Automatic Speed Adjustment for Travel through Immersive Virtual Environments based on Viewpoint Quality

Sebastian Freitag* Benjamin Weyers* Torsten W. Kuhlen*

Visual Computing Institute, RWTH Aachen University — JARA – High-Performance Computing

Abstract

When traveling virtually through large scenes, long distances and different detail densities render fixed movement speeds impractical. However, to manually adjust the travel speed, users have to control an additional parameter, which may be uncomfortable and requires cognitive effort. Although automatic speed adjustment techniques exist, many of them can be problematic in indoor scenes. Therefore, we propose to automatically adjust travel speed based on viewpoint quality, originally a measure of the informativeness of a viewpoint. In a user study, we show that our technique is easy to use, allowing users to reach targets faster and use less cognitive resources than when choosing their speed manually.

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities; I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction technqiues

1 Introduction

For traveling through Immersive Virtual Environments (IVEs), often *steering* techniques are used for travel. In contrast to *target-based* techniques, they require no explicit specification of a target location, are usually easy to understand and provide the user with a high level of control [3]. However, in large scenes, it can take a long time to reach a destination using a fixed speed. Furthermore, just increasing the travel speed globally is usually not an option if travel accuracy in detailed regions should be maintained. As an alternative, interfaces to change the travel speed interactively can be used. However, they require the user to control an additional parameter while traveling, potentially increasing cognitive load. As travel is usually only a supporting task [3], this can be undesirable.

Therefore, methods that automatically control the travel speed based on the user's surroundings have been developed. Mackinlay et al. proposed to change the travel speed depending on the distance to a target object selected by the user [7]. Kopper et al. define different discrete levels of scale that the user can change between, using the hierarchy of the scene models to determine the relative scales [6]. In contrast, Argelaguet and Andújar adapt the speed along a non-interactive camera path using optical flow and image saliency [2]. Based on the observation that in general, users want to travel faster when they are further away from scene objects [13], Ware and Fleet proposed an approach that samples the depth buffer to estimate the distances to the environment in the field of view [12]. Based on this, they adapted the travel speed in different ways, achieving the best results when the movement speed was calculated from the distance to the closest object. In a combination of both approaches, Argelaguet later used the distance to the environment, the current speed, as well as optical flow for automatic speed adjustment in interactive travel [1]. In the Cubemap approach, the distance to the environment in all directions is considered for speed selection by rendering a depth cubemap from the camera position [8]. This method was later improved upon by additionally considering the distance to the scene in the travel direction, smoothed over time [10]. In comparison, Taunay et al. move the distance calculation to the CPU, using cell-based clustering and k-d trees to speed up the computation [9].

*e-mail: {freitag | weyers | kuhlen}@vr.rwth-aachen.de

However, methods based on the distance to surrounding geometry can be problematic in scenes containing significant indoor parts, as the user is close to geometry (e.g., walls and floors) most of the time in these parts [10]. Therefore, as an alternative, *viewpoint quality* has been suggested as a criterion for travel speed adjustment [4].

The viewpoint quality for a position in a virtual scene is a scalar value describing its informativeness [4]. It can be calculated in different ways, based on which parts of the scene can be seen from a point, using Viewpoint Quality Estimation (VQE) algorithms (such as viewpoint entropy [11], object uniqueness [4], etc.). In most of these measures, viewpoint quality is low for positions in empty corridors or halls, as not much information about the scene can be gathered from there, while it is higher in furnished rooms or close to detailed objects.

In this work, we have developed an approach to automatically adjust the travel speed based on viewpoint quality. Furthermore, we have conducted a user study to compare the efficiency, accuracy and effects on cognitive load of an interface equipped with this technique against a similar interface with manual speed selection.

The rest of the article is structured as follows. In section 2, we describe our method. In section 3, we present the user study we conducted and discuss its results. Section 4 then discusses some limitations of our method, followed by a conclusion in section 5.

