Demo Abstract: Adaptive AR Visual Output Security using Reinforcement Learning Trained Policies

Joseph DeChicchis
Duke University
Durham, NC, USA
joseph.dechicchis@duke.edu

Surin Ahn
Stanford University
Palo Alto, CA, USA
surinahn@stanford.edu

Maria Gorlatova
Duke University
Durham, NC, USA
maria.gorlatova@duke.edu

ABSTRACT

Augmented reality (AR) technologies have seen significant improvement in recent years with several consumer and commercial solutions being developed. New security challenges arise as AR becomes increasingly ubiquitous. Previous work has proposed techniques for securing the output of AR devices and used reinforcement learning (RL) to train security policies which can be difficult to define manually. However, whether such systems and policies can be deployed on a physical AR device without degrading performance was left an open question. We develop a visual output security application using a RL trained policy and deploy it on a Magic Leap One head-mounted AR device. The demonstration illustrates that RL based visual output security systems are feasible.

CCS CONCEPTS

• **Security and privacy → Systems security**; • **Computing methodologies → Mixed / augmented reality**; *Reinforcement learning*.

KEYWORDS

Augmented reality, visual output security, reinforcement learning, policy optimization, Magic Leap AR headset.

ACM Reference Format:
Joseph DeChicchis, Surin Ahn, and Maria Gorlatova. 2019. Demo Abstract: Adaptive AR Visual Output Security using Reinforcement Learning Trained Policies. In *The 17th ACM Conference on Embedded Networked Sensor Systems (SenSys '19), November 10–13, 2019, New York, NY, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3356250.3361935

1 INTRODUCTION

Augmented Reality (AR) is becoming increasingly ubiquitous. Research has shown that AR will be a $100 billion industry by 2020 [11] and companies are actively developing AR technologies for consumer and commercial use [2, 4, 7, 10]. While the increasing proliferation of AR devices will undoubtedly enable many new applications, issues of privacy and security cannot be ignored. AR security is concerned with both the inputs [5, 12] and outputs [1, 8, 9] of AR devices.

SenSys '19, November 10–13, 2019, New York, NY, USA

ACM ISBN 978-1-4503-6950-3/19/11…$15.00
https://doi.org/10.1145/3356250.3361935

In particular, visual output security is concerned with two issues pertaining to the user's visual field [1, 9]:

- Regulating visual content displayed to reduce distraction and prevent obstruction of the real-world context.
- Preventing holograms with a lower priority from obstructing holograms with a higher priority.

For example, in the case of displaying holograms in car windshields, it would be dangerous for a hologram to obstruct a stop sign. Similarly, a hologram which displays the speedometer should not be obstructed by a hologram which displays the album art of the song the driver is currently playing.

Although previous work has investigated an operating system enforced AR output security module which relies on developer written policies [9], we previously realized that, while promising, these hand-coded policies can be difficult to define for real-world use [1]. Instead, we proposed the use of reinforcement learning (RL) to generate visual output security policies through trial and error, demonstrating the RL approach's effectiveness in simulation [1].

While previous work has illustrated the importance of visual output security and demonstrated its feasibility in simulation [1, 9], it did not deploy policies on a physical AR device. To fill the gap, in this work we train a visual output security policy using RL and deploy it on a Magic Leap One head-mounted AR device [7] to demonstrate that RL trained models can be used as visual output security policies *without any noticeable performance degradation* (i.e. a drop in quality of experience due to factors such as a reduced frame rate). To our knowledge, although it has been feasible to deploy RL policies on Magic Leap One devices, no previous work has done so.

2 VISUAL OUTPUT SECURITY APPLICATION

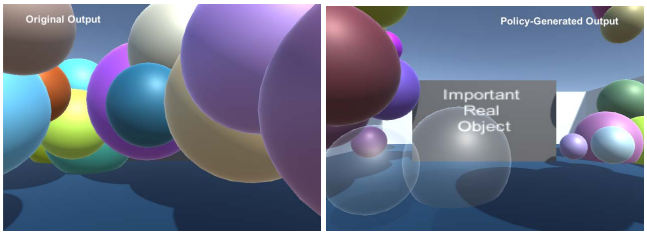

Figure 1: Example of visual output before (left) and after (right) a RL generated policy is applied (from Ahn et al. [1]).

Application Setup. A full visual output security application was developed to test the deployment of a RL trained visual output security policy on a Magic Leap One. The application was built

using Unity version 2018.1.9f2 [15] with the Magic Leap Lumin SDK version 0.19.0 [6]. The RL policy was trained and deployed using the ml-agents framework [14]. Training was conducted in a simulated environment. The goal of the training is to produce a policy which moves the holograms so that they do not obstruct the important real-world object while keeping the holograms as close to their original position as possible. Fig. 1 shows an example of a simulated visual output when a RL trained visual output security policy is applied [1].

Finding the Right RL Policy. Finding the best RL policy proved challenging. For example, rewarding an agent for moving holograms which obstruct an important real-world object to a new location which does not obstruct it resulted in an agent which learned to always move the holograms to the top right of the user's field of view. Therefore, the reward function was tweaked to penalize agents for moving holograms too far from their original location.

