View Splicing for Effective VR Collaboration

Lili Wang[1;2] * Wentao Wu[1;2] Zijing Zhou[1] Voicu Popescu[3]

[1]State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, China
[2]Peng Cheng Laboratory, Shengzhen, China
[3]Purdue University, West Lafayette, Indiana, U.S.

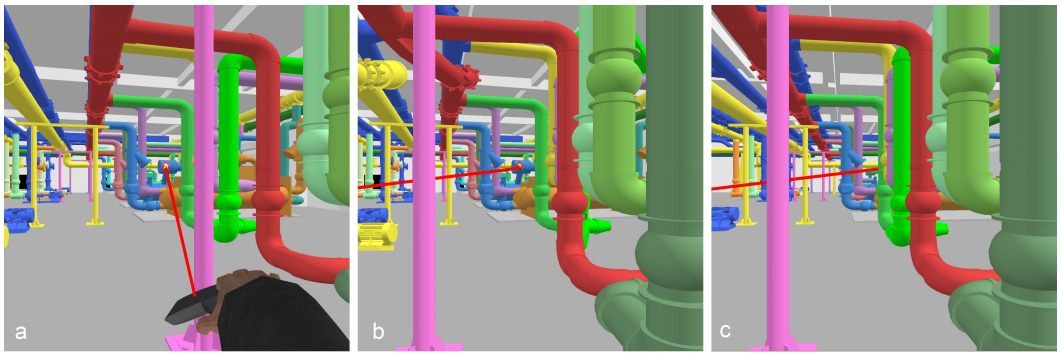

Figure 1: Switch camera view splicing. A collaborator indicates to the user a part of the virtual environment with a virtual laser pointer (a). The laser dot is not visible to the user due to occlusions (c), which hinders collaboration. We propose to show the user a visualization computed with a novel multiperspective camera called the *switch camera* (b), which splices the user and collaborator views seamlessly. The geometry near the user is shown from the user's perspective, see near part of b and c, and the geometry away from the user is shown from the collaborator's perspective, revealing the laser dot, see far part of b and a.

Abstract

In a co-located multi-user collaborative virtual reality (VR) application a collaborator should be able to indicate a workspace location to the user, such that they can refer to it simultaneously as they work together. Due to their different viewpoints, the collaborator sees some parts of the virtual environment (VE) that the user does not, and communication breaks down when the collaborator's reference is not visible to the user. The conventional solutions of asking the user to move around to gain line of sight to the collaborator's reference, or of asking the user to toggle back and forth between their view and that of the collaborator can be inefficient and ineffective. This paper proposes a method for improving collaboration in VR by alleviating the disparity between the user and the collaborator views of the VE. The user is shown a multiperspective visualization of the VE that transitions smoothly from the user to the collaborator's perspective. The multiperpsective visualization is based on the *switch camera*, a novel camera model with curved rays that splice together the user and collaborator views. The multiperspective visualization is computed first by warping the VE geometry, through projection with the switch camera followed by unprojection with a conventional camera, and then by rendering the warped VE conventionally, for each user eye. A controlled user study with three tasks shows that VR collaboration using the switch camera multiperpsective visualization is faster, more reliable, and less taxing on the user than the conventional approaches of viewpoint translation or view toggling.

Index Terms: Virtual reality—Collaborative—Occlusion management—Multiperspective visualization;

*e-mail: wanglily@buaa.edu.cn

1 Introduction

Consider a collaborative virtual reality (VR) application where a user works side by side with a collaborator. This is a typical, "shoulder to shoulder", collaboration scenario, where the user and the collaborator explore the virtual environment (VE) from the "inside looking out", as is the case, for example, when a new worker is trained on operating a power plant, or when a guide gives a virtual tour of a historical site.

The collaborator has to be able to indicate a workspace location to the user, such that they can both refer to it as they work together. Due to their different viewpoints, the collaborator sees some parts of the virtual environment that the user cannot see due to occlusions, and vice versa, and communication breaks down. If the collaborator uses a virtual laser pointer, the laser dot might be hidden from the user. A possible solution is for the user to move around to gain line of sight to the laser dot, but this takes time and effort, making collaboration inefficient. Another possible solution is for the user to toggle between their view and that of the collaborator. However, this has the disadvantage that the user cannot see the collaborator's avatar when using the collaborator's view, as needed, for example, to see a gesture made by the collaborator. This complicates communication as it requires the user to toggle back and forth between the two views exactly at the right time.

In this paper we propose to improve collaboration in VR by alleviating the disparity between the user and the collaborator's views of the VE. The user is shown a multiperspective visualization of the VE that transitions gradually from the user to the collaborator's perspective (Figure 1). The multiperspective visualization partitions the VE into a *user region*, which corresponds to the part of the VE close to the user and includes the collaborator avatar, a *collaborator region*, which is the part of the VE on which the user and the collaborator work together, and a *transition region*, which connects the user and collaborator regions. The multiperspective visualization shows the

978-1-7281-8508-8/20/$31.00 ©2020 IEEE
DOI 10.1109/ISMAR50242.2020.00079

user region from the user's perspective (Figure 1), it switches from the user to the collaborator perspective over the transition region, and it shows the collaborator region from the collaborator's perspective. This way, no matter where the collaborator places the virtual laser dot in the collaborator region, if the collaborator can see it, the user will see it as well. Furthermore, the user can look at the collaborator with a simple head turn, as needed, for example, to see a gesture made by the collaborator. The view splicing is implemented with the *switch camera*, a novel camera model with curved rays that merges the user and collaborator views.

We measured the effectiveness of our view splicing approach in a user study with two control conditions and three tasks. The results show that using our approach, the user (1) grasps the spatial reference made by the collaborator more quickly, (2) follows the collaborator's alternating spatial references and gesture more reliably, (3) traces visually the VE more reliably, and (4) reports a lower task load than for the control conditions.

2 Prior Work

In recent years, a variety of cooperative observation methods have been proposed, and they have been shown to improve cooperation efficiency between collaborators.

Some methods focus on the collaborative exploration of scientific data. For example, one immersive visualization system allows multiple users to visualize and interact with multidimensional data cooperatively [6]. Another research effort supports artist-scientist-technologist collaboration through multivariate VR visualizations and sketching [23]. Neither of these methods attempt to mitigate the influence of occlusions, or to facilitate attention sharing amongst collaborators. Numerous prior art methods focus, as does our method, on the collaborative exploration and use of a 3D virtual environment in which the user is immersed. We briefly review prior VR collaboration methods for sharing attention cues (Section 2.1), for view sharing (Section 2.2), for using multiple views in parallel (Section 2.3), and for object selection (Section 2.4).

We do not review prior work on making VR collaboration tractable in the context of projection-based systems (e.g., CAVEs), where the challenge is to modify the shared displayed image quickly enough to provide each user with their own perspective [1, 18, 25]; we provide each user their own perspective with VR head-mounted displays (HMDs). We also do not review prior work on VR group navigation [25], which is beyond the scope of our paper, as we focus on collaboration from positions typically assumed by users as they work together, without considering the navigation mechanisms needed for the collaborators to move from one position to the next.

2.1 Shared attention cues

Helping a user to know what a collaborator is looking at has been approached from many angles. Some systems convey to the user the gaze of a remote collaborator with a dot on the user's view of the VE and the hand gesture of the remote collaborator with a picture in picture effect [17]. Our work can be used in conjunction with these earlier systems to avoid that the dot indicating the collaborator's gaze become occluded. Our work addresses the scenario where the user and the collaborator are collocated in the same physical space, and our method allows the user to see the collaborator when desired with a natural head rotation. Prior work also relied on user and collaborator placed landmarks to improve the collaborative referencing of the VE [31,32]. A landmark visible to the collaborator can be hidden to the user, and our work could be used to ensure that the user and the collaborator see the exact same set of landmarks.

Different combinations of awareness cues, such as the visualization of the collaborator's field-of-view frustum, eye-gaze ray, and head-gaze ray were compared to a control condition where only collaborator's head and hands were shown [35]. The results showed that awareness cues significantly improved user performance in a collaborative object finding and placing task. Usability and subjective preferences were also higher when attention cues were used, with the combination of field of view and head-gaze ray visualization scoring the highest. Our method can be readily used to improve head-gaze ray visualization further, by making sure that the intersection between the ray and the VE is visible to the user.

Another approach is to eliminate the distance between the collaborator and the VE focus point, which requires that the collaborator navigate within touching distance of each focus point [7]. The approach alleviates the referencing ambiguity at the cost of a less efficient interface that slows down collaboration. We also refer the reader to a comprehensive survey of methods for communication and awareness in collaborative virtual environments [33].