2 Method

Successful VQE algorithms compute a higher value for positions that provide more information about the scene and lower values for points from which little relevant information can be seen. Our approach is based on the observation that viewpoints with higher quality often correspond to areas where the preferred movement speed is lower and vice versa. For example, in highly detailed rooms with high viewpoint quality, there are usually more interesting aspects to see and collisions to avoid, requiring low speeds. In contrast, in empty corridors or large halls with low viewpoint quality, collision avoidance is easy and few interesting details prompt users to stay at one place or move slowly. Furthermore, viewpoint quality usually decreases with increasing distance to interesting scene content.

Our approach is not based on a specific VQE algorithm to allow for the viewpoint quality to be computed by any method suitable for a given scene or application. However, we assume that the minimum viewpoint quality of the used VQE algorithm is 0, and that values close to 0 are also realistically achieved for low-quality viewpoints in the scene (if there are any). Note that if this property is not fulfilled by an algorithm, a simple transformation (e.g., $value \leftarrow 2^{value} - 1$ for many entropy-based algorithms) can often establish it.

As most VQE algorithms produce *global* values (i.e., the value depends on the visibility of the whole scene), the absolute value can usually not be interpreted directly. For example, when a scene is larger, the viewpoint quality value at most points is lower, as a larger part of the scene is invisible from most positions. Therefore, we compute the normalized viewpoint quality $\bar{q}(p) \in [0,1]$ from the viewpoint quality $q(p)$ at each position p by dividing it by the 95% quantile value $q_{0.95}$, clamping larger values to 1: $\bar{q}(p) = \min(q(p)/q_{0.95}, 1)$. Note that similar values, such as the 90% quantile, usually result in a similar normalization and can be used equally well, although we do not use the maximum value to avoid an influence of extreme outliers. Note that this normalization requires a precalculation of the viewpoint quality in the scene. However, as viewpoint quality usually varies only slightly for similar positions (cf. [4]), a very coarse sampling of the scene is sufficient for a good approximation.

At its core, our approach computes the travel speed v at position p as the reciprocal of the normalized viewpoint quality $\bar{q}(p)$:

$$v(p) = \min\left(\frac{v_{min}}{\bar{q}(p)}, v_{max}\right),$$

where v_{min} is the minimum speed to be reached at positions with highest viewpoint quality, and v_{max} is an upper limit for the speed even at very low-quality positions (if there are any). This means that the maximum speed is not necessarily reached anywhere in the scene if there are no positions with low viewpoint quality. For example, small scenes that are highly detailed everywhere usually show similar viewpoint quality at most positions, but also do not require large differences in travel speed.

Furthermore, to avoid frequent changes in speed during travel, we apply a Gaussian smoothing to the viewpoint quality. For each position, the smoothed value is computed by integrating over the viewpoint quality values of the surrounding positions at the same height, weighted according to a 2D horizontal Gaussian distribution (as we expect mostly horizontal travel). To avoid taking, e.g., adjacent rooms into account when computing the Gaussian, we perform intersection tests on the scene geometry and only regard points that are not occluded.

For static scenes, both viewpoint quality and travel speeds can be precomputed. In dynamic scenes, a similar smoothing can be achieved with less computational effort by limiting the maximum speed change based on the distance traveled, such that the VQE algorithm is the only possibly computationally expensive step.

Similar to [10], it is possible to additionally consider the user's travel direction. We examined incorporating the viewpoint quality at some distance in travel direction, calculating the direction-dependent viewpoint quality in travel direction, and combinations of these with the direction-independent viewpoint quality. However, we found that the influence of the direction was either too small or caused the resulting speed to vary too strongly when changing directions. Therefore, for the user study, we did not consider the travel direction.

3 User Study

We conducted a user study to test the usability, efficiency, and cognitive load of our approach, comparing viewpoint-quality-based travel speed adjustment against manual speed selection. The study took place in a 5-sided CAVE equipped with 60 Hz optical tracking and loudspeakers, using an ART Flystick2 to control travel.

3.1 Scene

As test scene, we acquired the office scene used in [4] (see Figure 1), as it contains areas where precise travel is necessary (small, furnished rooms), as well as regions where using higher travel speeds is beneficial (empty corridors). However, we slightly broadened the corridors, as we suspected that narrow pathways during virtual travel might contribute to simulator sickness.

