Eye Contact over Video

Jesper Kjeldskov
Computer Science Dept.
Socio+Interactive Design
Aalborg University, Denmark
jesper@cs.aau.dk

Mikael B. Skov
Computer Science Dept.
Socio+Interactive Design
Aalborg University, Denmark
dubois@cs.aau.dk

Jacob H. Smedegård
Computer Science Dept.
Socio+Interactive Design
Aalborg University, Denmark
jhaubach @cs.aau.dk

Jeni Paay
Computer Science Dept.
Socio+Interactive Design
Aalborg University, Denmark
jeni@cs.aau.dk

Thomas S. Nielsen
Computer Science Dept.
Socio+Interactive Design
Aalborg University, Denmark
primogens@gmail.com

CHI 2014, Apr 26 - May 01 2014, Toronto, ON, Canada
ACM 978-1-4503-2474-8/14/04.
http://dx.doi.org/10.1145/2559206.2581221

Abstract

Video communication systems traditionally offer limited or no experience of eye contact due to the offset between cameras and the screen. In response, we are experimenting with the use of multiple Kinect cameras for generating a 3D model of the user, and then rendering a virtual camera angle giving the user an experience of eye contact. In doing this, we use concepts from KinectFusion, such as a volumetric voxel data representation and GPU accelerated ray tracing for viewpoint rendering. This achieves a detailed 3D model from a noisy source, and delivers a promising video output in terms of visual quality, lag and frame rate, enabling the experience of eye contact and face gaze.

Author Keywords

Eye contact; gaze; virtual view camera; Kinect

ACM Classification Keywords

H.5.5. Group and Organization Interfaces: Computer-supported cooperative work, Synchronous interaction.

Introduction

Eye contact plays an important role in interpersonal face-to-face communication [3]. Contemporary video communication systems, however, offer very limited experience of eye contact with a remote person due to their physical setup, which involves an offset between the camera and the screen. According to [11] if this

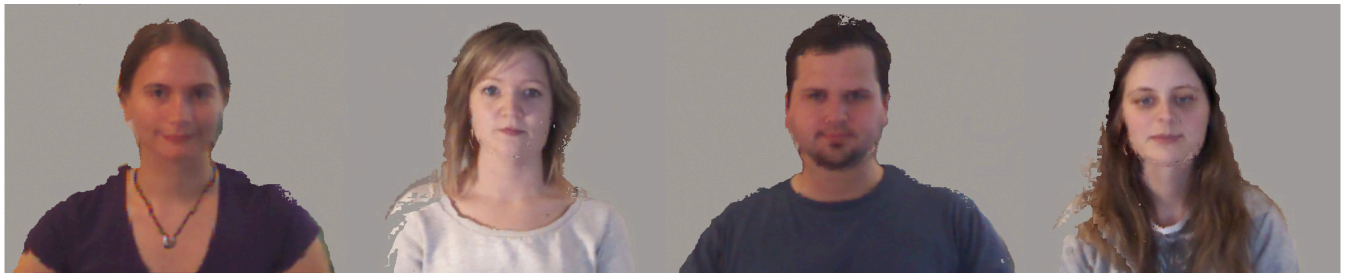


Figure 1. Virtual camera views rendered by EyeGaze for enabling eye contact over video.

offset is greater than 5 degrees, at conversational distance, then the experience of eye contact is lost. Instead, the remote person appears to be looking slightly away from their communication partner, depending on where the camera is placed. Placing a physical camera where the remote viewer's eyes are depicted on the screen is obviously challenging, if not impossible, as it would either occlude, or be occluded by, the screen. As an alternative approach we have explored the feasibility of creating a virtual camera view from the correct perspective. To achieve this we have used a number of Microsofts Kinect cameras placed along the edges of the screen to generate a real time 3D model, and then rendered a video stream from the desired camera perspective. Here we introduce our prototype system called EyeGaze, and illustrate the quality of the rendering output.

Related Work

Face gaze and eye contact has been subject of research in social psychology since the mid 1960s. Its significance in human relationships and communication has been summarized in, for example [3]. Firstly, people's gazing behaviours have been shown to influence other people's judgements of liking/attraction, attentiveness, competence social skills, credibility, and dominance, and also as an expression of the intensity of their feelings, and of their intimacy. Secondly, gazing behaviour has been shown to provide an important regulatory function in communication, such as turn taking and synchronization between verbal and kinesic behaviours. As a third factor, gaze and eye contact has also been shown to have an important social control function in relation to acts like persuasion, deception, ingratiation, threat, dominance, avoidance, and compliance. Since these communicative roles of gaze and eye contact are largely lost when communicating over video, several strands of research has explored technologies that facilitate better mediation of them.

