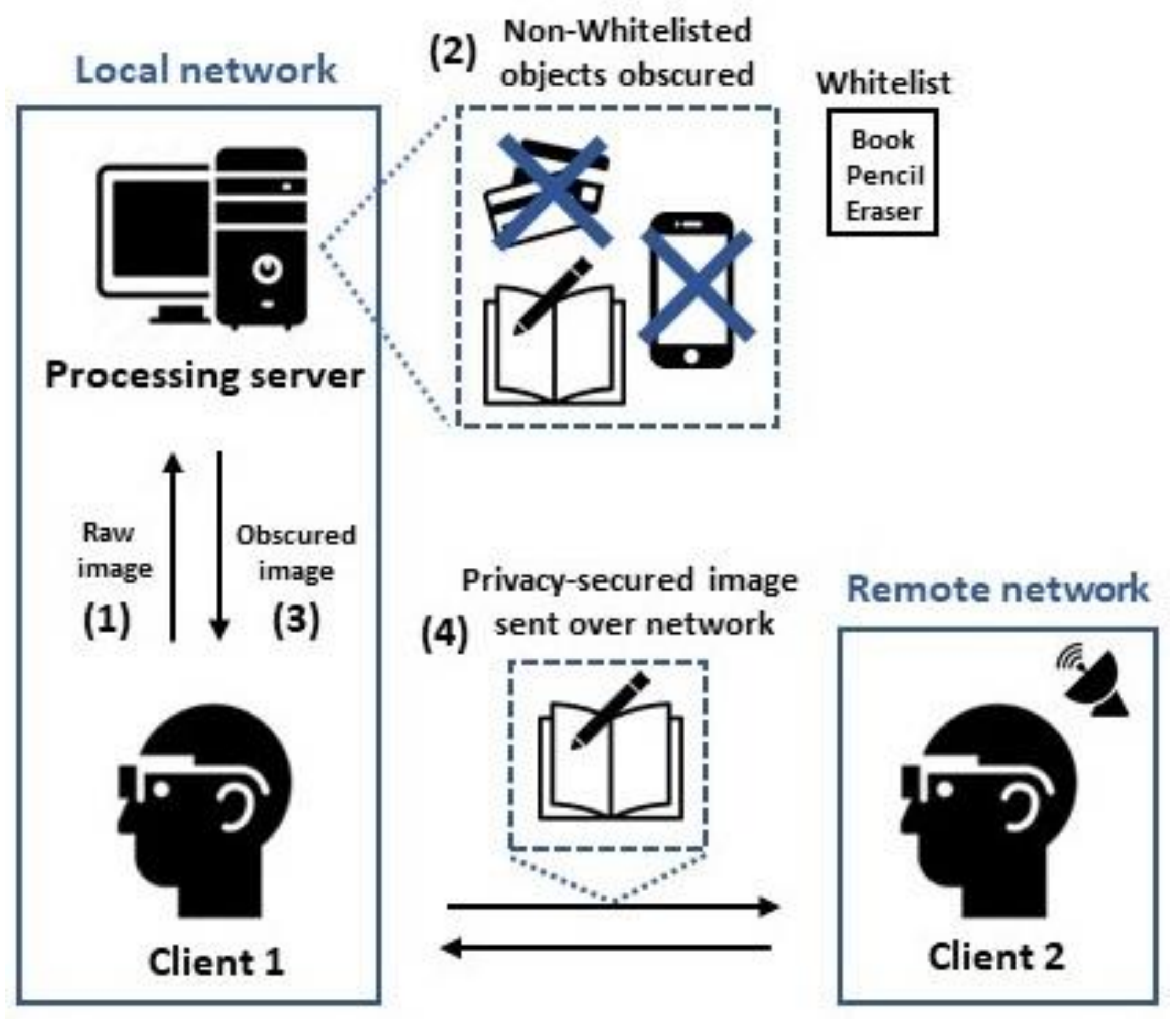

Figure 1. Architectural overview of Erebus and its four main steps

Whitelisting: Client maintains a list of object names to be classified as <u>Public</u> type objects. Objects that are not contained in this list are obscured and considered as <u>Private</u> type data. The concept of Whitelisting is visualized in Figure 2.

Data Obscuring: Obscuring user's data before it is transmitted to a remote client. Utilizing YoloV4 object detection model, we recognize objects and obscure them if they are <u>Private</u> data. Obscuring occurs on the bounding-box-level of each object.

Task Partitioning: We added a trusted device with superior computation power to the flow within the sender client's local network. This device is noted as Processing server. The concept enables distribution of GPU-heavy task such as object recognition, to a computationally superior device and relatively light-weight task such as network communication, to a client's mobile AR device.

System Transferability: We use Unity game engine for the client environment so that our application can utilized across different AR-capable devices. Also, because we allocate the object detection task to the Processing server as long as the format of returned the detection result is consistent, the object detection model can easily be replaced.

