Representation-Independent In-Place Magnification with Sigma Lenses

Emmanuel Pietriga, Olivier Bau, Caroline Appert

Representation-Independent In-Place Magnification with Sigma Lenses

Emmanuel Pietriga, Olivier Bau, and Caroline Appert

Abstract—Focus+context interaction techniques based on the metaphor of lenses are used to navigate and interact with objects in large information spaces. They provide in-place magnification of a region of the display without requiring users to zoom into the representation and consequently lose context. In order to avoid occlusion of its immediate surroundings, the magnified region is often integrated in the context using smooth transitions based on spatial distortion. Such lenses have been developed for various types of representations using techniques often tightly coupled with the underlying graphics framework. We describe a representation-independent solution that can be implemented with minimal effort in different graphics frameworks, ranging from 3D graphics to rich multi-scale 2D graphics combining text, bitmaps and vector graphics. Our solution is not limited to spatial distortion and provides a unified model that makes it possible to define new focus+context interaction techniques based on lenses whose transition is defined by a combination of dynamic displacement and compositing functions. We present the results of a series of user evaluations that show that one such new lens, the speed-coupled blending lens, significantly outperforms all others.

Index Terms—Graphical user interfaces, Visualization techniques and methodologies, Interaction techniques, Evaluation/methodology

1 Introduction

Bifocal display techniques complement conventional navigation techniques such as pan & zoom, typically used in 2D multi-scale environments [1], [2], [3], and those based on the *walking* and *flying vehicle* [4] metaphors commonly employed to interactively explore 3D worlds. Beyond the now-ubiquitous map-like overviews introduced thirty years ago [5], a variety of bifocal techniques termed *focus+context* have been designed to further help users navigate in complex visual representations such as large trees [6], [7], graphs [8], and high-resolution bitmap representations [9]. These techniques can also help users interact with objects in vector graphics editors [10] or select small interface widgets [11]. The defining characteristic of these focus+context techniques is that they provide *in-place* magnification of a region (the focus) of the current display (the context), allowing users to get more detailed information about specific elements of the interface without having to zoom in the whole representation.

Interactive in-place magnification lenses have been available on the user's desktop for more than a decade, from simple magnifier lenses used as accessibility tools [12] to fancy screen savers distorting the user's workspace. These early implementations were based on a simple magnification method that consisted in merely duplicating the pixels of the original representation. While implementing such lenses is fairly trivial, implementing lenses that actually render objects in the magnified region with more detail is not. Models and implementations of the latter usually require information about, and some level of control on, the inner structure and elements of the representation, which might actually get modified by the lens. Such models and implementations are thus often tightly coupled with a particular graphics framework or application. From a purely theoretical perspective, however, the same general approach applies to many types of applications: 2D vector graphics editors, geographical information systems, CAD tools or any other 3D application. No matter the representation and underlying graphics framework, the process consists in rendering a subregion F of the current representation C at a larger scale and with more detail, and integrating F into C through a non-linear transformation to achieve a smooth transition between the two.

The Sigma Lens framework [13] extends this general process by defining transitions between focus and context as a combination of dynamic displacement and compositing functions, making it possible to create a variety of lenses that use techniques other than spatial distortion to achieve smooth transitions between focus and context, and whose properties adapt to the users' actions, all in an effort to facilitate interaction. In this paper, we extend and strengthen that framework by describing a general approach to its design and implementation that shows how Sigma Lenses can be representation-independent. The main contribution is a rendering technique based on a unified model that can be integrated with minimal effort in different graphics frameworks, ranging from 3D graphics consisting of complex textured meshes to rich multi-scale 2D graphics combining text, bitmaps and

- *E. Pietriga and O. Bau are with INRIA Saclay – Île-de-France and LRI (Université Paris-Sud & CNRS), 91405 Orsay, France. E-mail: emmanuel.pietriga@inria.fr, bau@lri.fr*
- *C. Appert is with LRI (Université Paris-Sud & CNRS) and INRIA Saclay – Île-de-France, 91405 Orsay, France. E-mail: appert@lri.fr*

Manuscript received Mar 18, 2009; revised Aug 10, 2009.

