Open access • Proceedings Article • DOI:10.1109/3DUI.2014.7027325

Adaptive navigation for virtual environments — Source link

Ferran Argelaguet

Institutions: French Institute for Research in Computer Science and Automation

Published on: 29 Mar 2014 - Symposium on 3D User Interfaces

Topics: Mobile robot navigation, Turn-by-turn navigation, Jerkiness, Simulator sickness and User experience design

Adaptive Navigation for Virtual Environments

Ferran Argelaguet Sanz

Adaptive Navigation for Virtual Environments

Ferran Argelaguet[*]
Inria Rennes

Abstract

Navigation speed for most navigation interfaces is still determined by rate-control devices (e.g. joystick). The interface designer is in charge of adjusting the range of optimal speeds according to the scale of the environment and the desired user experience. However, this approach is not valid for complex environments (e.g. multi-scale environments). Optimal speeds might vary for each section of the environment, leading to non-desired side effects such as collisions or simulator sickness. Thereby, we propose a speed-adaptation algorithm based on the spatial relationship between the user and the environment and the user's perception of motion. The computed information is used to adjust the navigation speed in order to provide an optimal navigation speed and avoid collisions. Two main benefits of our approach is firstly, the ability to adapt the navigation speed in multi-scale environments and secondly, the capacity to provide a smooth navigation experience by decreasing the jerkiness of described trajectories. The evaluation showed that our approach provides comparable performance as existing navigation techniques but it significantly decreases the jerkiness of described trajectories.

Index Terms: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Virtual Reality H.5.2 [Information Interfaces and Presentation]: User Interfaces—Input devices and strategies, Interaction styles, User-centered design

1 Introduction

Navigation is a key task for any virtual reality application. The user should be able to modify their viewpoint in order to explore, search or maneuver in the virtual environment [6]. However, although a wide range of navigation interfaces exist (e.g. locomotion interfaces, flying interfaces), few works have addressed how the navigation speed can be adjusted in order to ensure a smooth and pleasant navigation experience. In rate-control interfaces, the range of the input device is mapped into a fixed range of acceptable navigation speeds. This range might not be optimal for a wide range of virtual environments, resulting on motions that are potentially too fast (Which can induce motion sickness [11]) or too slow (decreasing user engagement).

In this work, we have explored how navigation speed can be automatically adjusted in order to provide an optimal navigation speed. Our approach monitors how navigation speed is perceived by the user (optical flow) and the relationship between the user and the virtual environment (time to collision). Using this information, we adjust the navigation speed according to the user and scene demands. The proposed approach can be easily integrated in any navigation interface and based on the results obtained from the conducted evaluation, it can (1) conveniently adapt the navigation speed according to the virtual content and (2) simplify the user control enabling experts and amateurs alike to efficiently navigate in virtual environments.

[*]e-mail: fernando.argelaguet_sanz@inria.fr

2 Previous Work

2.1 Speed Control

The speed control of a navigation technique is linked with the scale of the environment and the user's preferences. While the scale of the environment provides a threshold for the maximum allowed speed, users, through the navigation interface, are able to adjust the speed using a number of input commands [4] and speed mappings [1]. In terms of speed, we can distinguish the linear component of the motion (tangential speed) and the rotational component (rotational speed). If no speed adaptation is available, and if the scale of the environment is known *a priori*, then the maximum speed can be properly adjusted by the interface designer. In addition, if we are navigating through a real scale model, the tangential speed can be bounded to human walking speed ($0.6m/s$ to $1.4m/s$) [7]. Moreover, to better resemble the human walking speed, when rotational speed increases, tangential speed should be decreased [7].

However, due to the wide range of heterogeneity of virtual environments and differences in user preferences, only few papers have addressed the adaptation of navigation speed. The first approach considering dynamic speed adaptation was the "Context Sensitive Flying Interface" from Ware and Fleet [12], who proposed the usage of depth information in order to adjust the navigation speed. In their implementation, the best results were achieved when the movement speed was coupled with the minimum value of the depth map. An improved version of Ware and Fleet interface is the Cubemap navigation proposed by McCrae *et al.* [8]. The Cubemap approach considers the entire surroundings of the user by computing a depth cubemap from the camera position. This information is used to adjust the speed (following a similar approach as [12]) and avoid collisions. Following a different perspective, in previous work Argelaguet *et al.* [2], we proposed a perceptual-based approach based on visual attention models. Our approach was based on optimizing the navigation speed at each point of a predefined camera path according to the desired perceived motion. No interactivity was provided.

