Poster– PROMAR: Practical Reference Object-based Multi-user Augmented Reality*

Tengpeng Li, Nam Son Nguyen, Xiaoqian Zhang, Teng Wang and Bo Sheng

Department of Computer Science, University of Massachusetts Boston

{Tengpeng.Li001, Sonnam.Nguyen001, Xiaoqian.Zhang001, Teng.Wang002, Bo.Sheng}@umb.edu

ACM Reference Format:

Tengpeng Li, Nam Son Nguyen, Xiaoqian Zhang, Teng Wang and Bo Sheng. 2019. Poster– PROMAR: Practical Reference Object-based Multi-user Augmented Reality. In *The 17th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '19)*, June 17–21, 2019, Seoul, Republic of Korea. ACM, New York, NY, USA, 2 pages. <https://doi.org/10.1145/3307334.3328610>

1 INTRODUCTION

Mobile Augmented Reality (MAR) represents an emerging category of applications that bring users interactive experiences with the physical experiences environment. In such applications, users can place virtual objects in real-world space, and view them through a camera view. In this work, we develop a framework that supports multi-users interactions for MAR apps, where one user places a virtual object that can be recognized by other users.

Our system does not require special hardware or preprocessing. The virtual objects can be arbitrarily placed in the physical environment. Most importantly, our system tolerates moderate viewpoint difference between the two users. Our basic design is based on image feature analysis. We have developed novel algorithms for image feature extraction and comparison particular for MAR applications. We have prototyped our system and conducted extensive experiments. The results show that scene recognition accuracy is high even with the viewpoint changes, and the runtime image processing is fast suitable for real-time applications.

2 DESIGN OF PROMAR

In our system, there are two user roles, and we briefly describe their basic steps as follows.

VO owner: This role is the user who places VOs in the scene. The VO is placed in the center of the phone's camera view by default (Fig. 1a). We provide a seekbar for the user to adjust the size of the VO that indicates the distance between the phone and the VO (denoted by VO_{dist}). Once the VO owner confirms the position and size of the VO, it will be rendered via the AR platform (Google ARCore [2]). Then, PROMAR uses the following three steps to prepare a package file for the VO viewers. (1) Identify the reference objects (RO) with the help of object detection system such

*This work is supported by National Science Foundation grant CNS-1527336 and UMass Boston Healey Grant.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

MobiSys '19, June 17–21, 2019, Seoul, Republic of Korea

© 2019 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-6661-8/19/06.

<https://doi.org/10.1145/3307334.3328610>

Figure 1: UI of our prototype: VO owner places a VO that can be viewed by VO viewer

as Tensorflow [3] or YOLO [4]. (2) Extract image features for each RO with a novel algorithm. (3) Record phone's pose information that defines the camera view angle (orientation sensor readings) and distance towards the VO. The major components in the VO owner's process are illustrated in Fig. 2, where the solid lines and dashed lines indicate the control flow and data flow respectively.

All these pieces of information are stored in a package file that will be delivered to the VO viewers.

VO viewer: A VO viewer receives the package file from the VO owner. Through her or his phone's camera view, the viewer keeps capturing image frames and conducting the following steps for each frame. (1) Identify the objects in it in the same way as in the VO owner's process. After obtaining the recognized objects, the viewer compares their labels with the recorded ROs'. If there is no match, the VO viewer discards this frame. (2) Obtain phone's pose. If there exists matching ROs, the viewer will derive the phone's pose information including the view angles and the distance between the phone and recognized ROs. (3) Extract image features and conduct featuring matching. Based on the viewer phone's pose information, the viewer will select a set of features from the VO owner's package, and compare to the features extracted from the viewer's frame. (4) Restore the VO based on the ROs. If a matching RO is found, the viewer will derive the VO's position on its frame according to the relative position information in the package file, and render the VO via the AR platform. Fig. 3 illustrates the architecture for VO reviewers.

3 TECHNICAL CONTRIBUTIONS

Our system design is based on the traditional image feature analysis. We mainly develop two novel algorithms, one for feature point

Figure 2: Program architecture for VO owners

extraction (FPE) and the other one for feature point comparison (FPC).

Distortion-based FPE. Our algorithm aims to select a set of *robust* FPs. Our main idea is to classify the possible viewpoint changes into multiple types of *distortion*. For each type, the VO owner prepares a set of FPs that are particularly robust to this type of distortion. Specifically, we consider four relevant properties of the viewpoint when defining distortion types: (1) horizontal angle, (2) vertical angle, (3) rotation, and (4) scale (the distance to the scene).

Pose-assisted FPC. For the VO viewers, we conduct the feature comparison in the following two steps. (1) *FP selection*: Based on the pose information of the viewer, we first assign a weight value to each involved distortion type. Then we select FPs so that the number of FPs chosen from each distortion type is proportional to its weight value (2) *FP matching*: The VO viewer also extracts N FPs from the matching object in its own frame. We develop a new matching algorithm with two rounds, strict matching and loose matching. In the first round, we enforce strong constraints and aim to find a small subset of matches that are very likely to be true matches. And then, we use these matches to derive a coordinate mapping between the two object images. In loose matching, we relax the constraints to include more features if they fit in the derived mapping.

4 PERFORMANCE EVALUATION

We evaluate our system with ALOI [1]. The experiments show our extraction algorithm yields 93% recall rate when the true negative rate is 100% with a single type of distortion. In comparison, regular extraction algorithm only maintains a recall rate of 64%.

In addition, we examine the performance with two types of distortion (horizontal angle and scale) and 10 objects in ALOI. The results are illustrated in Fig. 4, where horizontal angle change lies on the x -axis, scale change on the y -axis, and z values as well as the heat map indicate the matching ratios. For the simplicity of the plot, we do not include the matching ratio of false images, of which the average is 0.17. It would be straightforward to distinguish true images from false images with moderate viewpoint changes.

We also evaluate the system on an Android phone. Specifically, we place virtual objects with different ROs and play the role of a VO viewer moving the camera around for 10 minutes. The data recorded by the logger is illustrated in Fig. 5. The axis z shows the matching ratio, while axis x indicates the horizontal angle change, axis y indicates the vertical angle change, and the color represents the scale change (illustrated on the heat map). The average of matching ratio in our test is 0.28 which is consistent with our previous test with image data sets. These results confirm that PROMAR can

Figure 3: Program architecture for VO viewers

Figure 4: The matching ratio with two types of distortions (scale and horizontal angle).

Figure 5: The real-time matching ratio on mobile.

achieve remarkable performance in real-world tests even when the distortion is significant, e.g., shrinking size to about half or changing view angle for more than 45°.

REFERENCES

- [1] Jan-Mark Geusebroek, Gertjan J Burghouts, and Arnold WM Smeulders. 2005. *The Amsterdam library of object images*.
- [2] Google. 2018. Google ARCore. <https://developers.google.com/ar/>. (2018).
- [3] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy. 2016. *Speed/accuracy trade-offs for modern convolutional object detectors*. arXiv:1611.10012 <http://arxiv.org/abs/1611.10012>
- [4] Joseph Redmon and Ali Farhadi. 2018. *YOLOv3: An Incremental Improvement*.