Demo Abstract: Catch My Eye: Gaze-Based Activity Recognition in an Augmented Reality Art Gallery

Tim Scargill
Duke University
Durham, North Carolina, USA
ts352@duke.edu

Guohao Lan
Delft University of Technology
Delft, The Netherlands
g.lan@tudelft.nl

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

ABSTRACT

The personalization of augmented reality (AR) experiences based on environmental and user context is key to unlocking their full potential. The recent addition of eye tracking to AR headsets provides a convenient method for detecting user context, but complex analysis of raw gaze data is required to detect where a user's attention and thoughts truly lie. In this demo we present Catch My Eye, the first system to incorporate deep neural network (DNN)-based activity recognition from user gaze into a realistic mobile AR app. We develop an edge computing-based architecture to offload context computation from resource-constrained AR devices, and present a working example of content adaptation based on user context, for the scenario of a virtual art gallery. It shows that user activities can be accurately recognized and employed with sufficiently low latency for practical AR applications.

CCS CONCEPTS

• **Human-centered computing** → **Mixed / augmented reality**.

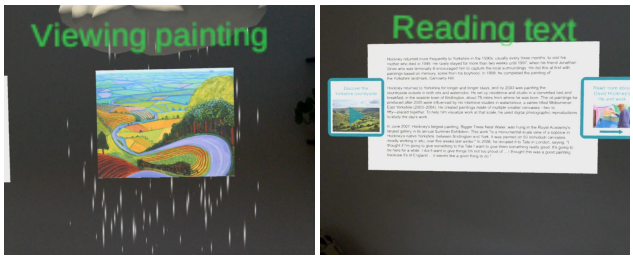
KEYWORDS

Augmented reality, eye tracking, activity recognition, edge computing, augmented reality art gallery

1 INTRODUCTION

Immersive art gallery and museum experiences are highly promising use cases for augmented reality (AR). These experiences may take place in an existing institution, such as those already trialed in the Kunsthalle Munich [7] and Tate Britain [4], or consist entirely of virtual content presented in the home or classroom, an approach which holds great potential in democratizing access to artworks. What both have in common is the capacity to present, alongside exhibits, additional virtual content, contextualized and personalized for each user, that improves engagement and educational outcomes.

However, we only wish to display additional virtual content that is relevant to the user at a given point in time; rendering too many AR objects at once has been shown to cause a drop in frame rate due to the associated resource demands, even on specialized devices like headsets [1], as well as cognitive overload of users in educational scenarios [3]. We therefore require a method of detecting which exhibit a user is currently engaging with. The addition of eye tracking to AR headsets such as the Magic Leap One and the Microsoft HoloLens 2 raises the prospect of employing a user's gaze for this type of context detection. Critically though, *just because a user is looking in the direction of an exhibit it does not mean they are paying attention to it.* Imagine you are having a conversation with someone standing in front of a painting: the direction of your gaze might indicate you are looking at the painting, but it would be distracting if additional virtual content started playing behind the other person.



(a) (b)

Figure 1: Screenshots from Catch My Eye in action on the Magic Leap One AR headset. Different content is displayed in a virtual art gallery depending on the user activity, i.e., viewing painting or reading the text, that is detected using the gaze signal captured by the eye tracker.

We propose a solution by supplementing gaze direction with gaze-based activity recognition and head pose, to determine if a user is truly engaging with the content they are looking towards. We developed a gaze-based activity recognition classifier for the case of a virtual art gallery in our accompanying paper, EyeSyn [6], and in this demo we present Catch My Eye, a full working system with a commercial AR headset and realistic virtual content. An illustration of Catch My Eye in action is shown in Figure 1. While gaze-based prediction of user intent has recently been demonstrated in virtual reality [2], to the best of our knowledge, *our work is the first to incorporate DNN-based activity recognition from user gaze into mobile AR, which we demonstrate in a context-aware app for the scenario of an augmented reality art gallery.* As shown in Figure 2, we also propose an edge computing-based system design (below) to ensure low computational latency for DNN-based context-awareness.

2 SYSTEM DESIGN

The system architecture of Catch My Eye is shown in Figure 2. We implement an edge computing architecture to obtain online activity classification results. This architecture enables us to offload the computationally expensive DNN inference task from resource-constrained AR devices, while ensuring low latency from data transmission time and avoiding the wider dissemination of sensitive user gaze data. Communication between the AR device and the edge server is over a one-hop wireless local area network connection. We now detail system functionality through an overview of each of the modules contained in Figure 2.

Motion detection: This module takes the user head pose (position and orientation) in each frame of the AR app as input, which is available as the headset camera pose in standard APIs, and outputs a

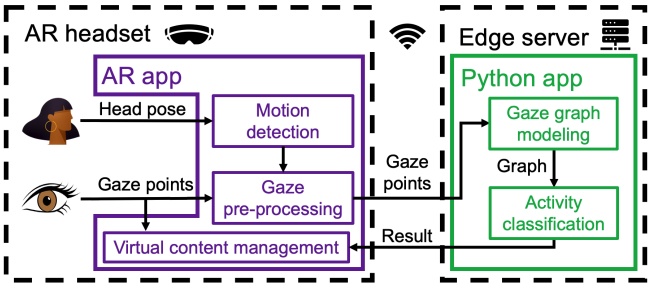


Figure 2: The system architecture of Catch My Eye. Gaze points available through eye tracking are sent from the AR headset to the edge server. The classification result is sent back to the headset to display appropriate virtual content.

binary indicator of high user motion (e.g., when the user is walking or looking around). The time series head pose data is analyzed using a window size of 0.5s; if the Euclidean distance between the head positions at the start and end of the window is greater than 0.25m, or the angle between the start and end head rotations is greater than 30°, a high level of user motion is detected. This enables us to filter out time periods during which DNN-based activity classification is not required: a high level of user motion indicates that a user is unlikely to be engaging with one of our static art exhibits, and we can avoid making unnecessary classification requests at these times by disabling the gaze pre-processing module.