2.2 Optical Flow and Perceived Motion

Optical flow can be defined as the velocities of 3D surface points projected onto the imaging plane. Studies on optical flow have shown that it plays a key role in how camera motion is perceived by the audience (e.g. locomotion control [9] and motion sickness [11]).

Optical flow analysis has been explored for the estimation of the perceived speed when navigating in virtual environments: flight simulators [5], animations in outdoor environments [10] and predefined camera paths along arbitrary scenes [2]. However, none of these approaches are able to provide a real-time solution which can handle arbitrary virtual environments. While the work of Konasinski [5] focused mainly on the optical flow accounted by ground motion (flight simulators), the work from So *et al.* [10] considered that the environment always has a similar complexity. In contrast, although the approach proposed by [2] handles arbitrary environments, it is not suited for real-time navigation due to its complexity and the fact that it operates within predefined camera paths.

3 Adaptive Navigation

Leveraging in current visual attention models [2], our approach proposes a real-time adaptation of the navigation speed. While the previous approache [2] relied on predefined paths, we account for interactive navigations in which the user has total control of the camera movement and can also influence the camera speed. The following subsections describe the two components of the proposed approach: the quantification of the perceived speed and the adjustment of the navigation speed in order to ensure an optimal speed.

3.1 Estimation of the User's Perceived Speed

The estimation of the perceived speed is computed from (1) the state of the system (the relative distance between the 3D environment and the user, and the current navigation speed) and (2) the quantification of the optical flow between the current and the previous frame. The heuristic employed operates in image space and only takes into account the visible part of the scene. All the computations are done using vertex and fragment shaders and stored in additional render targets (256 by 256 pixels).

Time to Collision (TTC) The TTC map is meant to gather information about the relative distance between the user and the virtual environment. For each pixel, we compute the relative speed and estimate the time when a potential collision might occur. The computation only takes into account the minimum distance between each pixel and the viewing plane (z axis). Algorithm 1 describes the pseudo-code of the fragment shader used to compute the TTC map. Similar to [2], the time to collision is modulated through an exponential decay.

Algorithm 1 Time to Collision map computation.

```
Require: $P_t$: 3D coordinates for each pixel (Observer's referenced)
Require: $P_{t-1}$: 3D coordinates for each pixel in the last frame
Require: $\Delta t$: Time between frames
Require: $d$: Weighting factor. We used $2.5s$ for the experiments.
  for all pixels do
    $speed = \frac{|(||P_{t,i}|| - ||P_{t-1,i}||)|}{t}$
    $ttc = P_{t,i}(z)/speed$
    $TTC_i = e^{(-ttc^2/2d^2)}$
  end for
  return TTC
```

Optical Flow (OF) The OF map captures the perceived motion of the scene from the user's point of view. For each pixel, we compute the amount of displacement (in screen space) between two consecutive frames. Algorithm 2 describes the pseudo-code of the fragment shader used to compute the OF map. The final result is also modulated as proposed by [2].

Algorithm 2 Optical Flow map computation.

```
Require: $P_t$: 2D normalized coordinates for each pixel
Require: $P_{t-1}$: 2D normalized coordinates for the previous frame
Require: $\Delta t$: Time between frames
Require: $pixelSize$: Screen pixel size in $mm$ ($0.27mm$)
Require: $mapSize$: Map size in pixels ($256px, 256px$)
Require: $m$: Weighting factor in $mm/s$ ($100mm/s$)
  for all pixels do
    $distance_i = |(P_{t,i} - P_{t-1,i})| \cdot mapSize$
    $speed_i = ||distance|| \cdot pixelSize/time$
    $OF_i = 1 - e^{(-speed_i^2/2m^2)}$
  end for
  return OF
```

Once both maps are computed, they are combined through a weighted sum (see Equation 1), although we explored different alternatives to combine the information, the weighted sum provided the best results in terms of stability and convergence. The combination of the TTC map and the inverse of the OF map accounts for objects close to the focus-of-expansion (FoE) which have a small time to collision and the optical flow accounts for the perceived camera motion.

$$h = \sum_{i=0}^{n} \frac{(TTC_i * (1 - OF_i) + OF_i}{n} \quad (1)$$

The h estimator provides the perceived user speed (no units) according to the current speed and the environment.