2.2 View sharing

An approach popular in remote collaboration is to let the user share their view of their workspace with the remote collaborator. Based on this visualization, the remote collaborator provides instructions to the user that aid them in completing the task [17]. A recent study compares three view-sharing techniques for one-to-many mixed reality collaboration [28]. The techniques share the view through 2D video, 360^o video, and through a 3D model augmented with 2D video. The results show that the 3D model augmented with 2D video was time-efficient, usable, less demanding and most preferred compared to the other techniques.

One of the challenges of view sharing is that the user's first person view is an inadequate workspace visualization for the remote collaborator, as it changes abruptly and substantially with the user's frequent head motions. Prior work has attempted to stabilize this visualization [29] by projection on a planar workspace proxy. In our work we do compare our method to a control condition where the user can toggle back and forth between their own and the collaborator's view. We avoid the visualization instability with a looser coupling of the two views: when the user switches to the collaborator view, the user adopts the collaborator's viewpoint, but the view direction remains under the interactive control of the user.

An ingenious system displays what a user of a VR app sees using a phone taped to the user's headset, with the screen facing towards the audience [36]. The system enables the collaboration between a user of a VR application and a collaborator that is not immersed in the VE. Another special case of view sharing is to provide a visualization of a real world environment to a user through a collaborator who wears a head mounted panoramic camera, and who executes verbal commands from the user to acquire the real world environment from locations that interest the user [22].

2.3 Using multiple views

A natural approach for supporting collaboration in VR is to let the users simultaneously see multiple views of the VE. In one system, users can define allocentric and egocentric views of the VE that they can then share through virtual handheld displays that capture each view [26]. The system allows defining multiple views, suitable for a the comprehensive exploration of a complex VE. However, the views are not integrated, so the user has to switch context from each virtual "tablet" to the next. Our approach provides the user the benefit of only two views—that of the user and of the collaborator—but the two views are seamlessly integrated.

A collaborative VR system for interior design allows multiple users to navigate the building with an allocentric "dollhouse" view followed by conventional first-person exploration [19]. The user can point in first-person mode and the user's avatar replicates the pointing gesture in the dollhouse view catered to the other users. Due to the considerable change in perspective, the pointing target might not be visible in the dollhouse view.

Researchers have recognized and addressed the need to integrate the multiple views of the VE in a more seamless way. Indeed, visu-

alizing a VE in parallel through multiple images places a substantial cognitive burden on the user, who reverts to examining one image at a time, in serial fashion. Multiperspective visualization provides a way of integrating multiple images without redundancy and with good continuity, allowing the user to truly visualize the scene in parallel from multiple viewpoints, and across multiple scales [13, 30, 34].

Our work takes the multiperspective visualization approach to connect the user's view to the view of the collaborator. Our visualization is based on the switch camera, a novel camera model that builds upon the earlier graph camera [37]. The graph camera is literally a graph of conventional (planar pinhole) cameras. The graph camera is constructed from a root view frustum through recursive frustum bending, splitting, and merging operations, each of which add additional viewpoints. The resulting piecewise linear rays are routed around occluders to gather scene samples from multiple viewpoints.

The graph camera was later extended to overcome some of its limitations. The first limitation addressed was the local distortion occurring when perspective switches abruptly from one viewpoint to the next through view frustum bending; the curved ray camera [10] upgrades the C^0 continuity of the piecewise-linear rays of the original graph camera to C^1 continuity, transitioning from one viewpoint to the next over a greater distance, alleviating the local distortion artifact. However, the curved ray camera cannot serve our purpose of providing a view of the collaborator region that is identical to what the collaborator sees–this is a global constraint, which cannot satisfied by local ray modifications.

Graph camera multiperspective visualization has been used successfully in the context of making VR and AR exploration more efficient, by showing the user more that they can see from the current position [41, 43]. A comparative study tests VR multiperspective visualization against prior occlusion management techniques, such as transparency and top-view visualization [42]. The study reveals that multiperspective visualization has advantages over transparency when there is more than one occluding layer, which make the transparent visualization hard to understand; the study also reveals that multiperspective should be preferred over top view when the objects of interest cannot be easily recognized from above.

Our paper brings the benefit of multiperspective visualization to the context of VR collaboration, which requires an important extension of the graph camera to splice the user and collaborator views, which none of the prior graph cameras can achieve.

2.4 Object selection

Our method allows collaborators to refer to the same VE element as they work together. The collaborator points with a virtual laser pointer and our method makes sure that the user can see the laser dot, free of occlusions.

The challenge posed by occlusions to selection in VR has been recognized early on [2]. In collaborative VR, a "bent pick ray" was proposed to handle the situation when both collaborators attempt to move the same object, which is illustrated with a bending of their individual selection rays [21]. The ray bending was used to make the two collaborators aware of the conflict, and not to circumvent occluders. A bending selection ray was also used to avoid that the selected object intersects with the VE as it is moved by a user [8]. A flexible pointing ray was used to reach occluded selection targets [2]. The flexible pointer approach has the advantage of not distorting the VE. However, there is no guarantee that if the point of intersection with the VE is visible to one collaborator, then it will also be visible to the second collaborator. Furthermore, our method relies on straight, rigid virtual laser pointers which are easier to control compared to curved pointers that bend.

Occlusion has been used as a way of modulating task complexity in VR studies, where it was shown to increase task completion time. For example, when two users were asked to collaborate in VR to dock small parts, task completion time increased significantly when the walls of the boxes containing the parts were taller [12]. Taller walls increase the disparity between the views of the two users, disparity that our method addresses through view splicing.

2.5 Co-presence and co-location in collaborative VR

VR research has long considered the benefits and challenges of having more than a single user involved in a VR application. A successful collaborative VR application should make each user perceive the presence of all the other users. For some applications the users are located at different physical spaces, such as in remote technical support [14], social networking [27], distance education [5], and remote gameplay [4], and the sense of co-presence is re-enforced by rendering user avatars and by facilitating direct communication among pairs of users.

The co-location of the multiple users of a collaborative VR application has both advantages in terms of facilitating a sense of co-presence, and disadvantages, such as inconsistencies between relative physical and virtual locations of the users due to individual user redirections needed to fold the larger VE into the smaller physical space, or to physical collisions between the multiple users [4].

Our paper proposes a fundamental mechanism for alleviating the disparity between the views of two users of a collaborative VR application. We tackle the scenario where the two users are co-located, which brings the benefit of natural and accurate voice communication between the two users. In order to maintain this benefit, the collaborator is required to remain in the VE at their true (i.e., real world) relative position with respect to the user. Our method could be applied to the remote collaborative VR scenario, which is less challenging in the sense that there is more freedom in placing the remote collaborator in the VE, as their voice is conveyed to the user through a headset. Furthermore, the VR collaboration afforded by the switch camera has the potential to surpass what can be achieved in real world collaboration, where it is not possible to remove the difference between what can be seen from two disparate viewpoints without moving the viewpoints together.

3 View Splicing

We describe the design constraints that have to be met by view splicing (Section 3.1), we give an overview of our view splicing method (Section 3.2), and then we describe each step of our method in detail (Sections 3.3 through 3.6).

3.1 Design constraints

In order for a user of a VR application to work with a collaborator effectively, the visualization at the user has to meet the following design constraints:

Strict Workspace View Agreement. The user should see the workspace exactly as the collaborator does, such that any instruction can be easily interpreted by the user, without any context translation. We define the workspace as the part of the VE on which the user and their collaborator work together. The view agreement implies that, although the user and the collaborator have different viewpoints, there should be no occlusion difference between their views of the workspace [3, 40]. In other words, the collaborator should not see a part of the workspace that is hidden to the user due to occlusions, and, conversely, the user should not see a part of the workspace that the collaborator cannot see. This way, any part to which the collaborator points is visible to the user, and vice versa. The strict view agreement also implies that angles and lengths are the same for the collaborator and the user, so qualitative references to workspace geometry, e.g., about symmetry or relative size, should remain valid across the two systems of reference.

Conventional Viewing of Immediate Surroundings. The user should have a conventional first-person view of the part of the VE that is close to them, including of the avatar of the collaborator. This

implies that the user should be able to turn naturally towards the collaborator, guided by the direction of the sound of the collaborator's voice, for a normal conversation, including to see any non-verbal instructions given by the collaborator through gesture [11, 41].

Visualization Continuity. The switch from the user first-person view to the view of the collaborator should occur gradually. The switch should happen over a transition region, where geometric continuity is maintained. The continuity between the two parts of the visualization should allow the user to quickly establish connections between the two parts of the VE, for a good understanding of the overall scene, without excessive user cognitive load [20, 26, 40, 42].