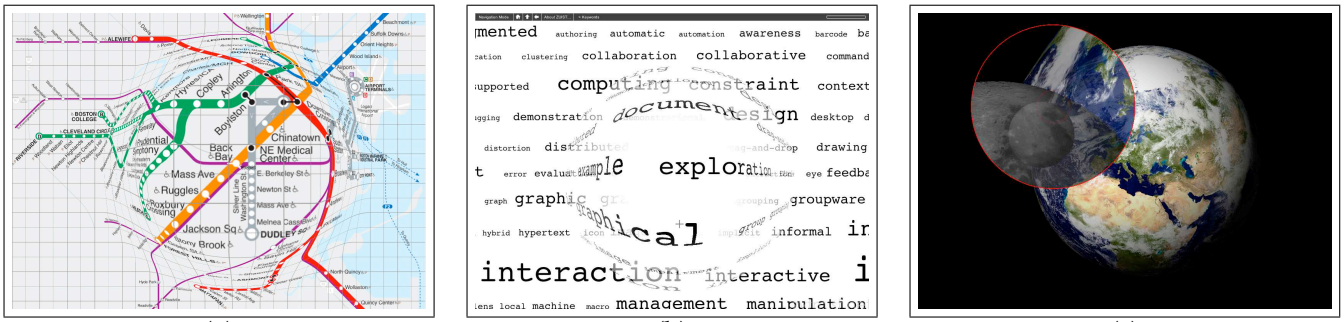


Fig. 1. Three different lenses obtained with minor modifications to the scale and compositing functions: (a) a distortion lens on a high-resolution bitmap (subway map), (b) a hovering lens on 2D vector graphics and text (ACM UIST keyword tag cloud), (c) a speed-coupled blending lens on a 3D model of the Moon orbiting the Earth

vector graphics. The technique does not need to have knowledge about, or access to, the graphical objects that constitute the representation.

After an overview of related work in Section 2, we give a theoretical description of our framework in Section 3, also summarizing the Sigma Lens unified model originally defined in [13]. We then instantiate lenses of particular interest in this model, and present a series of evaluations that test their limit performance on 2D navigations tasks. We provide a detailed summary of the experiments reported in [13], and report the results of a new controlled experiment that covers cognitive aspects of interaction with lenses that were left as future work, further confirming that one of our new lens designs, namely the SPEED-COUPLED BLENDING lens, significantly outperforms all others. A technical overview of two implementations follows, one for OpenGL 3D graphics taking advantage of hardware acceleration, the other for a general-purpose, cross-platform 2D application programming interface. Advantages and limitations of our approach are discussed in Section 8, along with plans for future work.

2 RELATED WORK

Many focus+context techniques are based on the concept of lenses. The simplest lens is the electronic equivalent of a magnifying glass. While easy to understand, this type of lens occludes the immediate surroundings of the magnified region [14], as a physical magnifying glass does, thus hiding an area of significant interest and making it difficult for users to relate focus and context in the representation. In order to avoid this problem, the magnified region is often integrated in the context using smooth transitions based on non-linear magnification techniques. These techniques create a transition zone in which the representation is distorted (see Figure 1-a).

Distortion-based visualization techniques often rely on metaphors inspired by the physical world such as stretchable rubber sheets [15] and, more generally, surface deformations [16]. Others work with more fundamental concepts such as hyperbolic projection [6] or non-linear magnification fields [17]. The distortion can either extend to the limits of the representation [14], [15], or it can be bounded to a specific area. This paper focuses more specifically on the latter case, i.e., on lenses, termed *constrained lenses* [9], [17], [18], [19], that leave a significant part of the context undistorted. Cockburn et al. [20] survey many of these techniques in their recent review of overview+detail, zooming, and focus+context interfaces.