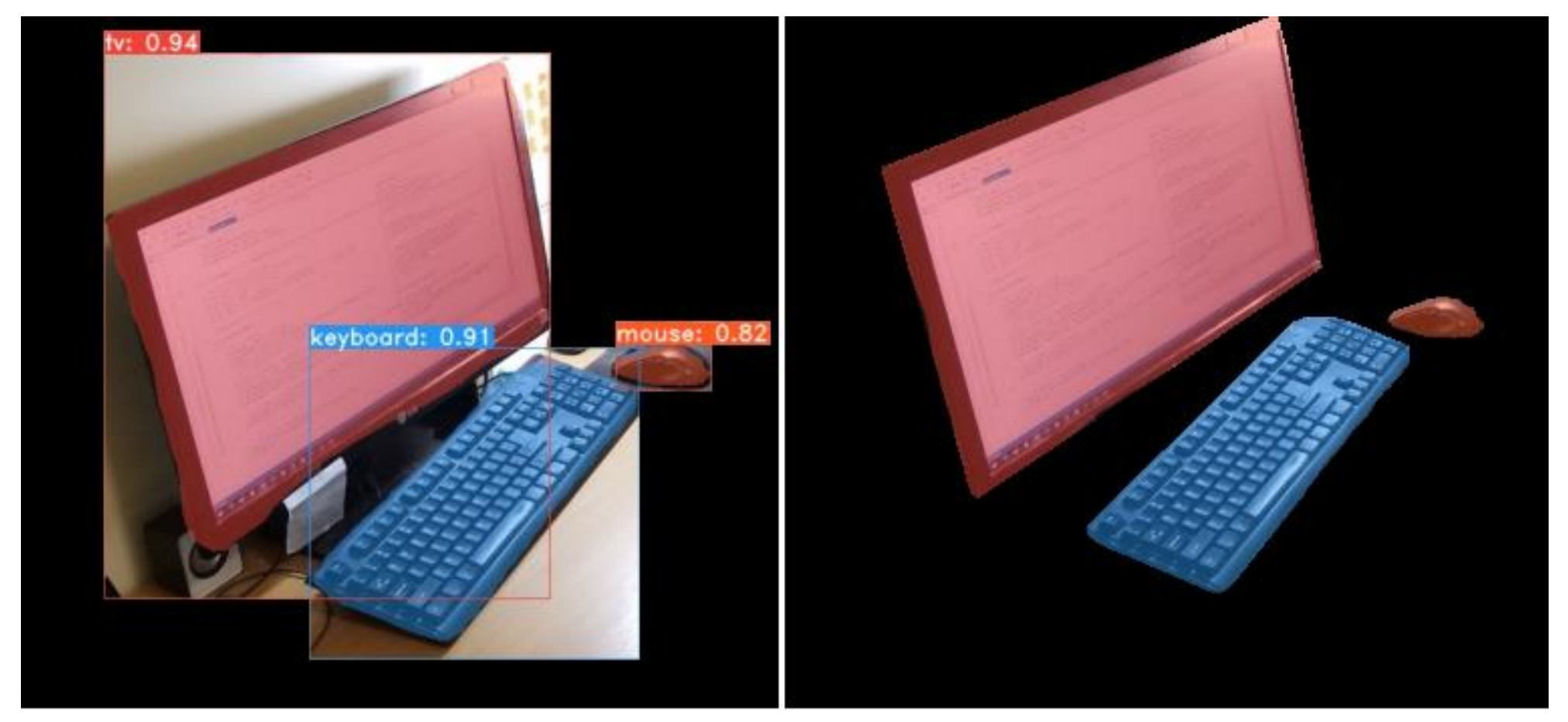

Figure 2. Concept visualization of Whitelisting (Left), Ideal Whitelisting method: Instance segmented (Right)

Results & Analysis

We analyze the latency of each step to identify the performance bottleneck of the application and to set the direction for our future studies. The most amount of time was consumed in the Networking step followed by the Object detection step. The remaining steps contributed 4.3% (Mean:13.4ms) to the total consumed time. The test was performed under the TCP protocol, with wireless network (Download:16.18Mbps, Upload:21.61Mbps), and NVIDIA GTX 960, Intel i5-8400, 16GB

Table 1. Time consumption analysis per step (ms)

Trial	COM	DCOM	NET	DET	PAR	WL	TOT
1	10	2	170	130	1	<1	313
2	10	3	158	125	1	<1	297
3	7	2	160	129	1	<1	299
4	12	4	171	126	<1	<1	313
5	9	3	169	126	1	<1	308

COM: Image compression; DCOM : Image decompression;
NET: Time spent on networking–send/recv–with processing server;
DET : Object detection/classification; PAR : Parse object detection result
WL: Time spent on Whitelisting; TOT: Total consumed time

As shown in Table 1, our analysis indicates that fast and stable network condition is crucial for our design. In addition, detection time needs to be improved. Thus, we plan to explore the possibility of minimizing network communication in the flow and the use of modest-sized but relatively accurate object detection model. This may require further research on a light-weight object detection model and the use of client device-level processing.

Conclusion

We proposed our preliminary research, Erebus, a framework for safeguarding user's privacy in AR environment. Our concept accentuates the importance of privacy preservation in AR applications and presents a crucial element to be considered in the AR development amidst the recent upswing trend of AR. For future works, we plan to improve our framework so that it can be applicable to real-time AR applications with acceptable magnitude of latency. Also, we expect to develop example applications to demonstrate our concept.

Acknowledgements

This work was supported by the Office of Naval Research Award No. N00014-20-1-2858