The user and collaborator can swap roles, with the user using a laser pointer to indicate a part of the VE to the collaborator, and the same design constraints apply for the collaborator, symmetrically. These view splicing design constraints call for showing the user a multiperspective visualization that starts out with the conventional user view and then switches gradually to the collaborator view.

3.2 View splicing overview

We propose to facilitate VR collaboration by modifying the visualization at the user such that the user sees the part of the VE that is the focus of the collaboration the same way the collaborator does. The visualization at the collaborator is not modified, so the collaborator uses conventional VR visualization. We compute the image shown to the user with the *switch camera*, a novel multiperspective camera model that changes gradually from the user view to the collaborator view, while meeting the three design constraints enumerated above. Our view splicing pipeline has three main steps.

In a first step, our pipeline constructs a switch camera given the main user and collaborator views. These views are the default views that the user and the collaborator assume as they work together, for example standing side by side and looking at the same region of the VE. The switch camera construction routes generalized rays to connect the user view to the collaborator view (Figure 2). The switch camera construction is described in Section 3.3 and Section3.4.

In a second step, the switch camera is used to warp the VE by displacing the vertices of its geometry, such that, when rendered conventionally, the warped VE provides the desired gradual transition from the user view to the collaborator view (Figure 3). The displacement of a vertex is implemented by a projection with the switch camera, followed by an unprojection with a conventional camera. VE warping is described in Section 3.5.

In a third step, the warped VE is rendered conventionally with the current left and right user eye cameras, producing the final stereo multiperspective visualization. This multiperspective visualization starts out as a conventional visualization of the VE close to the user, and then switches to the collaborator view for the part of the VE that is the focus of collaboration. The collaborator avatar is imaged with the user view, which enables natural communication, in agreement with the direction of the collaborator voice. The rendering of the multiperspective visualization is described in Section 3.6.

The first step of the pipeline is executed every time the main user and collaborator views change. These views might remain constant throughout a collaboration session, or they might change occasionally, as is the case, for example, when the user and collaborator walk to a different location to work on a different part of the VE. The second step of the pipeline is executed every time the first step is executed, and, for dynamic VE's, for every frame. The third step of the pipeline is executed for every frame, so the user can explore the warped VE freely, at interactive rates.

3.3 Switch camera construction

The switch camera is constructed with Algorithm 1, which is illustrated in Figure 4.

The construction proceeds in the epipolar plane s defined by points V_1, V_2, and O (line 1). The algorithm computes planes s_1 and

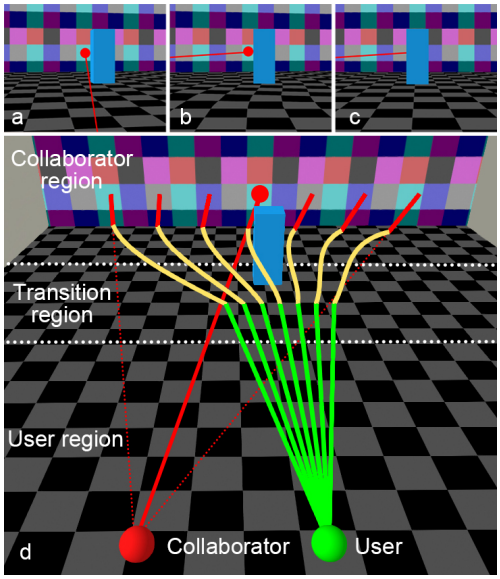

Figure 2: Switch camera view splicing concept. A collaborator uses a virtual laser pointer (a) and the laser dot is not visible to the user in a conventional view (c). We compute a multiperspective image with a switch camera (b) to show the laser dot. The switch camera partitions the VE with vertical planes into a user region, a collaborator region, and a transition region (d). A switch camera ray has three parts: a segment of a conventional user ray over the user region (green in d), a segment of a conventional collaborator ray over the collaborator region (red), and a curve that connects the two segments over the transition region (yellow).

s_2 perpendicular to V_1O and V_2O, at distances d_1 and d_2 from V_1 and V_2 (line 1). Planes s_1 and s_2 partition the user view frustum into the user view region, the transition region, and the collaborator view region. Points P_1 and P_2 are computed as the intersection of V_1O and V_2O with s_1 and s_2, respectively (line 2).

The algorithm computes a cubic Bézier curve arc that connects P_1 to P_2, using control points P_2, C_2, C_1, P_1. This Bézier arc essentially switches from the user view direction V_1P_1 to the collaborator view direction P_2O, with C^1 continuity at P_1 and P_2. The family of possible Bézier arcs is defined with two degrees of freedom, as control points C_1 and C_2 can be anywhere on segments P_1O and P_2V_2. The algorithm selects a good Bézier arc with an iterative optimization process that tries multiple locations of control points C_1 and C_2, and chooses their locations that result in the highest quality Bézier arc (lines 3-10). The candidate positions C_{2i} and C_{1j} are defined with a uniform sampling of segments V_2P_2 and P_1O (lines 4-5). The candidate Bézier arc b_{ij} (line 6) is evaluated according to Algorithm 2 (line 7), described below in Section 3.4.

The current best Bézier arc b is updated if the current Bézier arc has a strictly higher score q_{ij}, which means that ties are broken in favor of the first arc encountered (line 8). This is an acceptable strategy since any of the highest quality Bézier arcs would avoid discontinuities and produce good results. The availability of a quality Bézier arc depends on the user and collaborator viewpoints, as well as on the distances d_1 and d_2 that define the transition region. The Bézier arc and the overall switch camera construction do not depend on the VE geometry. Therefore, the construction of the switch camera succeeds robustly for typical "shoulder to shoulder" configurations, including the ones in our user study (Section 4).

The algorithm returns the switch camera (line 11), defined by the main user view (V_1, F_1), by the main collaborator view (V_2, F_2), by

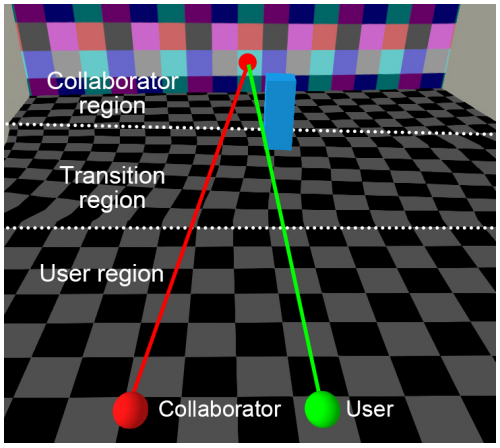

Figure 3: VE warping. The VE is warped with the switch camera such that the warped VE produces the desired multiperspective visualization when rendered with a conventional user camera.

the transition region (s_1, s_2), and by the best Bézier arc found b. The resulting switch camera rays are visualized in Figure 2d.

Algorithm 1 Switch camera construction

Input: user main view (V_1, F_1), collaborator main view (V_2, F_2), center of workspace O, distance to transition region d_1, and distance to collaborator view region d_2.
Output: Switch camera S

1: $s = Plane(V_1, V_2, O); s_1 = Plane(O - V_1, d_1); s_2 = Plane(O - V_2, d_2)$
2: $P_1 = V_1O \cap s_1; P_2 = V_2O \cap s_2;$
3: $q = 0$
4: **for each** C_{2i} on V_2P_2 **do**
5: **for each** C_{1j} on P_1O **do**
6: $b_{ij} = CubicBézier(P_2, C_{2i}, C_{1j}, P_1)$
7: $q_{ij} = EvaluateBézier(V_1, F_1, s, s_1, s_2, b_{ij})$
8: **if** $q < q_{ij}$ **then** $q = q_{ij}; b = b_{ij}$
9: **end for**
10: **end for**
11: **return** $S = (V_1, F_1, V_2, F_2, s_1, s_2, b)$

3.4 Bézier arc quality evaluation

Our switch camera construction chooses a good Bézier arc to guide the curved camera rays over the transition region. A good Bézier arc is one that connects the user view to the collaborator view without folds. Figure 5, top, illustrates a Bézier arc that is not suitable for our switch camera, as the normals to the Bézier arc intersect one another. Figure 5, bottom, shows a Bézier arc that is suitable for switch camera construction. The only difference between the two figures are the locations of control points C_1 and C_2.

Algorithm 1 finds a good position for control points C_1 and C_2 by scoring candidate Bézier arcs b according to Algorithm 2.