3.2 Speed adaptation

The speed adaptation is based on updating the current acceleration of the camera to ensure that optimal perceived speed is achieved without abrupt speed changes. The speed adaptation algorithm (see Algorithm 3) first computes the optimal speed according to the ratio (h_{opt}/h) between the optimal perceived speed (h_{opt}) and the current perceived speed (h). Then it estimates the "acceptable" acceleration to reach the desired speed. The estimation depends on the (h_{opt}/h) ratio, if the ratio is greater than 1 (too fast) the estimation ensures that a fast deceleration will ensure a fast convergence of h into h_{opt}. On the contrary, if the ratio is smaller than 1 (too slow) a sigmoid function is used in order to rapidly increase the speed, but limiting the increase above a certain threshold.

Algorithm 3 Tangential speed update.

```
Require: $s$: Current tangential speed
Require: $a$: Current acceleration
Require: $\Delta t$: Frame time
Require: $h_{opt}$: Target threshold for $h$
  $h$ = computePercievedOpticalFlow()
  $s_{opt} = s \cdot (h_{opt}/h)$
  if $h > h_{opt}$ then
    $\Delta h_{max} = -1200 + 20 \cdot h$
  else
    $\Delta h_{max} = 7 + 13 \cdot e^{-h^2/(2*5^2)}$
  end if
  $t_{req} = |h_{opt} - h|/\Delta h_{max}$
  $a_{opt} = (s_{opt} - Speed)/t_{req}$
  $a = a + 0.05 \cdot (a_{opt} - a)$
  $s = s + a * \Delta t$
```

However, the approach failed to provide smooth navigations for strong camera rotations. The contribution of camera rotation to the optical flow is mainly dependent on the rotation speed. Camera rotations dramatically increase the optical flow, resulting in excessive slowdowns when turning. We decided to remove the optical flow accounted for camera rotations from the computation of the *OF* map and provide a fixed mapping to decrease the tangential speed when turning. We applied the approach detailed by [7], which limits the tangential speed according to the rotational speed.

In our implementation, due to practical reasons, we used a joystick as the input device. Forward and backward motion was determined by the x-axis of the joystick while yaw rotation was controlled through the y-axis. Algorithm 4 describes how h is modulated according to the user input, where two scale factors are computed according to the tangential (f_t) and rotational components (f_r). The last step was the computation of the optimal perceived speed (h_{opt}). In several controlled environments, using the navigation technique proposed by [7], we computed h_{opt} analytically. Several runs were performed and the mean h values computed were averaged to compute h_{opt}. Our analysis determined that $h_{opt} = 65$.

Algorithm 4 Update of the h according to the user input

```
Require: $joy$: 2DoF Joystick data [-1..1]
  $\omega = joy(y) \cdot 1^{rad}/_s$
  $f_t = ||joy(x)||$
  $f_r = e^{-||\omega||/2*0.7}$
  $h = {}^h/_{f_t * f_r}$
  return $h$
```

3.3 Collision Avoidance

Collision avoidance is a key element to ensure that the user is able to naturally navigate in the virtual environment while avoiding obstacles. In order to provide a collision avoidance mechanism, we followed a similar approach as McCrae *et al.* [8]. The information encoded in a depth map is used to determine potential collisions and avoid them. Depth information is used to compute a repulsion force altering the user trajectory and avoiding potential collisions. The algorithm is dependent on the collision boundary δ and a softness parameter σ. In our implementation the collision boundary was computed according to the current tangential speed: $\delta = s/2$, thus adapting the boundary to the speed and also to the level of scale. The softness parameter was also dependent on the the current tangential speed (see Equation 2).

$$w(d_i) = e^{\frac{(\delta - d_i)^2}{(0.25 \cdot \delta)^2}} \tag{2}$$

$w(d_i)$ is computed for each pixel and d_i represents the distance towards the observer. The repulsion force is computed for each pixel, averaged and applied to the direction of movement at each frame. If the repulsion force is greater than a certain threshold, we consider that the collision cannot be avoided and the navigation stops.