While all the above techniques apply to 2D graphics, other techniques have been developed for distortion in 3D. A first set of techniques deform 3D representations by projecting a texture on a mesh that models the distortion, as do pliable surfaces for 2D representations [16]. LaMar et al.'s magnification lenses [21] are based on homogeneous texture coordinates and special geometries. They can be applied to both 2D and 3D representations but are limited in the type of distortion and lens shapes they can model. Non-linear perspective projections [22] project the RGB image produced by a 3D pipeline on a surface shape inserted in front of the flat projection plane. They can model spatially bounded distortions. Related to the latter is Brosz et al.'s single camera flexible projection framework [23], which is capable of modeling non-linear projections through the parametric representation of the viewing volume.

There is also an impressive set of space deformation techniques (see [24] for an overview), ranging from early works on the deformation of solid primitives [25], [26] to view-dependent geometry [27] and deformation based on hardware-accelerated displacement mapping [28], [29] and deflectors [30]. These techniques distort 3D geometry, but often do so in an object-centric manner, and are thus not well suited to the implementation of focus+context navigation lenses, which deform a region of the current display, i.e., a subsection of the current viewing frustum that intersects a set of objects, some of them only partially. Camera textures [31] are among the few to actually apply constrained magnification lenses to 3D meshes, but the technique requires a sufficient level of tessellation of the target mesh to produce distortions of good quality.

Techniques such as the last one are typical examples of approaches strongly coupled with a specific type of graphics. This coupling is both a strength and a weakness. The strength lies in the capacity to access and modify the graphical objects that are to be distorted and rendered in a focus+context view. This is also a weakness, however, as a technique that works with 2D vector graphics will not be applicable to 3D meshes, or even to representations that include bitmap images. Other techniques, such as those based on 3D surface deformation and texture mapping, suffer from a somewhat opposite problem: while generating focus+context views of 2D representations, they require capabilities that are often not available in 2D graphics libraries.

One goal of our approach is to provide a method for implementing constrained magnification lenses that is as independent as possible of the nature of the representation and graphics library employed, ranging from 3D scenes in OpenGL to rich 2D vector graphics, as typically manipulated with tools such as Adobe Illustrator™. Our other goal is to create a flexible framework of higher expressive power than existing solutions, going beyond distortion to achieve smooth visual transitions between focus and context, resulting in lenses that can be manipulated more efficiently by users.

3 General Framework

Our general approach is positioned at a level of abstraction high enough for the model to be applicable to a variety of graphics frameworks, requiring the underlying libraries to provide as small a number of features as possible. It basically consists in transforming the representation at the pixel level after it has been rendered, independently of how it was rendered. To employ terminology drawn from 3D graphics, our technique works in image space, as opposed to object space. This approach has advantages beyond its wide applicability but also some limitations, which will be discussed later in Section 8.

3.1 Dual Rendering

All constrained magnification lenses featuring a regular shape share the following general properties, no matter how they transition between focus and context (see Figure 2):

- R_I : the radius of the focus region (a.k.a the flat-top), which we call *inner radius*,
- R_O : the radius of the lens at its base, i.e., its extent, which we call *outer radius*,
- MM : the magnification factor in the flat-top.

Applying a constrained lens to a representation effectively splits the viewing window into two regions: the *context region*, which corresponds to the part of the representation that is not affected by the lens, and the *lens region*, in which the representation is transformed. We consider the lens and context regions as separate

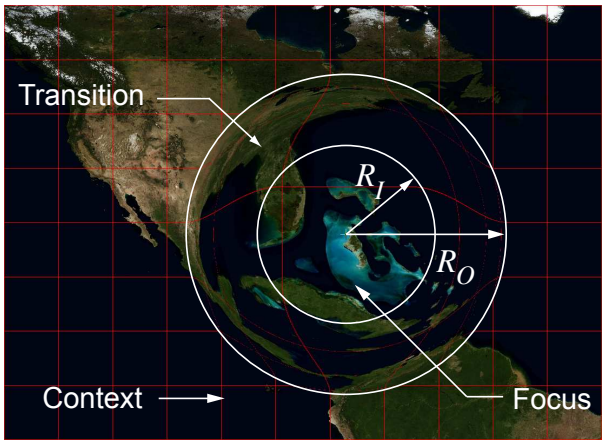


Fig. 2. Gaussian distortion lens. The level of detail in the flat-top is increased by a factor of $MM = 4.0$