The algorithm computes in lines 1 and 2 the quadrilateral $(L_1L_2R_2R_1)$ that defines the transition region in the epipolar plane s, see Figure 5. Then the algorithm samples the Bézier arc b to investigate how the normals to b rule the transition region, counting the number of errors found variable e (lines 3 to 12). The Bézier arc b is traversed from P_2 down to P_1, with equal increments Δt of the Bézier curve parameter t (line 4). For each point on b, the algorithm computes the normal n_k to b (line 5), which is then intersected with the view frustum left and right boundary lines L_1L_2 and R_1R_2 (line 6). The number of errors e_k introduced by the current normal n_k is found by considering the last good, i.e. error free, normal, whose index is stored in g.

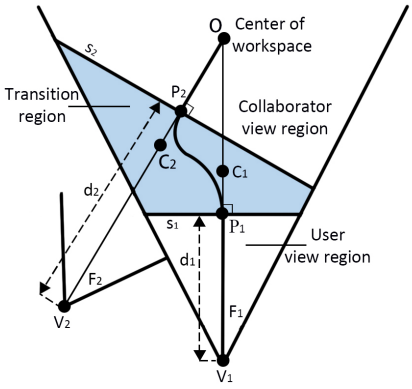

Figure 4: Switch Camera Construction. The user view direction V_1O switches to the collaborator view direction P_2O with the help of a cubic Bézier arc defined by control points P_2, C_2, C_1, and P_1.

Algorithm 2 *EvaluateBézier*

Input: V_1, F_1, epipolar plane s, transition region planes s_1 and s_2, and candidate cubic Bézier arc b
Output: q

1: $L_1 = s_1 \cap s \cap F_1; L_2 = s_2 \cap s \cap F_1;$
2: $R_1 = s_1 \cap s \cap F_1; R_2 = s_2 \cap s \cap F_1;$
3: $e = 0; g = 0$
4: **for** $t = 0, k = 0; t \leq 1; t$+=$\Delta t, k$++ **do**
5: $n_k = BézierNormal(b, t)$
6: $l_k = n_k \cap L_1L_2; r_k = n_k \cap R_1R_2; e_k = 0$
7: **if** t = 0 **then continue**
8: **if** $(l_k \notin L_1L_2) \vee (l_k \in [L_2l_g])$ **then** e_k++
9: **if** $(r_k \notin R_1R_2) \vee (r_k \in [R_2r_g])$ **then** e_k++
10: **if** $(e_k = 0)$ **then** $g = k$
11: $e += e_k$
12: **end for**
13: **return** $q = 1/(e+1)$

The errors introduced by n_k are detected by examining where it intersects the view frustum, by also taking into account the view frustum intersections of the last good normal. If intersection point l_k is outside the transition region, i.e. outside segment L_1L_2, or if l_k is closer to L_2 than the intersection point l_g of the last good normal, the number of errors e_k is incremented by 1 (line 8). For example, in Figure 5, top, the first four normals starting from and including L_2R_2 do not introduce any errors. The fifth normal intersects the last good normal l_g, i.e. the fourth normal, which is detected by out of order intersections on R_1R_2. Normals 5 to 8 (red) introduce errors.

If the current normal n_k has not introduced any error, the index of the last good normal is updated (line 10). The total number of errors e is incremented with the number of errors e_k introduced by the current normal, which could be 0, 1, or 2 (line 11). Once all points on the Bézier arc b are examined, the quality of b is returned as the inverse of the total number of errors e plus 1, which puts the possible quality values in the interval (0, 1] (line 13).

3.5 Virtual environment warping

The switch camera is used to modify the geometry of the VE by displacing its vertices, such that the warped VE produces the desired view splicing effect when rendered with a conventional camera. The warped VE is rendered with the left and right eye cameras of the

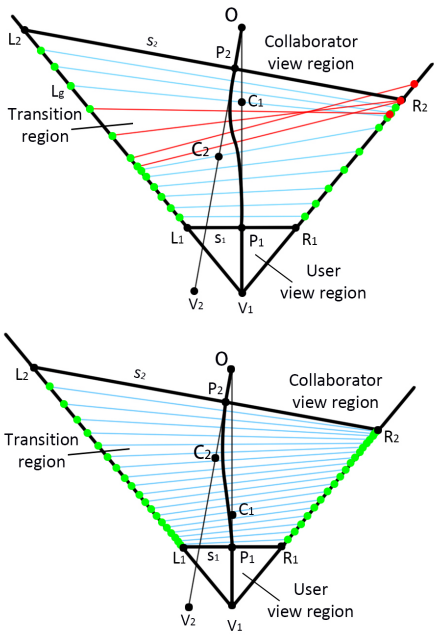

Figure 5: Example of a Bézier arc that *is not* (top) and that *is* (bottom) suitable for switch camera construction, as its normals rule the transition region with folds (top) and without folds (bottom).

current user view, generating a real-time stereo visualization for the user, based on their current head position and orientation. The vertices of the VE geometry are displaced according to Algorithm 3.

Algorithm 3 Switch camera vertex displacement

Input: VE vertex Q, switch camera $S(V_1,F_1,V_2,F_2,s_1,s_2,b)$
Output: displaced vertex Q'

1: **if** $s_1(Q) < 0$ **then** /* Q ∈ user region */
2: $Q_p.xy$ = Project$(Q,V_1,F_1).xy$
3: **else if** $s_2(Q) < 0$ **then** /* Q ∈ collaborator region */
4: $Q_p.xy$ = Project$(Q,V_2,F_2).xy$
5: **else if** $s_1(Q) \geq 0 \wedge s_2(Q) \geq 0$ **then** /* Q ∈ transition region */
6: t = BinarySearch(Q,b)
7: P_t = BézierPoint(b,t); n_t = BézierNormal(b,t)
8: $(x_t,y_t) = (P_tQ \cdot n_t, P_tQ \cdot n_s)$
9: $f = V_1P_1/(V_2P_2 + (V_1P_1 - V_2P_2)t)$
10: $n_1 = (L_1 - P_1)/L_1P_1$
11: $Q_1 = P_1 + n_1x_tf + n_sy_tf$
12: $Q_p.xy$ = Project$(Q_1,V_1,F_1).xy$
13: **end if**
14: $Q_p.z$ = Project$(Q,V_1,F_1).z$
15: **return** Q' = Unproject(Q_p,V_1,F_1)

The algorithm displaces Q by first projecting it with the switch camera to the image plane location Q_p of the main user view (V_1,F_1) (lines 1-14), and then by unprojecting Q_p to Q' with the conventional user camera (line 15).

The projection with the switch camera proceeds according to the region to which vertex Q belongs. If Q is in the user region, then Q is projected conventionally with the user view (V_1,F_1) (lines 1-2). For now, the algorithm only computes the image coordinates of the projection, i.e. x and y. If Q belongs to the collaborator region, then Q is projected conventionally with the collaborator view (V_2,F_2) (lines 3-4). If Q belongs to the transition region, the projection uses the Bézier arc of the switch camera to blend in between the collaborator and user view projections, based on the position of Q within the transition region (lines 5-13). If Q is close to s_2, the projection is similar to a conventional projection from the collaborator view. The closer Q is to s_1, the more its projection is similar to a conventional projection from the user view. The switch camera projection is continuous at s_1 and s_2. In other words, vertices on s_2 (s_1) are projected at the same location with both the collaborator (user) and the transition region projections.

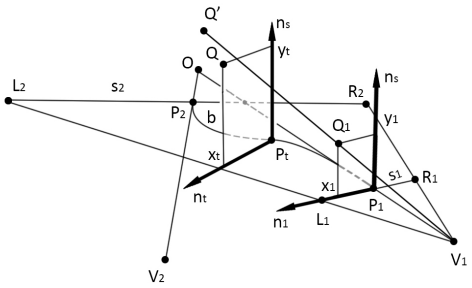

Figure 6: Displacement of transition region vertex Q to Q'.

The algorithm finds a plane perpendicular to the Bézier arc b that contains Q, using a binary search on the Bézier parameter t (line 6). In Figure 6, the plane found is the plane with 2D coordinate system (P_t,n_t,n_s). The initial search interval is [0, 1], defining planes s_2 and s_1, which are known to sandwich any vertex Q in the transition region. Then binary search proceeds to halving the interval while keeping Q in between the two interval endpoint planes. Using the value t found through binary search, the algorithm defines point P_t on b, and the normal n_t to b (line 7).

The algorithm computes the 2D coordinates (x_t,y_t) of Q (line 8). Then the algorithm computes the projection Q_1 of Q on s_1 by scaling (x_t,y_t) to account for the position of Q relative to s_1 and s_2 (lines 9-11). The scaling factor f is V_1P_1/V_2P_2 when Q is on s_2, i.e. t = 0, and $f = 1$ when Q is on s_1, i.e. t = 1 (line 9). Q_1 is constructed in plane s_1 with local 2D coordinate system (P_1,n_1,n_s), where n_1 is the normalized vector from P_1 to L_1 (line 10). The 2D coordinates of Q_1 are the coordinates (x_t,y_t) of Q scaled by f (line 11). Finally, the projection Q_p of Q is obtained by projecting Q_1 conventionally with the main user view (line 12).