3.2 Method details

We compared two travel techniques, our automated approach (**A**) and a method using manual speed selection (**M**). In both techniques, the travel direction was determined by the pointing direction of the Flystick (restricted to horizontal travel), while the index finger button indicated whether the user wanted to travel. Additionally, in **M**, the desired travel speed could be changed by pressing one of the

Figure 1: Top view of the scene used in the user study.

four thumb buttons on the Flystick. For the speed range, we chose $v_{min} = 1.5$ m/s (walking), and $v_{max} = 8$ m/s (fast run), which we also used for the slowest and fastest settings in **M**. For the intermediate speed settings in **M**, we interpolated linearly, resulting in speeds of 3.67 m/s and 5.83 m/s. In both techniques, an acceleration of 16 m/s^2 was used when changing the speed, while travel always stopped instantly when releasing the index finger button. Note that, although there are many possibilities to realize manual speed selection, this particular implementation was chosen to keep the interfaces for both techniques as similar as possible. Furthermore, it would be possible (and likely useful) to provide the user with additional control over the automatically selected speed in **A**. Moreover, different relationships between viewpoint quality and speed, such as polynomial, power or exponential functions (or combinations thereof) could be used. However, to avoid tuning the method to a specific scene and to observe and evaluate the effect of only the automatic adjustment more directly, we chose the simplest approach for the study.

To estimate viewpoint quality, we used the *object uniqueness* algorithm, implemented as described in the original source [4], as it was shown to produce good results on indoor scenes. The algorithm partitions the scene into objects, and assigns each object i a uniqueness value $U(i) \in (0,1]$ that indicates how visually distinctive it is compared to other objects in the scene. From this, the viewpoint quality is calculated in a way that it is higher when the apparent (projected) size of all objects is closer to being proportional to their uniqueness. In the result, the viewpoint quality is higher for points that are closer to more unique objects, and farther away from less unique objects. For bad viewpoints, it approaches 0, fulfilling the requirement to be used with our method (see section 2). For the Gaussian smoothing of the scores, we used $\sigma = 2$ m. As rendering the scene ($\approx$3.5M triangles) occupied most of the resources of our system, we precomputed both viewpoint quality and travel speeds using a 25 cm regular spacing throughout the scene (visualized in Figure 2) to ensure a frame rate of at least 60 Hz during the study.

3.3 Procedure

The study task consisted of finding a series of numbers displayed on screens or in boxes, for which participants had to navigate to different rooms in the office. To avoid searching and ensure that all participants traveled approximately the same path, the way to the target room was indicated by virtual arrows. Once a number was found, participants spoke it out loud, whereupon the path to the next room was displayed. In addition, as travel is usually only a supporting task in real applications, participants had to perform an additional task simultaneously. For this, audio recordings of numbers between 1 and 50 were played back to the participant once every 3 seconds. Whenever a number they heard was a multiple of the last number they

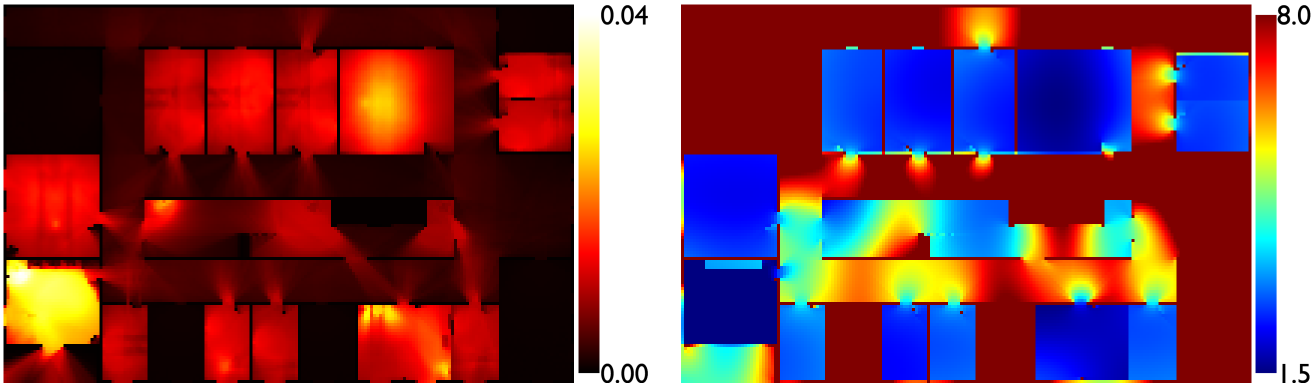


Figure 2: **Left:** Viewpoint quality $q(p)$ for a horizontal slice of the scene (1.60 m above the ground) computed using the *object uniqueness* algorithm. **Right:** Travel speeds $v(p)$ (in m/s) resulting from our approach. In both images, a pixel corresponds to an area of 25 cm$\times$25 cm in the scene.

