Gaze3D: Framework for Gaze Analysis on 3D Reconstructed Scenes

Thomas Booth†
Rochester Institute of Technology

Srinivas Sridharan‡
Rochester Institute of Technology

Vasudev Bethamcherla§
Rochester Institute of Technology

Reynold Bailey¶
Rochester Institute of Technology

Abstract

An ongoing challenge with head-mounted eye-trackers is how to analyze the data from multiple individuals looking at the same scene. Our work focuses on static scenes. Previous approaches involve capturing a high resolution panorama of the scene and then mapping the fixations from all viewers onto this panorama. However such approaches are limited as they typically restrict all viewers to observe the scene from the same stationary vantage point. We present a system which incorporates user-perspective gaze data with a 3D reconstruction of the scene. The system enables the visualization of gaze data from multiple viewers on a single 3D model of the scene instead of multiple 2D panoramas. The subjects are free to move about the scene as they see fit which leads to more natural task performance. Furthermore since it is not necessary to warp the scene camera video into a flat panorama, our system preserves the relative positions of the objects in the scene during the visualization process. This gives better insight into the viewer’s problem solving and search task strategies. Our system has high applicability in complex static environments such as crime scenes and marketing studies in retail stores.

CR Categories: H.5.1 [Information Interfaces and Presentation]; Multimedia Information Systems—Artificial, augmented, and virtual realities I.4.7 [Image Processing and Computer Vision]; Feature Measurements—Feature representation;

Keywords: eye-tracking, scene reconstruction, virtual reality, image feature descriptors

1 Introduction

Eye-tracking provides a mechanism for monitoring where our high acuity vision gets focused in a scene or on a display. Today, most commercial eye-tracking companies offer head-mounted eye-tracking solutions which allow researchers to capture information about the visual behavior and perceptual strategies of people who were engaged in tasks outside of the laboratory. Modern head-mounted eye-trackers are usually equipped with a front-facing camera to capture the scene that the viewer is looking at as well as rear-facing cameras to capture eye-movements. They have already been used in a wide range of settings including driving [Sodhi et al. 2002], sports [Chajka et al. 2006], geology [Evans et al. 2012], and mental health monitoring [Vidal et al. 2011]. The use of head mounted eye-trackers in virtual and augmented reality research and applications is also gaining traction [Yeo et al. 2012; Diaz et al. 2013; Booth et al. 2013].

An ongoing challenge with head-mounted eye-trackers is how to analyze the data from multiple individuals looking at the same scene. The visualization techniques that have been developed for desktop-based eye-tracking systems (bee swarm, heat maps, scan path, focus maps etc.) are not directly applicable with head-mounted eye-trackers. In desktop systems, the eye tracker is stationary relative to the display on which the stimuli are presented making it straightforward to visualize the gaze behavior of multiple individuals on the same stimuli. However with head mounted eye-trackers the viewer can move freely about the 3D scene hence both the perspective and stimuli may not be the same across subjects.

We propose a framework for visualizing the gaze of multiple subjects on a 3D reconstructed scene. Our framework uses data from head mounted eye-trackers (scene camera videos and gaze points) to infer gaze position on the 3D reconstruction (see Figure 1). This approach solves the problem of maintaining the 3D scene context and grants researchers increased insight into subjects’ problem solving strategies.

The remainder of this paper is organized as follows: background and related work is presented in section 2, the system design is presented in section 3, analysis and discussion of the results are presented in section 4, and the paper concludes in section 5 with a summary of the contributions and potential avenues of future research.

2 Background

There is growing interest in human performance and interaction in real-world and 3D virtual environments. To better understand
gaze behavior in 3D space, researchers have previously used stereoscopic displays [Ki and Kwon 2008; Taherkhani and Kia 2012; Duchowski et al. 2014]. However such technologies are often associated with headaches, eyestrain, dizziness, or nausea [Lambooij et al. 2009; Wang et al. 2012; Kytö et al. 2013] and are prone to vergence and accommodation conflicts.