buffers: the *context buffer* holds what is displayed in the absence of any lens; the *lens buffer* contains a rendering of the region corresponding to the bounding box of the lens. These basically correspond to the context and focus viewing windows of the original Sigma Lens model [13] in the case of 2D multi-scale representations, though here we are not limited to this type of graphics representation.

Our technique consists in asking the underlying graphics library for two separate rendering passes. One corresponds to what is seen in the context region and is stored in the context buffer, whose dimensions $w \times h$ match that of the final viewing window displayed to the user. The other rendering pass corresponds to what is seen in the lens region. Since we want the lens to actually provide a more detailed representation of objects in the magnified region, and not merely duplicate pixels from the previous rendering, the actual dimensions of the lens buffer are $2 \cdot MM \cdot R_O \times 2 \cdot MM \cdot R_O$. The buffer can thus accommodate a uniform magnification by a factor of MM of the lens region. The final viewing window displayed on screen can then be obtained through the arbitrary transformation and composition of pixels from both buffers. This includes displacement and compositing functions that will control the transition between the focus and context regions, as defined in the Sigma Lens unified model [13], which we briefly summarize.

3.2 Sigma Lenses

Sigma lenses build upon prior work on distortion lenses [18], but combine space, time and translucence to define new types of transitions between focus and context, in an effort to create more usable lenses (see Sections 5 and 6).

The standard transformation performed by graphical fisheyes consists in displacing all points in the focus buffer to achieve a smooth transition between focus and context through spatial distortion. This type of transformation can be defined through a drop-off function which models the magnification profile of the lens. The drop-off

Definition 1: displacement and compositing function $\mathcal{R}$

$$\mathcal{R}(x,y) = \begin{cases} (x_c + \frac{x-x_c}{MM}, y_c + \frac{y-y_c}{MM}) \bigotimes_{\alpha_{FT}} (x,y) & \{\forall (x,y) | \mathcal{D}(x,y) \leqslant R_I\} \quad (1.1) \\ (x_c + \frac{x-x_c}{\mathcal{G}_{scale}(\mathcal{D}(x,y))}, y_c + \frac{y-y_c}{\mathcal{G}_{scale}(\mathcal{D}(x,y))}) \bigotimes_{\mathcal{G}_{comp}(\mathcal{D}(x,y))} (x,y) & \{\forall (x,y) | R_I < \mathcal{D}(x,y) < R_O\} \quad (1.2) \\ (x,y) & \{\forall (x,y) | \mathcal{D}(x,y) \geqslant R_O\} \quad (1.3) \end{cases}$$

function is defined as:

$$\mathcal{G}_{scale} : d \mapsto s$$

where d is the distance from the center of the lens and s is a scaling factor. Distance d is obtained from an arbitrary distance function $\mathcal{D}$. A Gaussian-like profile is often used to define drop-off function $\mathcal{G}_{scale}$, as it provides one of the smoothest visual transitions between focus and context (see Figures 2 and 3). It can be replaced by other functions, which are already well-described in the literature [9], [18].

Distance functions producing basic regular lens shapes are easily obtained through $L(P)$-metrics [18]:

$$\mathcal{D} : (x,y) \mapsto \sqrt[P]{|x-x_c|^P + |y-y_c|^P}$$

where (x,y) are the coordinates of a point seen through a lens centered in (x_c, y_c), and $P \in \mathbb{N}^*$. $P = 2$ corresponds to a circular lens and $P = \infty$ to a square lens. As discussed later, more complex or irregular lens shapes can easily be obtained, e.g., by making R_I and R_O angle-dependent, usually in combination with $L(2)$.