4 User Evaluation

We conducted a formal evaluation in order to explore the benefits and caveats of our approach compared to state of the art approaches. We evaluated our technique against a fixed control law described in [7] and an adaptive approach based only on depth information [8]. The aim of the experiment was to explore if the evaluated techniques allowed participants to navigate in a multi-scale environment and the jerk analysis of the described trajectories.

Procedure. The task was to traverse a virtual environment as fast as possible but limiting the number of collisions. The synthetic virtual environment build for this experiment considered two different scene configurations, first, a homogeneous zone shaped as a maze-like environment and non-homogeneous zone populated with geometrical objects (see Figure 1). Users could to control the forward-backward movement through the X-axis of the joystick and the yaw rotation was controlled through the Y-axis. For each control law, a short training session with no time limit was provided.

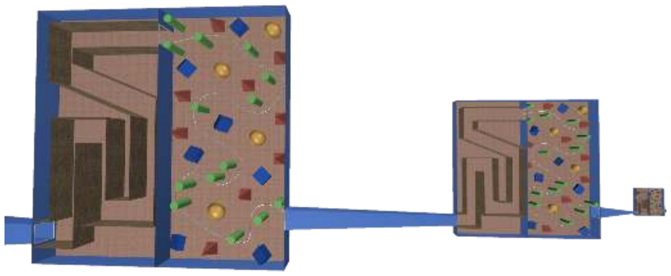
Figure 1: Virtual environment used for the navigation task. The environment had two different sections in order to explore the behavior of evaluated techniques in different scene configurations. Furthermore, three different levels of scale were used (1:1, 1:2 and 1:10).

Apparatus. The experiments were conducted with a 24″ wide screen monitor with a resolution of 1920 × 1200px, with Logitech Wingman Extreme 3 Joystick as solely input device. The distance between the user and the monitor was approximately 50cm.

Participants. Fifteen participants (thirteen male, two female) aged from 21 to 39 ($M = 28.2; SD = 4.95$), participated in the experiment.

4.1 Design and hypotheses

We used a factorial design in which the independent variables were (1) the speed control (Fixed, Depth, OpticalFlow), (2) the scale of the environment (Big, Medium, Small) and (3) the section of the virtual environment (Maze, Geometry). All variables were within-subjects. We adjusted all three navigation techniques in order to ensure that all of them provided a similar behavior for the first level of scale (*Big*). In order to avoid learning effects, the order of the techniques was counterbalanced using a Latin square design. For the section and the scale factor, the order was fixed due to the design of the virtual environment (see Figure 1).

For each area (two environments times three levels of scale) we recorded the task completion time, the mean value of speed, and jerk [3], and the mean value for the x-axis of the joystick (normalized values [-1..1]). At the end of each run, users were asked to provide their subjective impressions about the difficulty of the task (Control), the perceived speed, the smoothness of the trajectory in terms of speed (Jerk), and their preferences. Users had to rank each question using a 7-Likert scale.

Our hypotheses were: **[H1]** Higher task completion time as the level of scale decreases for the *Fixed* technique. **[H2]** The level of scale will not influence task completion time for the *Depth* and *OpticalFlow* techniques. **[H3]** Different mean values of speed for each *Section*. **[H4]** Different mean values of jerk for each *Section*. **[H5]** Lower mean values of jerk for the *OpticalFlow* technique. **[H6]** Lower subjective ratings for the *Fixed* technique.

4.2 Analysis

The statistical analysis is based in ANOVA and Bonferroni pairwise tests for post-hoc analysis. Only pairwise comparisons with a significant difference of $p < 0.05$ are discussed.