Other approaches utilize custom eye tracking setups (often combined with motion capture) to gather gaze information and explore the cognitive processes employed by humans in 3D environments [Shimizu and Fujiyoshi 2011; Essig et al. 2012; Diaz et al. 2013; Yeo et al. 2012]. Such systems are expensive and require careful placement of markers and calibration of multiple cameras - making them cumbersome and impractical for outdoor applications.

With all of these systems, analyzing gaze from multiple subjects presents an additional challenge. Pelz et al. [2011] addressed this challenge by creating a panorama using key frames from eye-tracker scene camera videos. The gaze points of all participants are then mapped onto this panorama as shown in Figure 3. However such approaches restrict the viewers to observe the scene from roughly the same vantage point and do not preserve the relative positions of objects in the scene. A similar approach was used by Booth et al. [2013] to infer, in real time, the fixations of a subject on a high resolution image of the scene. However this approach still relies on visualizing the gaze position in 2D. Our framework preserves the 3D context by combining a 3D reconstruction of the scene with scene camera frames to infer the gaze of the subject.

### 3 System Design

Figure 2 shows the system architecture and data flow for our framework. The data flow can be divided into four stages: data collection, data preprocessing, gaze inference, and visualization.

#### 3.1 Data Collection

A 3D scan of the scene is first obtained using a consumer depth camera (we use a Microsoft Kinect for Xbox 360 sensor). The Kinect is physically moved about the scene to ensure exhaustive coverage. Better quality scans can be obtained using the newer Kinect for xBox One or professional grade systems like Faro laser scanners. The Kinect Fusion software (included in Kinect for Windows SDK) creates a 3D reconstruction of the scene from the depth
sensor while retaining color information provided by the RGB camera. The output of this process is a PLY mesh.

Subjects view the previously scanned scene while wearing head mounted eye-trackers. The subjects are free to move about the scene as they see fit. The data captured by the eye trackers include scene camera videos and accompanying gaze data. We use SMI Eye Tracking Glasses, however any head mounted eye-tracker can be used with our framework.

3.2 Data Preprocessing

The PLY mesh generated by the Kinect Fusion software contains artifacts. To obtain a cleaner texture and scene geometry for subsequent stages of the pipeline, these artifacts are removed using MeshLab [Cignoni et al. 2008]. A texture for the scene model is created using the per-vertex color information from the Kinect’s RGB camera. The texture is based on a 2D parameterization of the XY perspective of the mesh. While other perspectives are possible, the XY viewpoint provides the most comprehensive coverage of the scene. This is due to the fact that Kinect Fusion begins the construction in 3D space while facing the positive Z-axis.

The resolution of the extracted texture is significantly lower in quality than the video captured by the scene camera of the eye-tracking glasses. Therefore we refactor the texture from the perspective of each subject. Refactoring in this case means replacing regions on the low resolution texture with corresponding high resolution regions from the eye-tracker scene camera video. We select a frame from each subject’s scene camera video that has the closest perspective to the low resolution texture. A sequential search is then performed at different scales for similar colored regions and the best matching region is selected. A color matching approach is preferred to other image feature matching techniques due to the distortion caused by the flattening of the 3D scene.

3.3 Gaze Inference

SURF descriptors are extracted from each frame of a subject’s scene camera video by performing a radial search about the gaze point for that frame. To obtain only local features around the gaze point, we constrain our search to 5% of the total image area. Corresponding features are identified in the subject’s refactored texture map to infer the gaze position. To eliminate incorrectly matched features we use Cook’s Distance measure [Cook 1977] which performs dimensionless weighting of all values in a set, with respect to the mean of that set. The end result of this process is a sequence of gaze points on the refactored texture maps that correspond to the subject’s gaze on the scene camera video.