In our approach, the overall process consists in applying a displacement function to all pixels in the lens buffer that fall into the transition zone: pixels between R_I and $MM \cdot R_O$ get compressed according to the drop-off function in such a way that they eventually all fit between R_I and R_O. Pixels of the lens buffer can then be composited with those of the context buffer that fall into the lens region.

When only interested in spatial distortion, generating the final representation simply consists in replacing pixels in the lens region of the context buffer by those of the lens buffer; in other words compositing them with the over operator ($\alpha = 1.0$). But other values of α and other operators in Porter & Duff's rich algebra [32] can be used to achieve interesting visual effects. It is for instance possible to obtain smooth, distortion-free transitions between focus and context by applying an alpha blending gradient centered on the lens. Or, as we will see later, a simple magnifier lens ($R_I = R_O$) can be made much more usable by making it uniformly translucent and coupling α to its speed.

The rendering of a point (x,y) in the final viewing window is controlled by function $\mathcal{R}$ (see Definition 1), where

$$p_{lens} \quad \otimes_\alpha \quad p_{context}$$

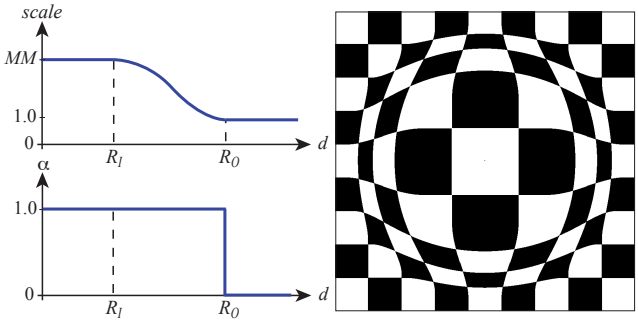


Fig. 3. Simple distortion lens

denotes the pixel resulting from alpha blending a pixel from the lens buffer and another from the context buffer with an alpha value of α. As with scale for distortion lenses, the alpha blending gradient can be defined by a drop-off function that maps a translucence level to a point (x,y) located at a distance d from the lens center:

$$\mathcal{G}_{comp} : d \mapsto \alpha$$

where α is an alpha blending value in [0, α_{FT}], α_{FT} being the translucence level used in the flat-top of the lens.

The flat-top region corresponds to case (1.1) of Definition 1, the transition to case (1.2), and the region beyond the lens boundaries, i.e., the context, corresponds to case (1.3). Detailed information about the model and functions summarized in this section can be found in the original paper on Sigma Lenses [13].

4 Instantiating Lenses in the Model

In our approach to the implementation of the Sigma Lens framework, new lenses are obtained very easily, only by defining functions $\mathcal{G}_{scale}$ and $\mathcal{G}_{comp}$. Some examples of interesting transitions follow.

First, for reference, a plain and simple distortion lens with a Gaussian-like drop-off function (Figure 3) can simply be obtained with the following instantiations of $\mathcal{G}_{scale}$ and $\mathcal{G}_{comp}$: $\{\forall d | R_I < d < R_O\}$

$$\begin{cases} \mathcal{G}_{scale}(d) = \frac{MM-1}{2} \cdot \cos(\frac{\pi}{R_O - R_I} \cdot d - \frac{\pi \cdot R_I}{R_O - R_I}) + \frac{MM+1}{2} \\ \mathcal{G}_{comp}(d) = 1.0 \end{cases}$$

Figures 1-a and 2 give examples of a distortion lens with a Gaussian-like drop-off.