4.2.1 Task Completion Time

To analyze the time required to traverse the environment, we do not consider each section independently (Maze, Geometry) but the time to traverse each level of scale. The two-way ANOVA *Technique*, *Scale* versus *Time* showed a strong interaction effect ($F_{4,116} = 35.09; p < 0.001; \eta_p^2 = 0.54$), main effects did not provide conclusive results. Post-hoc tests provided three main results. First, all pairwise tests did not show any significant differences for the *Big* level of scale between techniques. This result shows that the fine tuning of all techniques ensured a similar performance for the optimal level of scale. Second, pairwise tests showed significant differences for each level of scale for the *Fixed* technique (Fixed/Big: $M = 157.8s$, Fixed/Medium: $M = 141.17s$, Fixed/Small: $M = 192.05s$). Although we reject the null hypothesis, we observe that the task completion time does not increase monotonically. Participants completed the task faster for the *Medium* level of scale as they were still able to maneuver. However, for the *Small* level of scale the speed was excessive resulting in a lot of collisions with the environment, thus we cannot support **[H1]**. Third, pairwise tests did not show significant differences as the level of scale decreases neither for *Depth* nor for *OpticalFlow*. This result shows that the behavior of both techniques was not dependent on the level of scale, we cannot reject the null hypothesis but this result supports **[H2]**.

4.2.2 Average Speed

Due to the results obtained for the task completion time, we did not consider the data from the *Fixed* technique for follow up analysis. The two-way ANOVA of *Technique* and *Section* versus *Speed* showed a main effect on *Technique* ($F_{1,14} = 29.96; p < 0.001; \eta^2 = 0.22$) and *Environment* ($F_{1,14} = 78.75; p < 0.001; \eta^2 = 0.22$). Post-hoc tests showed that the mean speed values were higher for the *Maze* section ($M = 1.09m/s; SD = 5.84m/s$) than the *Geometry* section ($M = 1.01m/s; SD = 8.0m/s$), rejecting the null hypothesis and supporting **[H3]**. Post-hoc tests also showed that the mean speed was higher for the *Depth* ($M = 1.09m/s; SD = 7.6m/s$) than for the *OpticalFlow* technique ($M = 1.02m/s; SD = 6.27m/s$).

4.2.3 Average Jerk

The two-way ANOVA of *Technique* and *Section* versus *Jerk* showed a main effect on *Technique* ($F_{1,14} = 128.91; p < 0.001; \eta^2 = 0.34$), *Scale* ($F_{1,14} = 722.20; p < 0.001; \eta^2 = 0.39$). Post-hoc tests showed that the mean jerk was lower for the *OpticalFlow* technique ($M = 44.5m/s^3; SD = 10.101m/s^3$) than for the *Depth* technique ($M = 76.6m/s^3; SD = 29.9m/s^3$), rejecting the null hypothesis and supporting **[H5]**. Post-hoc tests also showed that the environment played a significant role (*Maze*: $M = 76.6m/s^3; SD = 29.9m/s^3$, *Geometry*: $M = 76.6m/s^3; SD = 29.9m/s^3$), rejecting the null hypothesis and supporting **[H4]**. This result shows that the proposed technique reduces the amount of jerk compared to state of the art approaches. However, the environment still plays an important role.

4.2.4 User Behavior

The value of the x-axis of the joystick provides information regarding the effort required for the user to adapt the navigation speed. The two-way ANOVA of *Technique* and *Section* versus the mean value of the x-axis showed a strong interaction effect ($F_{1,28} = 902.68; p < 0.001; \eta^2 = 0.32$). Pairwise tests did not find significant differences for the *Depth* and *OpticalFlow* techniques (for all conditions $M \sim 0.98$). In contrast, for the *Fixed* approach significant differences appeared for all levels of *Size* (Big $M = 0.96$, Medium $M = 0.61$ and Small $M = 0.1$). These results show how participants had to adjust the speed manually.

An interesting finding about usability was the fact that by observing participants and their comments during the experiment we found that they felt more comfortable when they used the joystick as a binary input (full forward, full stop). The requirement of constantly adjusting the speed for the *Fixed* technique during the *Medium* and *Small* levels of scale was sometimes disturbing, especially due to the inability to efficiently control the navigation speed. Although this effect is strongly dependent on the input device used and the level of user expertise, it shows that for controlling forward speed, a binary/discreet input might be more efficient.

4.2.5 Subjective Ratings

Friedman rank tests showed that the ranking provided by participants was significantly different for Control ($\chi^2(2) = 13.43; p < 0.001$), Speed ($\chi^2(2) = 13.63; p < 0.001$), Jerk ($\chi^2(2) = 8.10; p < 0.05$) and Preference ($\chi^2(2) = 8,13; p < 0.05$). Wilcoxon tests were used to follow up the analysis with Bonferroni adjustment ($\alpha < 0.0167$). No significant differences were found between *Depth* and *OpticalFlow* techniques. On the contrary, the *Fixed* technique was the worst rated in terms of Control and Speed.