In all cases, the z of the projected vertex Q_p is the z obtained through conventional projection with the main user view (line 14). Q_p is then pushed back into 3D space by conventional unprojection with the main user view, to obtain the displaced vertex Q'.

3.6 Multiperspective VR visualization

The displaced vertices define a warped VE that produces the desired view splicing effect when rendered with the current left and right eye user cameras. Using Figure 3 again, the warped VE moves the occluder away so the user gains line of sight to the laser dot. Since both the original and the displaced vertex have the same depth with respect to the user position, the user perceives depth with the switch camera multiperspective image the same way as in conventional VR.

The resulting multiperspective visualization provided to the user abides by the three view splicing design constraints: the user sees the focus of collaboration exactly the same way the collaborator does, the part of the VE close to the user is rendered conventionally, with the user's first-person view, and, finally, the transition between the two parts of the visualization is gradual and continuous, due to the continuity of the switch camera projection from the collaborator to the transition and to the user regions. Figure 7 illustrates the view splicing achieved by the switch camera multiperspective visualization. This precise view splicing cannot be achieved with

prior art multiperspective visualization techniques such as the graph camera [37]. Changing from the user to the collaborator perspective with a graph camera produces an image of the collaborator region that is different from what the collaborator sees (Figure 8).

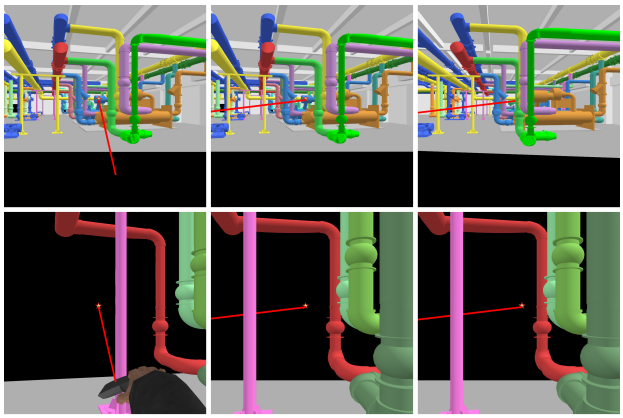

Figure 7: Illustration of the view splicing effect achieved by the switch camera, for the example shown in Figure 1. With the switch camera, the user sees the collaborator region the same way the collaborator does (top row), and the user sees the user region conventionally, from their own perspective (bottom). The laser beam is shown in its entirety in all images. The laser dot is not occluded in the bottom right image because the collaborator region geometry is not shown.

4 User Study

We have evaluated our view splicing visualization in a controlled user study. One control condition is conventional VR visualization, where the user moves around to see what the collaborator is pointing to, and another is view toggling, where the user changes to the collaborator's view and back at the press of a button.

4.1 Design

Participants. We have recruited 16 participants, 14 male and 2 female, between 20 and 40 years old. Two of our participants had used HMD VR applications before. Participants had normal and corrected vision, and none reported preexisting balance disorders. The collaborative VR application involved a collaborator and a user. Each participant served as the user, and a member of our research team acted as the collaborator.

Hardware and Software Implementation. For the collaborator we used a HTC Cosmos VR system with two hand-held controllers, which allowed the collaborator to point at the VE with a virtual laser and to make two arm gestures. For the user we used an HTC Vive VR system with a tracked hand-held controller, which allowed the user to select their answer. Each HMD was connected to its own workstation with a 3.6GHz Intel(R) Core(TM) i7-9900KF CPU, 32GB of RAM, and an NVIDIA GeForce GTX 1060 graphics card. The tracked physical space for the user was a 4m×4m empty space, which was sufficient to avoid redirection and teleportation. The collaborator did not change position during the experiments.

The user multiperspective visualization is rendered for each eye at 60fps. The multiperspective visualization pipeline is implemented on the GPU, with GLSL shaders within the OpenVR SDK. The switch camera construction step takes 1.5ms, the VE warping step takes 2.3ms, and the step of rendering the warped VE in stereo takes 0.7ms. Switch camera construction is implemented using a compute shader. The algorithm investigates 150 positions for the control point C_2 (line 4 in Algorithm 1) and 100 positions for the control point C_1 (line 5), for a total of 15,000 candidate Bézier arcs. A candidate Bézier arc is sampled uniformly with 20 equidistant parameter t values (line 4 in Algorithm 2). These parameters were established through tests that revealed that they explore the three dimensional switch camera design space in sufficient detail to converge to a quality solution robustly, with higher parameter values bringing only insignificant improvements of the Bézier arc quality, which was judged both quantitatively, through the quality score (Algorithm 2), and qualitatively, through visual inspection. VE warping is implemented using a geometry shader which subdivides the VE triangles and displaces the vertices (Algorithm 3). Triangle subdivision is needed to model the non-linear switch camera with high fidelity. The warped VE is rendered conventionally for each eye.

Figure 8: Collaborator region rendered conventionally from the collaborator view (left), rendered with a graph camera [37] (middle), and the difference between the two renderings (right). The graph camera construction switches from the user to the collaborator view with one bend at plane s_2, see Figure 4.

Task 1. The user study had three tasks. For the first task, the collaborator pointed to a pipe in a virtual factory and the user had to indicate the color of the pipe to which the collaborator is pointing (Fig. 1). The factory covers a 16m×12m area. The collaborator points with a virtual laser pointer aimed with the handheld controller. The laser beam is visualized with a red line. The laser dot, i.e. the intersection between the laser beam and the VE geometry, is visualized with a star placed in the 3D VE, of size 20cm. The user indicates the color of the pipe by selecting it from a palette of 3×3 colors displayed at the bottom of the screen. In the initial position, the user is 1.5m away from the collaborator. Due to the complex 3D pattern of pipes, the laser dot could be hidden from the user due to occlusions. For *Task 1*, there were two conditions. In the control condition CC_1, a participant uses conventional VR visualization. In order to overcome occlusions and to actually see the pipe to which the collaborator is pointing, the user would be required to walk around. In the experimental condition EC, the user benefited from our view splicing visualization based on the switch camera. The user engages the switch camera visualization with the press of a button at the beginning of the collaboration session. The switch camera visualization is then maintained until the end of collaboration session.

Task 2. The second task uses the same factory VE. The collaborator pointed to a pipe, then gestured using their arms, and then pointed to a second pipe; the user's task was to indicate the color of the first pipe, the collaborator gesture, and the color of the second pipe. The user answers were collected by asking them to select a color-gesture-color combination from six options (Figure 9). While the collaborator was pointing to pipes and gesturing, the collaborator would tell the user to 'look at this pipe', then to 'look at me', and finally to 'look at this other pipe'. The collaborator pointed to the first pipe, held the gesture, and then pointed to the second pipe for three seconds each.

The second task was performed by each participant in a control condition and in an experimental condition. In the control condition CC_2, the user was able to toggle between their own view and the collaborator's view using a controller button. Starting from the user

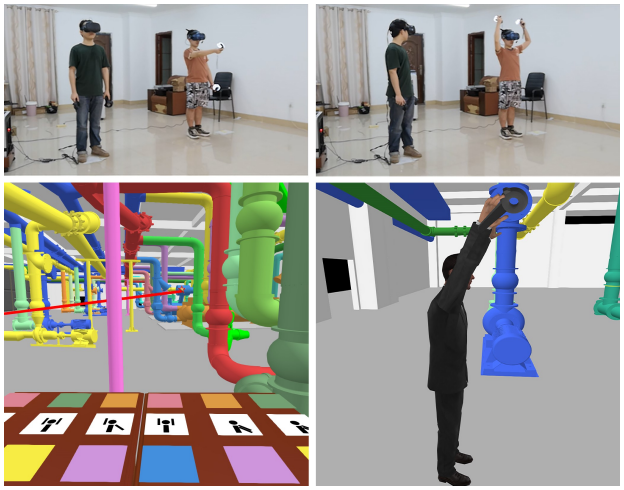

Figure 9: User study task 2. The collaborator points to a pipe (top left) and then makes a gesture (top right), and the user is asked to select the correct pipe color-gesture combination. Using the switch camera multiperspective visualization, the user can see the laser dot (bottom left) and can also naturally turn their head towards the collaborator to see the gesture (bottom right).