What worked best after several iterations was a policy which trained an agent to move the holograms by applying force rather than outputting a new position for the holograms. The agent used in this demonstration observes the location of the important real-world object and location and velocity of the holograms to calculate the x and y force for each hologram (i.e. the amount to move a hologram in a specific direction). The agent was rewarded for moving the holograms to a location which no longer obstructed the important real-world object. More reward was given the faster it achieved this goal.

Training the RL Policy. The proximal policy optimization algorithm was used to train the RL policy [13]. The important real-world object and holograms had a constant size and their locations were randomly initialized. The distance between a hologram and important real-world object needed for a hologram to be considered no longer obstructing the important real-world object was incrementally increased to aid model convergence.

Deploying the RL Policy. The image tracking library built into the Magic Leap One was used to recognize and track an important real-world object. A target with many features was needed for tracking to work reliably. In addition to the RL trained policy, two heuristics are used in the visual output security application:

- Only apply the RL trained policy if the hologram's original position obstructs the important real-world object.
- Once the RL trained policy is applied, move the hologram back to its original position as soon as its original position no longer obstructs the important real-world object.

3 INTERACTIVE DEMONSTRATION

In the interactive demonstration, participants wear a Magic Leap One device and see several large spherical holograms obstruct their view. Participants can move freely around their physical environment while wearing the Magic Leap One (Fig. 2). Once an important real-world object comes into view of the participant, the application detects the object in real-time and applies the RL generated policy, moving the holograms and revealing the important real-world object (Fig. 3). The holograms move back to their original position when they no longer obstruct the important real-world object. A video visualizing the RL policy during training is available at [3].

Figure 2: Interactive demonstration. A user is wearing a Magic Leap One. The printed image in the background is the important real-world object used in the demonstration.

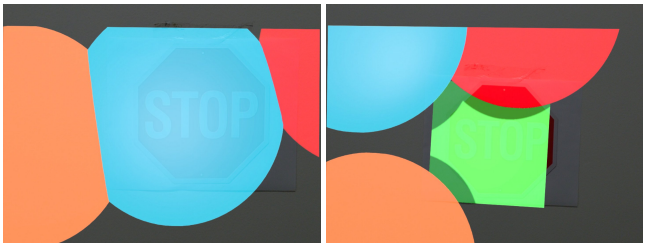

Figure 3: Example visual output before (left) and after (right) the security policy is applied. A green square is placed over the real-world object to indicate that it is tracked.

ACKNOWLEDGMENTS

This work was supported in part by the NSF award CSR-1903136 and by the Defense Advanced Research Projects Agency (DARPA) under contract No. HR001117C0048. The opinions, findings and conclusions expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency.

REFERENCES

[1] Surin Ahn, Maria Gorlatova, Parinaz Naghizadeh, Mung Chiang, and Prateek Mittal. 2018. Adaptive Fog-Based Output Security for Augmented Reality. In *Proc. ACM VR/AR Network Workshop'18.*
[2] Apple. 2019. ARKit. https://developer.apple.com/augmented-reality/
[3] Joseph DeChicchis, Surin Ahn, and Maria Gorlatova. 2019. Demo Video: RL Trained Visual Output Security Policy. https://youtu.be/WgOaaUE9wtQ
[4] Google. 2019. ARCore. https://developers.google.com/ar/
[5] Suman Jana, David Molnar, Alexander Moshchuk, Alan Dunn, Benjamin Livshits, Helen J. Wang, and Eyal Ofek. 2013. Enabling Fine-grained Permissions for Augmented Reality Applications with Recognizers. In *Proc. USENIX Security'13.*
[6] Magic Leap. 2019. Creator. https://www.magicleap.com/creator
[7] Magic Leap. 2019. Magic Leap One. https://www.magicleap.com/magic-leap-one
[8] Kiron Lebeck, Tadayoshi Kohno, and Franziska Roesner. 2016. How to Safely Augment Reality: Challenges and Directions. In *Proc. ACM HotMobile'16.*
[9] Kiron Lebeck, Kimberly Ruth, Tadayoshi Kohno, and Franziska Roesner. 2017. Securing Augmented Reality Output. In *Proc. IEEE S&P'17.*
[10] Microsoft. 2019. HoloLens. https://www.microsoft.com/en-us/hololens
[11] ABI Research. 2015. ABI Research Shows Augmented Reality on the Rise with Total Market Worth to Reach $100 Billion by 2020. https://www.abiresearch.com/press/abi-research-shows-augmented-reality-rise-total-ma/
[12] Franziska Roesner, Tadayoshi Kohno, and David Molnar. 2014. Security and Privacy for Augmented Reality Systems. *Commun. ACM* 57, 4 (April 2014).
[13] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. (2017).
[14] Unity. 2019. ml-agents. https://github.com/Unity-Technologies/ml-agents
[15] Unity. 2019. Unity. https://unity.com/