One of the earliest examples is the Hydra system [10] that preserves participants' spatial arrangement in a distributed multiparty meeting. More recent examples include "blended spaces" like HP Halo and BIS*i* where physical placement of displays, cameras and furniture creates an experience of two or more locations blending into one [7, 9]. As alternative to well-considered spatial arrangement and physical configurations, and the use of raw camera feeds, other research has explored the possibility of rendering live virtual viewpoints from multiple cameras. An early example of this is [8] where a virtual camera view is calculated based on stereoscopic analysis. More recently this approach has been extended with the use of depth sensors for better

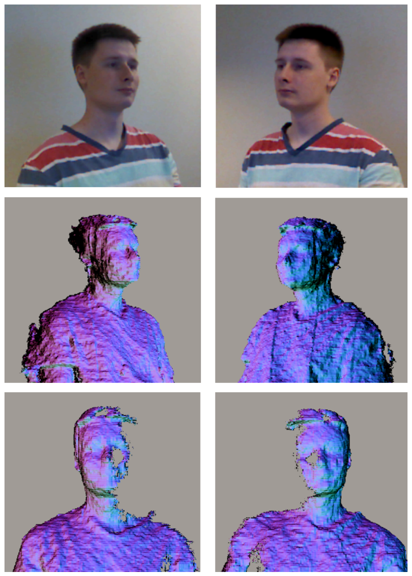


Figure 2. Output from the RGB camera with corresponding 3D models generated from the depth camera. Note that the models are incomplete, but complementary and overlapping.

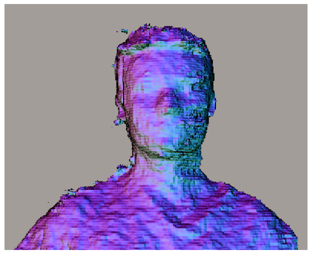


Figure 3. The merged 3D model

results, such as in [12]. Very visually promising results are also presented in [4] where a Kinect is used to rotate a facial texture in a video frame making it appear as if the person is looking straight at the camera. Notably [5] demonstrates the capturing of an entire environment into a 3D model, and subsequent rendering of virtual viewpoints in real time.

EyeGaze

Our system, EyeGaze, allows eye contact and face gaze between two people over video by creating visually realistic representations of the users from virtual camera views and rendering it in real time (Figure 1). The graphics pipeline of EyeGaze has two parts: 1) a merging algorithm storing data from cameras in a voxel grid, and 2) an engine for ray tracing the scene from a remote user's perspective. Building on the work in [5], we have focussed our technical efforts on using a volumetric data representation. Specifically, we use a voxel data representation, which allows us to merge data from several Microsoft Kinect depth cameras, and improve the quality and completeness of the 3D model over time. In order to achieve high frame rates, we have constructed a GPU accelerated ray tracer that renders the model stored in the voxel grid and texturizes it using the RGB data captured by the Kinect.

Volumetric Representation

Using off-the-shelf hardware such as the Kinect involves some challenges with accuracy of depth data. To overcome this, we apply a Signed Distance Function (SDF) to encode geometry by describing the distance to the surface of the model, inspired by [1, 2]. The SDF is represented as a fixed resolution three-dimensional grid of volumetric pixels (voxels), where the position of a voxel is implicit in the grid. Each voxel holds a discrete SDF value, describing its proximity to the surface of the model. The actual surface is thus an implicit surface that can be calculated when needed by interpolating the SDF from several voxels adjacent to it. New distances can be merged into the SDF for each frame and from each Kinect camera used, allowing continuous refinement of the 3D model from multiple angles.

Merging camera inputs

The merging algorithm captures depth and colour frames from all Kinects, and then for each Kinect transform the voxels into camera space and project them into 2D positions on the Kinect camera's depth frame. It then calculates the distance between the voxel and the surface observed by the Kinect camera, and finally average the surface distances obtained from the depth frames with the weighted distance obtained from the voxel's truncated SDF value, and store it in the grid. Figure 2 illustrates the data obtained from two Kinect cameras placed either side of a person. The top row shows the output from the RGB camera. The middle row shows the corresponding 3D models generated from the depth camera. In the bottom row we have rotated the 3D models to view the face straight on. This illustrates that they are both incomplete, but also that they are complementary and partly overlapping, with each missing data that is contained in the other. Figure 3 shows the merged 3D model created from the two depth cameras.