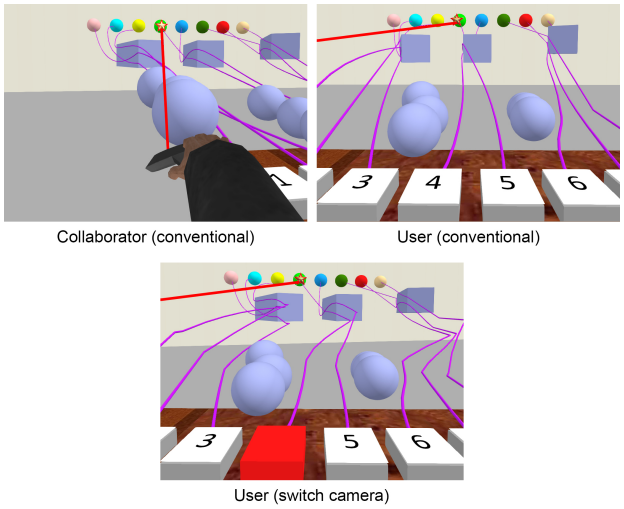

Figure 10: User study task 3. The user is asked to press the button that is connected to the sphere to which the collaborator is pointing. Due to occlusions and to the non-linear trajectory of the wires, the user cannot easily find the correct button with a conventional user view alone (top right), or by toggling back and forth between the user and the collaborator view (top left and right). The switch camera simplifies the task by allowing the user to trace the wire from the sphere to the correct button (bottom).

view, the user first had to toggle to the collaborator view to see the first pipe to which the collaborator was pointing. Then the user had to toggle back to their own view in order to be able to see the collaborator avatar by rotating their head, as the collaborator avatar is not visible from the collaborator's view. Then the user had to toggle to the collaborator view again to see the second pipe to which the collaborator was pointing; finally the user had to toggle back to the user view to select the answer. In this control condition the user has the ability to instantaneously assume a view that shows the laser dot, i.e. the view of the collaborator. *Task 2* requires complex communication with a rapidly changing visual reference, and the user is unable to complete the task by walking around to find a view that disoccludes the laser dot, as they did in CC_1 for *Task 1*. In the experimental condition *EC*, the user benefited from our view splicing visualization, which allowed the user to switch focus between the pipes to the collaborator through natural head rotations, without repeated button presses.

Task 3. The third task used a VE with eight spheres connected with wires to eight buttons (Figure 10). The wires are tangled such that one cannot tell which button is connected to a given sphere unless one could trace visually the wire from the sphere to its button. The collaborator pointed to a sphere and the user had to press the button to which the sphere is connected. Each participant performed *Task 3* in each of three conditions: the control condition CC_1 from *Task 1*, where the user used conventional visualization with walking around, the control condition CC_2 from *Task 2*, where the user toggled between their own and the collaborator's view, and the experimental condition *EC*, where the user benefits from our view splicing visualization. In CC_1, a participant had to walk such that they maintain the wire in view as they were tracing it from the sphere to the button. In CC_2, neither the collaborator nor the user view would show the entire wire, so each participant had to find the wire in the user view based on their recollection of the collaborator view. In *EC*, the view splicing visualization would show the wire continuously, from the sphere to the button, albeit distorted as the wire crossed the transition region between the collaborator view region and the user view region.

Experimental design. Our study used a within-subject design, with each participant serving in all conditions, for all tasks. Each task was repeated eight times for each condition, using different target pipes, collaborator gestures, and spheres. Condition and trial order was randomized. The eight trials of a task for a given condition were completed in about one minute. Participants had a 10 minute break between the conditions of the same task, and the tasks were performed on consecutive days. Before each task, participants were given a three minute tutorial to understand the task and to practice providing their answer. During the task, the system displayed to each participant the current number of correct answers.

4.2 Results

We have quantified the user study results using task performance (Section 4.2.1) and perceptual (Section 4.2.2) metrics.

4.2.1 Task performance metrics

The performance of user participants was quantified by counting the number of errors committed by each participant, for each task, and for each condition. For a trial, any incorrect answer counts as one error. Since there were eight trials per task and per condition, the number of errors is an integer between 0 and 8. Table 1 reports the average and the standard deviation of the number of errors over all participants, for each task, and for each condition. For each control condition, the table reports the relative difference between the *CC* and *EC* averages (column 4). A Shapiro-Wilk test reveals that the number of errors is not normally distributed, so we performed the statistical analysis of the differences between the control and experimental conditions, using the Wilcoxon signed-rank test, see p values in column 5. We also report Cohen's d (column 6), along with the conventional effect size quantification (column 6) [9, 38].

For all tasks, participants committed the fewest number of errors in the experimental condition *EC*. The user can see the workspace like the collaborator does, so the user can easily see the pipe to which the collaborator is pointing in *Task 1*. The user can see the collaborator with a natural head rotation, so the user can keep up with the alternating visual reference in *Task 2*. The continuous

visualization of the VE lets the user trace the wire from the sphere to its corresponding button in *Task 3*.

Participants committed very few errors in CC_1, where they walked to remove occlusions and gain line of sight to the laser dot for *Task 1*, and where they walked to trace the wire from the sphere to the button for *Task 3*. The advantage of *EC* over CC_1 is not statistically significant. However, in CC_1, participants were able to complete the tasks at the cost of a significant increase in effort, as discussed below. CC_2 participants committed a significantly larger number of errors compared to *EC* participants: in *Task 2* ($p = 0.023$), they often failed to synchronize the view toggling to the collaborators alternating pointing and gestures; in *Task 3* ($p = 0.011$), they often had no chance to trace the connection from the sphere to the button due to visualization discontinuity.

As shown in Table 1, CC_1 participants are able to perform the tasks with very few errors. However, this comes at the cost of significant additional effort. We quantify the effort difference with two metrics: viewpoint translation and task completion time. Table 2 shows that participants had to walk significantly farther in CC_1 than in *EC*. For *Task 1* ($p < 0.001$), *EC* participants had only minimal viewpoint translations due to normal head motions. For *Task 3* ($p < 0.001$), *EC* participants had to step forward to press the button, hence the slightly larger viewpoint translations. The additional VE navigation for CC_1 translates to longer task completion times, as shown in Table 3, and the difference is statistically significant for *Task 1* ($p = 0.05$). Both viewpoint translation and task completion time are normally distributed so the mean differences were analyzed with a repeated measures ANOVA.

Table 1: Number of errors for each task and each condition.

Task	Condition	Avg ± std. dev.	(CC-EC) / CC	p	Cohen's d	Effect size
Task 1	*EC*	0 ± 0				
	CC_1	0.17 ± 0.37	100%	0.317	0.63	Medium
Task 2	*EC*	0.125 ± 0.33				
	CC_2	1.125 ± 0.92	88.9%	0.023	1.44	Very large
Task 3	*EC*	0.125 ± 0.33				
	CC_1	0.125 ± 0.33	0%	1	0	-
	CC_2	3.25 ± 0.97	96%	0.011	4.32	Huge

Table 2: Viewpoint translation, in meters.

Task	Condition	Avg ± std. dev.	(CC-EC) / CC	p	Cohen's d	Effect size
Task 1	*EC*	0.29 ± 0.042				
	CC_1	5.99 ± 0.98	95%	< 0.001	8.22	Huge
Task 3	*EC*	9.43 ± 1.26				
	CC_1	16.92 ± 1.78	44%	< 0.001	4.86	Huge

Table 3: Task completion time, in seconds.

Task	Condition	Avg ± std. dev.	(CC-EC) / CC	p	Cohen's d	Effect size
Task 1	*EC*	28.23 ± 4.62				
	CC_1	38.81 ± 9.77	27.2%	0.05	1.38	Very large
Task 3	*EC*	47.34 ± 7.11				
	CC_1	54.53 ± 8.84	13.2%	0.11	0.89	Large

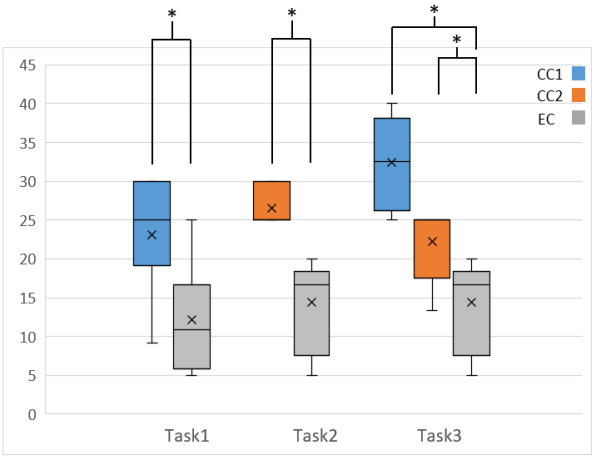

Figure 11: Participant task load per task and per condition.