The Blending lens described in [13], which achieves a smooth transition between focus and context through

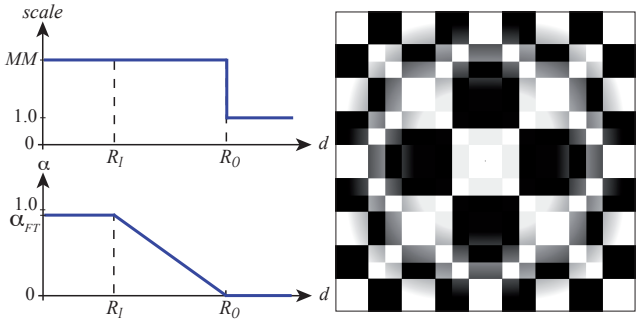


Fig. 4. Blending lens

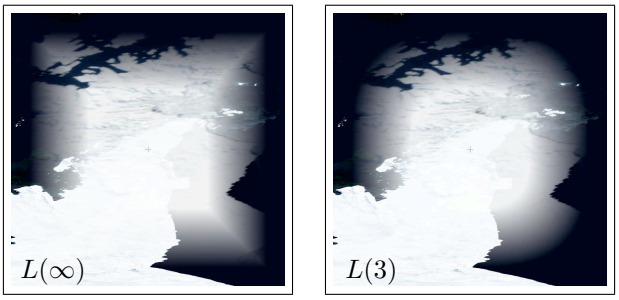


Fig. 5. Smoothing lens edges, on a magnified view of the Antarctic peninsula [34]

gradual alpha blending only (Figure 4), is easily instantiated using a linear drop-off for α:

$$\{\forall d | R_I < d < R_O\} \begin{cases} \mathcal{G}_{scale}(d) = MM \\ \mathcal{G}_{comp}(d) = \frac{\alpha_{FT}}{R_I - R_O} \cdot d - \frac{\alpha_{FT} \cdot R_O}{R_I - R_O} \end{cases}$$

This type of lens does not feature a continuous spatial transition between focus and context. Instead, visual continuity is achieved through increasing translucence.

As with any lens, other distance functions can be used [18]. $L(\infty)$, however, introduces artificial edges which give a false impression of spatial depth; if a square lens is required, $L(3)$ offers an interesting approximation, smoothing the edges while featuring a shape close to a square, as illustrated in Figure 5.

Spatial distortion and gradual alpha blending can of course be combined. Figure 6 shows an example of a so-called HOVERING lens as it appears to float above the representation.

$$\{\forall d | R_I < d < R_O\} \begin{cases} \mathcal{G}_{scale}(d) = \frac{1-MM}{R_O - R_I} \cdot d + \frac{MM \cdot R_O - R_I}{R_O - R_I} \\ \mathcal{G}_{comp}(d) = \frac{\alpha_{FT}}{R_I - R_O} \cdot d - \frac{\alpha_{FT} \cdot R_O}{R_I - R_O} \end{cases}$$

This lens mitigates issues of both distortion-only and alpha blending-only transitions: i) when the user is performing a focus targeting action[1], the target object no longer appears to temporarily move away from the approaching flat-top when entering the peripheral zone of the transition region (an effect due to the spatial distortion) as this zone is almost transparent; ii) this peripheral zone is still visible (undistorted) by translucence through the inner, distorted zone of the transition; iii) the distortion in this inner region contributes to visually differentiating focus and context during lens movements, and to the minimization of the distance between the point where an object disappears from the context and the point where it appears in the focus area.

4.1 Speed-coupling

In addition to the transition functions described earlier, the Sigma Lens framework allows for lens properties such as magnification factor, radius or flat-top opacity

1. The low-level and ubiquitous action which consists in moving the lens in the main window so as to position it over an object to be magnified in the flat-top is termed *focus targeting* [33].