Ray tracing

In order to obtain live video output we have constructed a GPU accelerated ray tracer that renders the 3D model stored in the voxel grid. Because natural light is a part of the texture, we do not need virtual lighting as in a traditional ray tracer. One of the

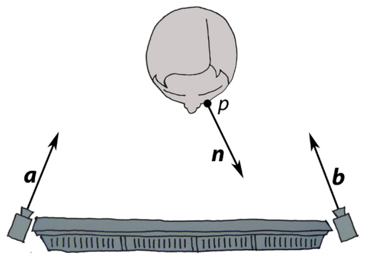
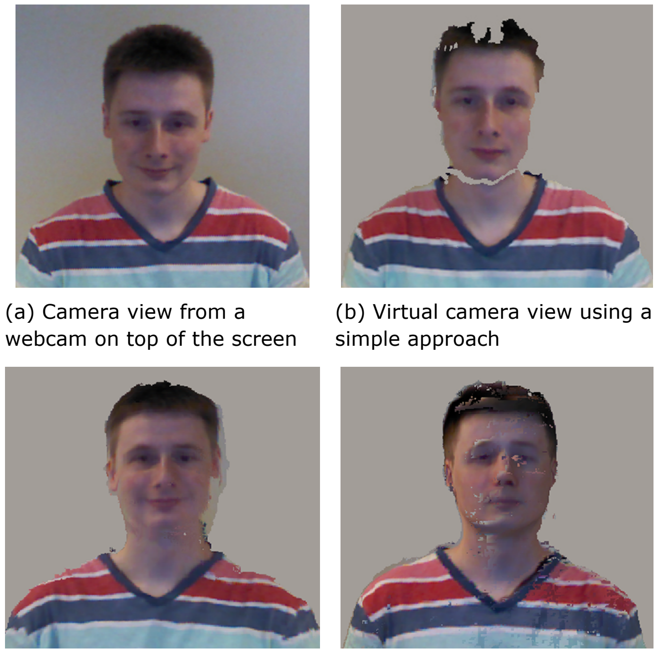


Figure 4. Using vectors to decide what camera provides best data for texturing.

challenges during ray tracing is to decide which Kinect RGB camera should provide the texturing colour for this point. Each point in the scene may be captured by more than one Kinect, and thus we need to identify what Kinect has the best view of it. We base this decision on the angle between the surface normal of the intersection point and the vectors of the Kinect cameras, as illustrated in figure 4. Knowing the surface normal (***n***) of the point *p*, and the vector of each Kinect (***a*** and ***b***) we can identify what Kinect has the smallest angle of disparity, and hence the best view of *p*. In this example, the angle between ***n*** and ***b*** is smaller than the angle between ***n*** and ***a***, telling us that the camera on the right side of the figure has the best data for texturing. The position of *p* is then transformed into the camera space for that Kinect, and perspective-projected, to acquire the pixel coordinates (*x, y*) needed to lookup data in the RGB image captured by it.

Quality and performance

Figure 5 show the output from a traditional webcam placed on top of the screen (a), video output from a simple virtual camera approach (b), and rendered video from EyeGaze (c and d). The figure illustrates that EyeGaze is capable of producing a high quality virtual camera view of the user's head and torso in good detail and enabling the experience of eye contact and face gaze. The difference between using one or two Kinect cameras for texturing is that two cameras provide more detail in "shaded" areas such as under the chin and in the eye sockets (figure 5d), while using only one camera results in those areas being blurred (figure 5c). As can be seen in figure 5d, our current texturing from two cameras does, however, create a bit of colouring-noise, which requires some correctional filtering.



(a) Camera view from a webcam on top of the screen

(b) Virtual camera view using a simple approach

(c) EyeGaze - with texture from one Kinect

(d) EyeGaze - with texture from two Kinects

Figure 5. Video output from webcam on top of the screen (a), a simple virtual camera approach (b), and EyeGaze (c and d)

Using depth-sensing cameras, volumetric representation within a Signed Distance Function, and exploring the powers of the GPU, EyeGaze is able to capture, model and render with a time lag of less than 40 ms and a frame rate of up to 25 fps. This means that the system is experienced as working in real time.

Using multiple Kinect cameras, EyeView can reliably capture enough perspective of the user's face and torso to render a realistic face-on view with very little missing data in each video frame. Due to the way volumetric data is represented EyeGaze is also able to "remember" objects and areas when they are temporarily occluded, which minimizes the number of video frames with missing data.

Model quality

Using a volumetric approach for storing information about the users and the background allows us to achieve a high quality image from a noisy input source. This is illustrated in the difference between figure 5b and 5c/d. Figure 5b shows a frame of texturized depth data from a single Kinect, where the model has been tilted to obtain a straight-on view. This results in a lot of missing information in the model, and therefore holes in the rendered video. In figure 5c/d we see a notable increase in model quality, with far less missing data. This is firstly because of the better coverage obtained with multiple cameras, but also because the volumetric approach allows us to compensate for the lack of data in one frame by reusing information from a previous one. Thereby, random misreading by the depth cameras have only very little effect on the model, and on the rendered video image.