Subjective ratings were consistent with the information gathered during informal discussions. However, participants perceived differences between both adaptive techniques although they were not able to describe them. In addition, some users stated that for the *OpticalFlow* and *Depth* techniques, their feeling was that the camera movement was too slow, while others perceived it as reasonable.

5 Conclusions

In this paper we have proposed and evaluated a speed adaptation algorithm which automatically adjusts the navigation speed based on the spatial relationship between the user and the environment, and the optical flow. In comparison with existing approaches, we provide a perceptual approach which not only considers the virtual environment but also considers how the user perceives the camera motion. The evaluation of the proposed approach has shown two main results. First, it is able to provide comparable navigation performances as state of the art navigation approaches. Second, the speed adaptation approach reduces the jerkiness of the described trajectories. Decreasing the jerkiness of the trajectory ensures smoother camera motions and the described trajectories will better resemble real locomotion trajectories [3].

However, we believe that real strength of our approach would be delivered in immersive virtual reality systems. Although not discussed in the paper, the computation of the optical flow is dependent on the field of view of the virtual camera. The amount of optical flow is highly dependent on the field of view, as we move away from the focus of expansion, the amount of optical flow increases. This is a key feature of our approach which is not handled by existing adaptive approaches. As future work, we plan on analyzing the behavior of the proposed approach in immersive systems (e.g. CAVE-like systems and HMDs). We will explore the role of the field of view on motion perception and analyze potential relationships between the perceived speed and simulator sickness.

References

[1] C. Anthes, P. Heinzlreiter, G. Kurka, and J. Volkert. Navigation models for a flexible, multi-mode VR navigation framework. In *Proceedings of the ACM SIGGRAPH International Conference on Virtual Reality Continuum and its Applications in Industry - VRCAI '04*, pages 476–479, 2004.

[2] F. Argelaguet and C. Andujar. Automatic speed graph generation for predefined camera paths. In *In Proceedings of the 10th International Symposium on Smart Graphics*, SG'10, pages 115–126, 2010.

[3] G. Cirio, A.-H. Olivier, M. Marchal, and J. Pettré. Kinematic evaluation of virtual walking trajectories. *IEEE transactions on visualization and computer graphics*, 19(4):671–680, 2013.

[4] D. H. Jeong, C. G. Song, R. Chang, and L. Hodges. User experimentation: an evaluation of velocity control techniques in immersive virtual environments. *Virtual Reality*, 13(1):41–50, 2008.

[5] E. Kolasinski. Simulator Sickness in Virtual Environments. Technical report, US Army Research Institute for the Behavioral and Social Sciences, 1995.

[6] A. Kulik. Building on Realism and Magic for Designing 3D Interaction Techniques. *IEEE Computer Graphics and Applications*, 29(6):22–33, 2009.

[7] M. Marchal, J. Pettré, and A. Lécuyer. Joyman: A human-scale joystick for navigating in virtual worlds. In *IEEE Symposium on 3D User Interfaces*, pages 19–26, 2011.

[8] J. McCrae, I. Mordatch, M. Glueck, and A. Khan. Multiscale 3D navigation. In *Proc. of the 2009 symposium on Interactive 3D graphics and games*, pages 7–14, 2009.

[9] T. Prokop, M. Schubert, and W. Berger. Visual influence on human locomotion. Modulation to changes in optic flow. *Experimental brain research*, 114(1):63–70, 1997.

[10] R. H. Y. So, A. T. K. Ho, and W. T. Lo. A Metric to Quantify Virtual Scene Movement for the Study of Cybersickness: Definition, Implementation, and Verification. *Presence: Teleoperators and Virtual Environments*, 10(2):193–215, 2001.

[11] R. H. Y. So, W. T. Lo, and A. T. K. Ho. Effects of Navigation Speed on Motion Sickness Caused by an Immersive Virtual Environment. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 43(3):452–461, 2001.

[12] C. Ware and D. Fleet. Context sensitive flying interface. In *Proceedings of the 1997 symposium on Interactive 3D graphics - SI3D '97*, pages 127–ff., 1997.