found, participants had to press a button on a presenter remote carried in their other hand. The numbers were chosen randomly in a way that the probability of the participant having to react was always 1/3.

In total, each participant completed four trials, alternating between techniques. The two different orders **MAMA** and **AMAM** were counter-balanced between participants. Each trial consisted of a route created by concatenating the same 12 path segments in one of 8 different ways. Out of the 8 possible routes along the same path resulting from this, one was chosen at random each time to reduce learning effects. The route started as soon as the participant had traveled to the first room and read the first number there. Before each of the first two trials, there was a training session in a small, artificial office scene where participants could practice the corresponding travel technique until they felt confident using it. After each trial, they could take an optional break and leave the CAVE.

Participants were told to complete each trial as fast as possible, but still travel precisely to avoid unnecessary corrections and collisions with the scene. Furthermore, they should avoid any mistakes in the additional task. Participants were told not to walk physically, although rotations and small movements were permitted.

Before the first trial, participants gave their informed consent, filled out a demographic questionnaire as well as Kennedy's SSQ [5], and were informed about the study procedure in detail. After the fourth trial, they filled out the SSQ again, as well as a questionnaire containing a series of statements about both techniques (e.g., regarding their subjective efficiency, precision, and ease of use) on a 5-point Likert scale. In total, the complete study procedure took 47.9 minutes (SD=9.7 min.) on average.

3.4 Participants

In total, 40 people (mean age 26.1, SD=4.4, 5 female) participated in the experiment. 16 of them were VR professionals, 14 further participants had used a CAVE at least once in an earlier study or demo, 10 had never used a CAVE before. All were unpaid volunteers. Three participants (all first-time VR users) had to abort the experiment due to simulator sickness and thus provided no or only partial data. Moreover, one log file was corrupted and therefore removed. Four further participants did not fully understand the task at first, so that their data from the first trial was not used. Whenever data from a trial was removed, we removed the corresponding trial from the other technique as well to preserve balancing. In total, data from 140 trials (70 per technique) was retained, while counter-balancing of the order of conditions was preserved. Furthermore, questionnaires from 37 participants were used in the analysis.

	Travel technique						
	M: mean (SD)		**A**: mean (SD)		$F_{1,132}$	p
total time [s]	199.3	(59.4)	179.7	(46.2)	4.37	**.039**
distance traveled [m]	496.0	(14.0)	501.0	(15.4)	4.73	**.031**
error rate	.116	(.054)	.089	(.055)	8.53	**.004**
reaction time [s]	1.54	(.181)	1.44	(.181)	9.13	**.003**

Table 1: Mean values and significant main effects of the travel technique on performance measures.

3.5 Hypotheses

As the speed does not have to be controlled manually, we expect that using **A** requires less cognitive resources than **M**. Furthermore, we expect that participants speed up earlier with **A**—as soon as an area affords higher speeds—which should make the technique more efficient. In addition, as we expect **A** to reduce the speed at the right time, participants should overshoot their target less often and require fewer corrections. Based on this, we formulate the following hypotheses:

H1: Using **A**, participants make less mistakes, and take less time to react in the additional task.
H2: The time to complete a trial is shorter with **A** than with **M**.
H3: Using **A**, participants travel a shorter distance per trial.

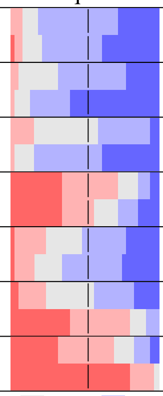
3.6 Results

We conducted a two-way ANOVA to analyze the effects of travel technique and trial number on the total time, distance traveled, and error rate and reaction time in the additional task. The analysis revealed significant main effects of the travel technique, shown in Table 1. Furthermore, we found a significant main effect of the trial number on the total time ($F_{3,132}$=9.86, p<.001, participants took less time in later trials), but on none of the other variables. We found no interaction effects. Moreover, we conducted an independent-samples Mann-Whitney U test to compare the number of head collisions with the scene geometry between techniques and found a significant effect (p=.006), indicating more collisions with **M** (M=.66 per trial) than with **A** (M=.24).