Gaze pre-processing: This module is only enabled when low user motion is detected, in which case we consider the user may be engaging with an exhibit. Here we collect and process the 3D gaze points provided by the on-board eye tracker. We remove raw gaze samples that are corrupted (e.g., the eye tracker fails to estimate gaze when the user is blinking or the user's eyes are closing). The Magic Leap One leverages the confidence level to assess its confidence on the correctness of the gaze estimation results. In our current design, we filter out gaze samples with a confidence level lower than 0.6. We use a sensing window size of 10s, convert the collected gaze samples within the sensing window to JSON format, and send them to the edge server via an HTTP PUT request.

Gaze graph modeling: After receiving the gaze data on the edge server, we perform data normalization and outlier filtering on the gaze points. We then model the processed data as a spatial-temporal graph [5] to ensure better system robustness against the heterogeneity of gazes among different users.

Activity classification: Our classification model (for details see Section 5.2 of our accompanying work, EyeSyn [6]), takes a 128×128 gaze heatmap as input (the graph representation of the gaze points), and outputs one of three common activities performed by users while engaging with our virtual exhibits: *Reading text*, *Viewing painting*, and *Watching video*. If the classification confidence score is less than 0.5, we consider it a 'null' result.

Virtual content management: The gaze vector calculated from a gaze point, and where it first intersects with the spatial mesh of either the real world or a virtual object, indicates when a user is looking towards an exhibit without it being occluded. We then use the activity classification result to confirm whether a user is engaging with that exhibit (i.e., their activity matches the type of virtual content). While a user remains engaged with an exhibit, the relevant additional virtual content is activated.

3 INTERACTIVE DEMONSTRATION

We use the same architecture shown in Figure 2 for our demonstration. It is performed on a Magic Leap One running Lumin SDK 0.26, and an AR app built with Unity 2019.4.4f1. We run the edge-based gaze graph modeling and activity recognition on a desktop computer with an Intel i7-9000 3GHz CPU and an Intel UHD Graphics 630 GPU. A video of the demonstration is available online.[1] In a live demo a laptop will be used as the edge server.

Prior to the demo, an administrator has placed persistent AR content related to different artists around the room; for each artist there is a painting, a piece of text, and a video. When the user starts the AR app, 'Gathering data' appears in green text to indicate the app is gathering a sufficient window of gaze data to serve as input to the gaze-based activity recognition classifier. Once this data is available, a result from the activity classification module is rapidly received (end-to-end system latency is less than 200ms). Because the result, *Viewing painting* (displayed in green text) matches the content type, a painting, specific virtual content is activated to accompany the landscape painting: a cloud and rain (shown in Figure 1a), plus music inspired by the region.

When the activity classification result indicates the user has stopped viewing the painting, the accompanying virtual content is deactivated. Green text now shows the result is *Reading text*, and because the user is looking towards a piece of text, the accompanying virtual content is displayed: buttons to the left and right of the text which link to related information (Figure 1b). When the user returns to view the painting, the activity change is once again detected; the content that accompanies the text is deactivated and the content accompanying the painting is reactivated. In a live demo, users will be able to examine additional exhibits.

ACKNOWLEDGMENTS

This work was supported in part by NSF grants CSR-1903136 and CNS-1908051, NSF CAREER Award IIS-2046072, and an IBM Faculty Award.

REFERENCES

[1] Jaewon Choi, HyeonJung Park, Jeongyeup Paek, Rajesh Krishna Balan, and Jeong-Gil Ko. 2019. LpGL: Low-power graphics library for mobile AR headsets. In *Proceedings of ACM Mobisys 2019.*
[2] Brendan David-John, Candace Peacock, Ting Zhang, T Scott Murdison, Hrvoje Benko, and Tanya R Jonker. 2021. Towards gaze-based prediction of the intent to interact in virtual reality. In *Proceedings of ACM ETRA 2021.*
[3] Matt Dunleavy, Chris Dede, and Rebecca Mitchell. 2009. Affordances and limitations of immersive participatory augmented reality simulations for teaching and learning. *Journal of Science Education and Technology* 18, 1 (2009), 7–22.
[4] Facebook. 2019. Augmenting abstraction: Facebook expands AR experiences with Tate. https://tech.fb.com/augmenting-abstraction-facebook-expands-ar-experiences-with-tate-britain/
[5] Guohao Lan, Bailey Heit, Tim Scargill, and Maria Gorlatova. 2020. GazeGraph: Graph-based few-shot cognitive context sensing from human visual behavior. In *Proceedings of ACM SenSys 2020.*
[6] Guohao Lan, Tim Scargill, and Maria Gorlatova. 2022. EyeSyn: Psychology-inspired Eye Movement Synthesis for Gaze-based Activity Recognition. In *Proceedings of ACM/IEEE IPSN 2022.*
[7] T-Systems. 2022. Impressionism meets augmented reality. https://www.t-systems-mms.com/en/references/kunsthalle-muenchen.html

[1] https://sites.duke.edu/timscargill/catchmyeye-demo/