Frame rate

For our current prototype we store the model in a 512^3 voxel grid and render at 1080p on an NVIDIA GeForce GTX 770 graphics card. The merging and ray tracing implementations are alternating in execution, limiting the output video to 30 fps. This will result in some frames being dropped due to timing constraints. While our frame rate does not match the performance in [5] it is still promising given the computation required for merging data and rendering video at this resolution.

Conclusions and further work

We have presented a gaze enabling video system that takes live depth data from two Microsoft Kinect cameras and renders a virtual camera view in real time. We have shown that using two Kinects, volumetric voxel data representations, concepts from KinectFusion [2], and exploring the power of GPUs for ray tracing, EyeGaze can create a visually convincing representation of a remote person from a virtual camera perspective enabling the experience of eye contact. EyeGaze renders this representation of the remote person in real time with low lag and promising frame rates.

Our work with EyeGaze opens several avenues for further work. Firstly, we are exploring the effect of rendering the virtual camera view from a dynamic set of coordinates matching the exact location of the viewer's eyes, creating an even more natural experience, as the visual perspective changes when you move your head, as it does face-to-face. We currently do this using the Kinects' built-in skeletal tracking and feeding this data into the ray-tracer. The effect is indeed promising. However, some texture errors do occur when real time RGB data for a particular perspective is incomplete. We speculate that this may be solved by storing known texture maps, and then simply updating these in sync with the model.

Related to this we are also exploring the rendering of a stereoscopic image, and displaying this to the remote user on a 3D screen. This is in principle a built-in potential of our approach, as one just needs to define *two* virtual camera views – for the left and right eye. However, there are some technical challenges of calibration and synchronization, as well as performance when rendering two streams of video rather than just one. Another potential of our approach is to render multiple perspectives for multiple remote viewers. This could be used to facilitate eye contact in meetings with several individual participants in a spatial arrangement similar to that of the Hydra system. If combined with multi-perspective capable screens [6] this could also be used to facilitate eye contact in meetings between *groups* of people, in a spatial arrangement similar to HP Halo and BIS*i*.

We have compared the experience of EyeGaze to face-to-face communication and the use of Skype through two within-subject laboratory experiments with a total of 130 participants. Findings from these experiments are still being analyzed in detail, and will be published elsewhere, but preliminary results indicate that while face-to-face is still superior, EyeGaze has added value over the traditional video system in terms of eye contact, involvement, turn-taking and co-presence.

References

[1] Curless, B. and Levoy, M. A volumetric method for building complex models from range images. In *Proc. SIGGRAPH 96,* ACM (1996), 303-312.

[2] Izadi, S. Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S. Freeman, D., Davison, A. and Fitzgibbon, A. KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. In *Proc. UIST 2011,* ACM (2011), 559-568.

[3] Kleinke, C.L. Gaze and Eye Contact: A Research Review. *Psychological Bulletin 100,* 1 (1986), 78-100.

[4] Kuster, C., Popa, T., Bazin, J.C., Gotsman, C., Gross, M. Gaze Correction for Home Video Conferencing. In *Proc. ACM SIGGRAPH Asia* (2012).

[5] Maimone, A. and Fuchs, H. Encumbrance-free telepresence system with real-time 3D capture and display using commodity depth cameras. In *Proc. ISMAR 2011,* ACM (2011), 237-146.

[6] Nguyen, D. and Canny, J. MultiView: spatially faithful group video conferencing. In *Proc. CHI 2005,* ACM (2005), 799-808.

[7] O'Hara, K., Kjeldskov, J. and Paay, J. Blended interaction spaces for distributed team collaboration. *Transactions on Computer-Human Interaction 18,* 1 (2011), Article No. 3.

[8] Ott, M., Lewis, J. P. and Cox, I. Teleconferencing eye contract using a virtual camera. In *Proc. INTERACT'93 and CHI'93,* ACM (1993), 109-110.

[9] Paay, J., Kjeldskov, J. and O'Hara, K. BISi: a blended interaction space. *Ext. Abstracts CHI 2011,* ACM (2011), 185-200.

[10] Sellen, A.J., Buxton, W., and Arnott, J. 1992. Using spatial cues to improve videoconferencing. *In Proc. CHI 1992,* ACM (1992), 651-652.

[11] Stokes, R. Human Factors and Appearance Design Considerations of the Mod II PICTUREPHONE & Station Set. *IEEE Transactions on Communication Technology 17,* 2 (1969), 318-323.

[12] Zhu, J., Yang, R. and Xiang, X. Eye contact in video conference via fusion of time-of-flight depth sensor and stereo. *3D Research 2,* 3 (2011), 1-10.