To analyze the questionnaires, we compared the answers to questions about both techniques using Wilcoxon signed rank tests. The results are summarized in Table 2. The average SSQ score was 12.8 (SD=14.7) before, and 38.2 (SD=26.4) after the experiment. Within the subgroups of advanced and inexperienced users (either by VR or video game experience), the tendencies in the values above—both measurements and questionnaire results—are analogous, such that details are omitted here for brevity.

3.7 Discussion

The results show that participants made significantly fewer errors in the additional task using **A**, reacting incorrectly or forgetting to react 23% less often on average. Furthermore, they needed less time ($\approx$0.1 s on average) to react when they had to. These findings confirm hypothesis **H1**, showing that participants had more cognitive



	Answer frequencies	Median	p
I could move precisely	**M**	4	.767
	A	4	
I reached my target fast	**M**	4	**.014**
	A	5	
I could successfully solve the additional task	**M**	3	**.000**
	A	4	
The technique causes discomfort	**M**	2	.143
	A	2	
Using the technique was fun	**M**	4	.427
	A	4	
I had to concentrate to use the technique	**M**	3	**.000**
	A	2	
The technique is complicated to use	**M**	2	**.000**
	A	1	

1 strongly disagree 2 disagree 3 neither 4 agree 5 strongly agree

Table 2: Questionnaire results comparing the travel techniques.

resources to spare when they did not have to control their speed manually. Furthermore, although it is conceivable that an automatically varying speed may cause additional mental load, as users have to adapt to it to steer correctly, this effect seems to be only minor, or was canceled out by the advantages of the automatic adjustment.

Moreover, participants needed significantly less time to complete a trial with **A** than with **M** (19.6 s less on average). This result can be explained in different ways. On the one hand, as argued in hypothesis **H2**, the time advantage may have been caused by **A** increasing the travel speed at the right time. In contrast, with **M**, participants may have switched to the higher speeds with some delay, as they had to control travel direction as well as speed, and also had to solve the additional task at the same time. However, on the other hand, some participants may simply have preferred to move slower in general, either not trusting their abilities to steer precisely using higher speeds, or just accepting that they will not complete the trial as fast as they could in favor of a more comfortable experience. This second explanation is supported by the fact that several participants commented on the higher speeds as being too fast. However, as some participants also noted that they found the speeds selected by **A** to be too slow, we believe that both explanations have a share in accounting for the faster completion time.

This ambivalence emphasizes that techniques for automatic travel speed adjustment should respect user preferences. Therefore, it should either be possible to manually adapt the speed range (or define a user-dependent speed coefficient), or adapt to user preferences automatically. This could be done, for example, by providing the user with a narrow range around the automatically determined speed controlled by a slider or joystick, and adapt it when they consistently travel at the upper or lower end of the range.

The results further show that participants traveled a slightly longer distance ($\approx$1% on average) per trial with **A** than with **M**, which contradicts our hypothesis **H3**. We expected that participants would travel less precisely using **M**—which is supported by the fact that users collided significantly more frequently with the scene using **M**—and therefore also travel a longer distance. However, the reason for the slightly longer distance is probably due to the higher overall speed when using **A**. Participants who could control higher speeds less precisely in general probably traveled slower with **M** to achieve higher accuracy. In total, none of the techniques seems to have been clearly superior regarding precision.

Participants agreed to the statement that they reached their targets fast significantly more often with **A**. Furthermore, they stated that they solved the additional task successfully significantly more often with **A**. Both of this is consistent with the objective measurements. Moreover, participants disagreed with having to concentrate to use **A** and with **A** being complicated to use significantly more often than for **M**, indicating that our approach can be used in practice.