3.4 Visualization

During visualization, the inferred gaze point from each subject is applied directly to the low-resolution texture and mapped onto the mesh, as shown in Figure 1 and Figure 4. This can then be viewed in a desktop OpenGL context or within an HMD such as the Oculus Rift to provide a more immersive experience. We provide camera controls which enable observer interaction and fly-throughs.

4 Results

Figure 4 demonstrates the successful tracking of multiple subjects on a simple test scene. We use this scene as an exemplar for evaluating performance as described in the remainder of this section.

The original mesh generated by Kinect Fusion had a size of 157 MB and contained 3,378,678 vertices and 1,126,226 faces. MeshLab was used to remove duplicate vertices and faces, and other artifacts present in the scene. We define artifacts as independent clusters of vertices with fewer than 25 faces or components of the mesh that do not reside within the primary scene (such as the floor). The resulting mesh was 19 MB in size and contained 578,217 vertices and 276,014 faces.

For our runtime tests we utilized a system equipped with an AMD Opteron 1214, Nvidia GeForce 8600 GTS, and 4 GB of RAM. We analyzed performance on two areas of interest – preprocessing and playback:

- In the preprocessing phase, each frame from the eye-tracker scene camera undergoes SURF decomposition, descriptor matching with the refactored texture, and gaze inference on the refactored texture. For the scene depicted in Figure 4, it takes about one second to process each frame in this manner.

- In the playback phase, our test system was capable of rendering the 3D reconstruction along with corresponding gaze points from multiple subjects in real-time. The performance impact from introducing additional subjects was negligible in comparison to rendering time.

A small percentage of captured frames from the eye-tracker do not include a gaze point. This occurs because of track-loss (i.e. the eye-tracker is not able to detect the pupil due to blinks or extreme pupil position outside the scene camera’s field-of-view). In this case the scene video frames will have no accompanying gaze information making fixation inference impossible. When this occurs we simply discard the frames and resume once gaze positions are again available. Also, if the viewer happens to look away from the scene, then the inferred gaze position will naturally be incorrect. We detect when this is happening by keeping track of the number of matched descriptors from frame-to-frame. If the number changes drastically (indicating dissimilar scenes) then we ignore the results from the SURF descriptor matching algorithm. Across the three subjects, a gaze point was present in 96.8% percent of frames and was found to lie within the primary scene in 86.0% percent of frames.

5 Conclusion & Future Work

In summary, we present a system which incorporates user-perspective gaze data with a 3D reconstruction of a scene. Our framework enables the visualization of gaze data from multiple viewers on a single 3D model of the scene instead of multiple 2D
panoramas. This approach preserves the relative position of objects in the scene and aids in visualizing the scanpaths of multiple subjects simultaneously. This gives better insight into the viewers’ problem solving and search task strategies. Our system has high applicability in complex static environments such as crime scenes and marketing studies in retail stores.

While our framework solves the problem of inferring multiple perspectives on a 3D reconstruction, the system is not without limitations:

- The primary limitation is the manual effort involved in artifact removal. While the MeshLab tool provides many useful features, manual input is still required to obtain the best results.
- The size of the 3D reconstruction is constrained by the available GPU memory. Kinect Fusion lacks the ability to automatically identify and remove redundant geometric information. This means that the GPU memory is not efficiently utilized.
- During scanning, the scene must be visually stable (i.e., unchanging external lighting, no moving objects, and all objects must be non-transparent and well defined in terms of color, contrast, size, shape and texture).

The Kinect-based Scalable Real-Time Volumetric Surface Reconstruction framework recently developed by Chen et al. [2013] overcomes many of the limitations of Kinect Fusion. It provides automatic filtering of redundant information and generates better quality scans over larger environments. Adapting our system to utilize this framework will significantly reduce the manual effort required during the data processing stage.

In addition to improving the performance of our framework, there are many other potential avenues for future work. We are particularly interested in collaborative applications where remote experts can monitor in real-time the gaze behavior of on-site observers and provide guidance.

Acknowledgements

This material is based on work supported by the National Science Foundation under Award No. IIS-0952631. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References