4.2.2 Perceptual metrics

We have also investigated task load, simulator sickness, and presence using standard questionnaires.

Table 4: Simulator Sickness Questionnaire data.

Task	Condition	preAvg ± std. dev.	postAvg ± std. dev.	p
Task 1	*EC*	1.39 ± 1.90	1.25 ± 1.75	0.56
	CC_1	2.50 ± 2.80	2.74 ± 2.67	0.71
Task 2	*EC*	1.47 ± 1.79	1.72 ± 1.58	0.41
	CC_2	1.59 ± 2.09	2.43 ± 2.13	0.29
Task 3	*EC*	1.25 ± 0.83	1.75 ± 0.97	0.10
	CC_1	1.84 ± 1.94	3.40 ± 1.88	0.10
	CC_2	1.71 ± 1.70	2.09 ± 1.39	0.26

We measured task load using Raw TLX [15, 16] (Figure 11). We averaged the scores of the six Raw TLX task load questions, which are between 5 and 100, with 5 point increments. The task load values do not follow a normal distribution so the differences were analyzed with the Wilcoxon signed-rank test. The task load is significantly higher for CC_1 and for CC_2, compared to *EC*. We attribute this expected difference to the additional amount of navigation and view toggling required by CC_1 and CC_2, compared to *EC*, for which the user engages the switch camera visualization just once.

We measured simulator sickness using the standard Simulator Sickness Questionnaire [24] (Table 4). The SSQ was administered pre-experiment and post-experiment for each task and each condition. The SSQ scores are not normally distributed and the pre to post differences were analyzed with the Shapiro-Wilk test. None of the differences are statistically significant, and the absolute value of the SSQ scores remains small. We conclude that, in these experiments, simulator sickness is not more of a concern for view splicing than it is for conventional VR visualization or for view toggling. This is expected since the multiperspective effect is akin to replacing he VE with a slightly modified one, which is then viewed conventionally.

We measured users' sense of presence in the VE using the standard Igroup Presence Questionnaire (IPQ) [39]. Table 5 shows our IPQ measurements broken down into the usual categories of general presence (GP), spatial presence (SP), involvement (INV), and realism (REAL). For each category the score ranges from 0 to 6. The experimental condition produces IPQ scores similar to both

control conditions and no difference is significant. However, the small size of our within-subject study is insufficient to shed light on any presence differences between the three conditions.

Table 5: Igroup Presence Questionnaire data.

Task	Condition	GP	SP	INV	REAL
Task 1	*EC*	4.7 ± 0.5	4.4 ± 0.7	4.2 ± 0.7	2.9 ± 0.9
	CC_1	4.3 ± 0.9	4.6 ± 0.6	3.9 ± 0.8	2.9 ± 1.1
Task 2	*EC*	4.8 ± 0.4	4.4 ± 0.3	3.4 ± 0.5	2.8 ± 0.7
	CC_2	4.8 ± 0.4	4.0 ± 0.4	3.2 ± 0.6	2.6 ± 0.9
Task 3	*EC*	4.3 ± 0.4	4.2 ± 0.2	3.1 ± 0.6	2.7 ± 0.8
	CC_1	5.1 ± 0.6	4.2 ± 0.4	3.3 ± 0.5	2.7 ± 0.8
	CC_2	4.5 ± 0.5	4.2 ± 0.4	3.2 ± 0.5	2.6 ± 0.9

4.3 Limitations

One limitation of our approach is that it requires separation between the collaborators and the workspace, for the switch camera to have space to transition from the user to the collaborator perspective. The transition region is defined by the depths d_1 and d_2, which are input to Algorithm 1. For our experiments, the user and the collaborator stand approximately 1.5m apart, with a d_1 and d_2 that are 6m and 10m for *Task 1* and *Task 2*, and 2.5m and 5m for *Task 3*. Like in depth from stereo, the switch camera design is specified up to a constant of proportionality. The closer together the user and the collaborator, the smaller the disparity between their views, and reducing the distances d_1 and d_2 can be done by reducing the distance between the collaborators.

Another limitation of our approach is the requirement that the user and their collaborator work "shoulder to shoulder". This is a frequent collaboration scenario, where the user and the collaborator explore the VE from the "inside looking out". In its present form, our approach cannot handle the face to face collaboration scenario, where the VE is explored from the "outside looking in". Our method aims to provide the exact same view to the user and their collaborator, and this cannot and should not be achieved in the face to face scenario. An alternative construction could route rays over the top of the VE geometry, making both views a top down view, with a 180 degree rotation between them, to reflect the opposite view direction of the user and their collaborator. Similarly, our method is not intended for the scenario where the collaborator is in front of the user, which would require a substantial displacement of the workspace to avoid its occlusion by the collaborator.

Our method was presented in the context of two collaborating users. However, as is, our method can support a group of N users who adopt a common view of the VE, e.g., a view from the center of the group. To accommodate the "shoulder to shoulder" restriction discussed above, the group has to be located together in one region of the VE, and to collaborate on a region of the VE that is outside of the group's region. This way, all users can refer to the same element of the VE beyond the transition planes, free of occlusions, enabling collaboration between any two and all of the users.

Our method works by warping the VE geometry to connect the user view to the collaborator view. The larger the distance between the collaborators and the closer the workspace, the larger the deformation introduced by the warp. We quantify the deformation through the maximum curvature along the central ray of the switch camera, which is 0.35 m^{-1} for *Task 1* and *Task 2*, and 1.0 m^{-1} for *Task 3*. As confirmed by the reported spatial perception (SP) score of the IPQ, this deformation did not affect spatial perception significantly in our experiments. Indeed, our method has larger (better) SP values than CC2 (toggling between views), and only occasionally slightly worse ($\sim$3%) than CC1 (conventional VR visualization). Whereas in our examples the deformation was relatively small, future work should investigate the thresholds for acceptable deformation.

Finally, our study involved a small number of participants, which was sufficient to show significant advantages for our method over the conventional approaches in terms of task performance and task load. Future work should investigate the presence differences between our method and prior art in larger, between-subject studies.

5 Conclusions

We have presented a method for alleviating the view difference between two collaborators who work in virtual reality, side by side, in the same virtual environment. The view difference is alleviated using the switch camera, a novel multiperspective camera model that starts out from one perspective and gradually transitions to a second perspective. Our method facilitates identification and connection tracing in collaborative VR. Identifying points of reference in the VE, as well as their connection to other elements, is not a specific task, but rather a fundamental requirement that enables collaboration in most tasks. Therefore our method is an infrastructure contribution, with broad applicability. In addition to the factory floor scenario illustrated in our paper, other scenarios include a guide in a virtual natural history museum explaining an exhibit to virtual visitors, a history teacher explaining the organization of an ancient city, or an interior designer discussing a proposed design with their client.

Prior work has established specific advantages of multiperspective visualization over prior occlusion management approaches such as transparency and top-view visualization [42]. Although it is plausible that these advantages are inherited by the switch camera multiperspective visualization, future work should compare the switch camera to other occlusion management approaches, such as a picture-in-picture or auxiliary display of the collaborator's point of view, and transparency or translation of occluding elements.

View splicing is an example of VR visualization that grants the user overt superpowers that are intuitive to invoke, that bring performance gains, and that do not lead to discomfort, research direction that we anticipate will lead to further advances in VR interfaces.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China through Projects 61932003 and 61772051, by National Key RD plan 2019YFC1521102, by the Beijing Natural Science Foundation L182016, by the Beijing Program for International ST Cooperation Project Z191100001619003, by the funding of Shenzhen Research Institute of Big Data(Shenzhen 518000).

References

[1] M. Agrawala, A. C. Beers, I. McDowall, B. Fröhlich, M. Bolas, and P. Hanrahan. The two-user responsive workbench: support for collaboration through individual views of a shared space. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pp. 327–332, 1997.

[2] O. Alex and F. Steven. The flexible pointer: An interaction technique for selection in augmented and virtual reality. In *Proc. UIST*, vol. 3, pp. 81–82, 2003.

[3] F. Argelaguet, A. Kunert, A. Kulik, and B. Froehlich. Improving co-located collaboration with show-through techniques. In *2010 IEEE Symposium on 3D User Interfaces (3DUI)*, pp. 55–62. IEEE, 2010.

[4] F. Born, P. Sykownik, and M. Masuch. Co-located vs. remote gameplay: The role of physical co-presence in multiplayer room-scale vr. In *2019 IEEE Conference on Games (CoG)*, pp. 1–8. IEEE, 2019.

[5] S. T. Bulu. Place presence, social presence, co-presence, and satisfaction in virtual worlds. *Computers & Education*, 58(1):154–161, 2012.