4 Limitations

Although the user study showed the effectiveness and usability of our method, some limitations can be observed. First, we only evaluated it on one scene, which not necessarily generalizes to other environments. The method's success depends on the used VQE algorithm to produce usable viewpoint quality values, which has to be tested on further scenes in the future. Furthermore, there are some aspects of travel not captured by the concept of viewpoint quality. For example, although the quality is only marginally higher in corridor corners, it may help users to slow down there. Moreover, most users probably want to travel slower in empty, but narrow spaces or when approaching walls. This could be improved by combining our method with an approach based on the distance to the environment, which could then be used for close distances only. Lastly, it should be examined whether automatic speed adjustment in general has an influence on spatial perception.

5 Conclusion

We presented an approach for automatic travel speed adjustment in virtual environments based on viewpoint quality. Even though we kept the method intentionally simple to explore the validity of the concept, we could show in a user study that it is easy to use, allowing users to reach targets faster and use less cognitive resources than with a technique using manual speed selection, albeit traveling about 1% farther.

In future work, we also want to successfully include the travel direction into our approach. Furthermore, as it depends heavily on the computed viewpoint quality, we want to evaluate our method on different scenes using different VQE algorithms, and improve existing algorithms to focus on better performance for our use case.

References

[1] F. Argelaguet. Adaptive Navigation for Virtual Environments. In *IEEE Symposium on 3D User Interfaces (3DUI)*, pages 123–126, 2014.

[2] F. Argelaguet and C. Andújar. Automatic Speed Graph Generation for Predefined Camera Paths. In *Proc. of the 10th International Conference on Smart Graphics*, pages 115–126, 2010.

[3] D. Bowman, E. Kruijff, J. LaViola Jr, and I. Poupyrev. *3D User Interfaces: Theory and Practice*. Addison-Wesley, 2004.

[4] S. Freitag, B. Weyers, A. Bönsch, and T. W. Kuhlen. Comparison and Evaluation of Viewpoint Quality Estimation Algorithms for Immersive Virtual Environments. In *ICAT-EGVE 2015*, pages 53–60, 2015.

[5] R. Kennedy, N. Lane, K. Berbaum, and M. Lilienthal. Simulator Sickness Questionnaire: An Enhanced Method for Quantifying Simulator Sickness. *The Int. Journal of Aviation Psychology*, 3(3):203–220, 1993.

[6] R. Kopper, T. Ni, D. Bowman, and M. Pinho. Design and Evaluation of Navigation Techniques for Multiscale Virtual Environments. In *IEEE Virtual Reality Conference (VR)*, pages 175–182, 2006.

[7] J. Mackinlay, S. Card, and G. Robertson. Rapid Controlled Movement through a Virtual 3D Workspace. In *ACM SIGGRAPH Computer Graphics*, volume 24, pages 171–176, 1990.

[8] J. McCrae, I. Mordatch, M. Glueck, and A. Khan. Multiscale 3D Navigation. In *Proc. of the ACM Symposium on Interactive 3D Graphics and Games*, pages 7–14, 2009.

[9] H. Taunay, V. Rodrigues, R. Braga, P. Elias, L. Reis, and A. Raposo. A Spatial Partitioning Heuristic for Automatic Adjustment of the 3D Navigation Speed in Multiscale Virtual Environments. In *IEEE Symposium on 3D User Interfaces (3DUI)*, pages 51–58, 2015.

[10] D. Trindade and A. Raposo. Improving 3D Navigation in Multiscale Environments using Cubemap-Based Techniques. In *Proc. of the ACM Symposium on Applied Computing*, pages 1215–1221, 2011.

[11] P.-P. Vázquez, M. Feixas, M. Sbert, and W. Heidrich. Automatic View Selection Using Viewpoint Entropy and its Application to Image-Based Modelling. In *Computer Graphics Forum*, volume 22, pages 689–700, 2003.

[12] C. Ware and D. Fleet. Context Sensitive Flying Interface. In *Proc. of the 1997 Symposium on Interactive 3D Graphics*, pages 127–ff., 1997.

[13] C. Ware and L. Slipp. Using Velocity Control to Navigate 3D Graphical Environments: A Comparison of Three Interfaces. In *Proc. of the Human Factors and Ergonomics Society Annual Meeting*, volume 35, pages 300–304, 1991.