[6] S. Butscher, S. Hubenschmid, J. Müller, J. Fuchs, and H. Reiterer. Clusters, trends, and outliers: How immersive technologies can facilitate the collaborative analysis of multidimensional data. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2018.

[7] W. Chen, C. Clavel, N. Férey, and P. Bourdot. Perceptual conflicts in a multi-stereoscopic immersive virtual environment: Case study on face-to-face interaction through an avatar. *Presence: Teleoperators and Virtual Environments*, 23(4):410–429, 2014.

[8] M. L. Chenechal, J. Lacoche, J. Royan, T. Duval, V. Gouranton, and B. Arnaldi. When the giant meets the ant an asymmetric approach for collaborative and concurrent object manipulation in a multi-scale environment. In *2016 IEEE Third VR International Workshop on Collaborative Virtual Environments (3DCVE)*, pp. 18–22, March 2016. doi: 10.1109/3DCVE.2016.7563562

[9] J. Cohen. *Statistical power analysis for the behavioral sciences*. Academic press, 2013.

[10] J. Cui, P. Rosen, V. Popescu, and C. Hoffmann. A curved ray camera for handling occlusions through continuous multiperspective visualization. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1235–1242, 2010.

[11] J. Du, Y. Shi, Z. Zou, and D. Zhao. Covr: Cloud-based multiuser virtual reality headset system for project communication of remote users. *Journal of Construction Engineering and Management*, 144(2):04017109, 2018.

[12] J. G. Grandi, H. G. Debarba, I. Bemdt, L. Nedel, and A. Maciel. Design and assessment of a collaborative 3d interaction technique for handheld augmented reality. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 49–56. IEEE, 2018.

[13] R. Gupta and R. I. Hartley. Linear pushbroom cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):963–975, 1997.

[14] S. Guven, M. Podlaseck, and G. Pingali. Exploring co-presence for next generation technical support. In *2009 IEEE Virtual Reality Conference*, pp. 103–106. IEEE, 2009.

[15] S. G. Hart. Nasa-task load index (nasa-tlx); 20 years later. In *Proceedings of the human factors and ergonomics society annual meeting*, vol. 50, pp. 904–908. Sage Publications Sage CA: Los Angeles, CA, 2006.

[16] S. G. Hart and L. E. Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology*, vol. 52, pp. 139–183. Elsevier, 1988.

[17] K. Higuch, R. Yonetani, and Y. Sato. Can eye help you? effects of visualizing eye fixations on remote collaboration scenarios for physical tasks. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 5180–5190, 2016.

[18] H. Hua, L. D. Brown, and C. Gao. Scape: supporting stereoscopic collaboration in augmented and projective environments. *IEEE Computer Graphics and Applications*, 24(1):66–75, Jan 2004. doi: 10.1109/MCG.2004.1255811

[19] H. Ibayashi, Y. Sugiura, D. Sakamoto, N. Miyata, M. Tada, T. Okuma, T. Kurata, M. Mochimaru, and T. Igarashi. Dollhouse vr: a multi-view, multi-user collaborative design workspace with vr technology. In *SIGGRAPH Asia 2015 Emerging Technologies*, pp. 1–2. 2015.

[20] A. Irlitti, T. Piumsomboon, D. Jackson, and B. H. Thomas. Conveying spatial awareness cues in xr collaborations. *IEEE transactions on visualization and computer graphics*, 25(11):3178–3189, 2019.

[21] Kai Riege, T. Holtkamper, G. Wesche, and B. Frohlich. The bent pick ray: An extended pointing technique for multi-user interaction. In *3D User Interfaces (3DUI'06)*, pp. 62–65, March 2006. doi: 10.1109/VR.2006.127

[22] S. Kasahara, S. Nagai, and J. Rekimoto. Jackin head: Immersive visual telepresence system with omnidirectional wearable camera. *IEEE transactions on visualization and computer graphics*, 23(3):1222–1234, 2016.

[23] D. F. Keefe, D. Acevedo, J. Miles, F. Drury, S. M. Swartz, and D. H. Laidlaw. Scientific sketching for collaborative vr visualization design. *IEEE Transactions on Visualization and Computer Graphics*, 14(4):835–847, 2008.

[24] R. S. Kennedy, N. E. Lane, K. S. Berbaum, and M. G. Lilienthal. Simulator sickness questionnaire: An enhanced method for quantifying simulator sickness. *The international journal of aviation psychology*, 3(3):203–220, 1993.

[25] A. Kulik, A. Kunert, S. Beck, R. Reichel, R. Blach, A. Zink, and B. Froehlich. C1x6: a stereoscopic six-user display for co-located collaboration in shared virtual environments. *ACM Transactions on Graphics (TOG)*, 30(6):1–12, 2011.

[26] A. Kunert, T. Weissker, B. Froehlich, and A. Kulik. Multi-window 3d interaction for collaborative virtual reality. *IEEE transactions on visualization and computer graphics*, pp. 1 – 1, 2019. doi: 10.1109/TVCG.2019.2914677

[27] M. E. Latoschik, F. Kern, J.-P. Stauffert, A. Bartl, M. Botsch, and J.-L. Lugrin. Not alone here?! scalability and user experience of embodied ambient crowds in distributed social virtual reality. *IEEE transactions on visualization and computer graphics*, 25(5):2134–2144, 2019.

[28] G. Lee, H. Kang, J. Lee, and J. Han. A user study on view-sharing techniques for one-to-many mixed reality collaborations. In *Proceedings of IEEE Conference on Virtual Reality and 3D User Interfaces*, 2020.

[29] C. Lin, E. Rojas-Muñoz, M. E. Cabrera, N. Sanchez-Tamayo, D. Andersen, V. Popescu, J. A. B. Noguera, B. Zarzaur, P. Murphy, K. Anderson, D. Thomas, G. Clare, and W. Juan. How about the mentor? effective workspace visualization in ar telementoring. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 212–220. IEEE, 2020.

[30] H. Lorenz, M. Trapp, J. Döllner, and M. Jobst. Interactive multi-perspective views of virtual 3d landscape and city models. In *In Proc. 11th International Conference on GI Sciences*, p. 301–321, 2008.

[31] J. Müller, R. Rädle, and H. Reiterer. Virtual objects as spatial cues in collaborative mixed reality environments: How they shape communication behavior and user task load. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 1245–1249, 2016.

[32] J. Müller, R. Rädle, and H. Reiterer. Remote collaboration with mixed reality displays: How shared virtual landmarks facilitate spatial referencing. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pp. 6481–6486, 2017.

[33] T. T. H. Nguyen and T. Duval. A survey of communication and awareness in collaborative virtual environments. In *2014 International Workshop on Collaborative Virtual Environments (3DCVE)*, pp. 1–8, March 2014. doi: 10.1109/3DCVE.2014.7160928

[34] S. Pasewaldt, M. Trapp, and J. Dollner. Multiscale visualization of 3d ¨ geovirtual environments using view-dependent multi-perspective views. *Journal of WSCG*, 19:111–118, 2011.

[35] T. Piumsomboon, A. Dey, B. Ens, G. Lee, and M. Billinghurst. The effects of sharing awareness cues in collaborative mixed reality. *Frontiers in Robotics and AI*, 6:5, 2019. doi: 10.3389/frobt.2019.00005

[36] D. Pohl and C. F. de Tejada Quemada. See what i see: Concepts to improve the social acceptance of hmds. In *2016 IEEE Virtual Reality (VR)*, pp. 267–268. IEEE, 2016.

[37] V. Popescu, P. Rosen, and N. Adamo-Villani. The graph camera. *ACM Transactions on Computer Graphics*, 8(5), 2009.

[38] S. S. Sawilowsky. New effect size rules of thumb. *Journal of Modern Applied Statistical Methods*, 8(2):26, 2009.

[39] T. W. Schubert. The sense of presence in virtual environments: A three-component scale measuring spatial presence, involvement, and realness. *Z. für Medienpsychologie*, 15(2):69–71, 2003.

[40] X. Tong, C. Li, and H.-W. Shen. Glyphlens: View-dependent occlusion management in the interactive glyph visualization. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):891–900, 2016.

[41] L. Wang, J. Wu, X. Yang, and V. Popescu. Vr exploration assistance through automatic occlusion removal. *IEEE transactions on visualization and computer graphics*, 25(5):2083–2092, 2019.

[42] L. Wang, H. Zhao, Z. Wang, J. Wu, B. Li, Z. He, and V. Popescu. Occlusion management in vr: A comparative study. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 708–716. IEEE, 2019.

[43] M.-L. Wu and V. Popescu. Efficient vr and ar navigation through multi-perspective occlusion management. *IEEE transactions on visualization and computer graphics*, 24(12):3069–3080, 2017.