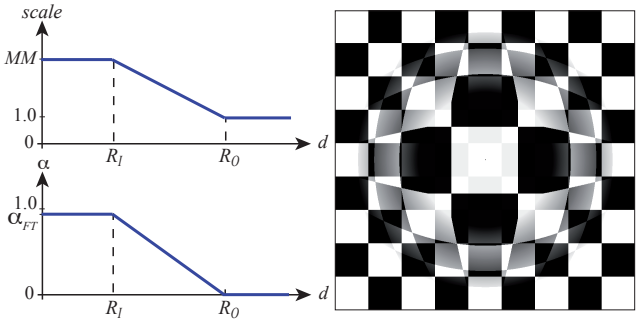


Fig. 6. Hovering Lens

to vary over time. The first example of lens to make use of dynamic properties was Gutwin's SPEED-COUPLED FLATTENING lens [33], which uses the lens' dynamics (velocity and acceleration) to automatically control magnification. By canceling distortion during focus targeting, speed-coupled flattening lenses improve the usability of distortion lenses. Basically, MM decreases toward 1.0 as the speed of the lens (operated by the user) increases, therefore flattening the lens into the context, and increases back to its original value as the lens comes to a full stop. Such behavior can easily be implemented in our approach using a simple interpolated low-pass filter (see [13] for detailed information). Let $\mathcal{S}(t)$ be the time-based function returning a numerical value that depends on the velocity and acceleration of the lens over time. The function is set to return a real value in $[0.0, 1.0]$. Making a lens parameter such as the magnification factor MM speed-dependent is then easily achieved by simply multiplying that parameter by the value of $\mathcal{S}(t)$, as shown in Figure 7-a.

Other properties can be made speed-dependent, including the radii R_I and R_O, as well as the translucence value in the lens' flat-top α_{FT}. For instance, the SPEED-COUPLED BLENDING lens, which to our knowledge features the best focus targeting performance, is also easily obtained as shown in Figure 7-b. This lens features a larger flat-top area compared to lenses of the same size that feature a transition zone. This makes the earlier-mentioned focus targeting task easier for the user from a purely motor perspective, but the occlusion stemming from the absence of a smooth transition zone counterbalances this theoretical advantage. The occlusion problem

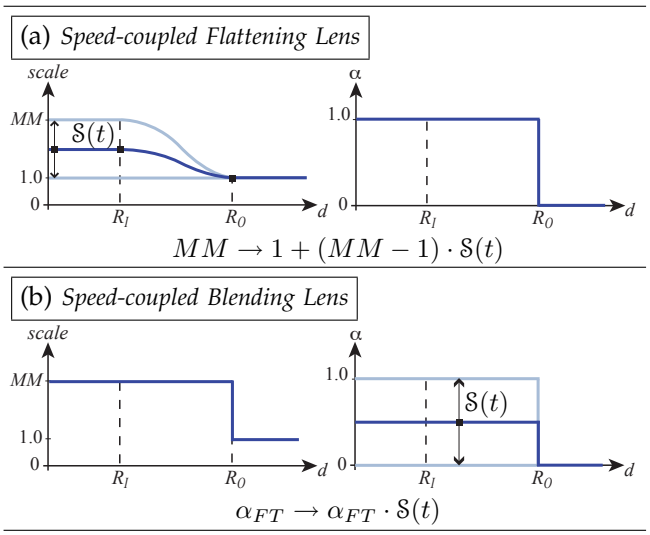


Fig. 7. Lenses with speed-coupled properties

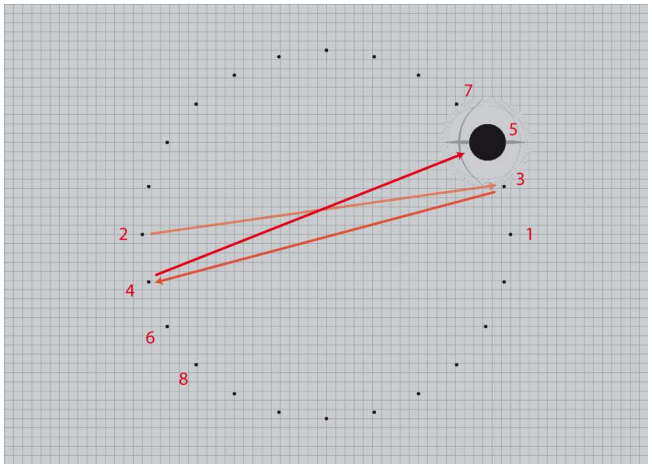


Fig. 8. Targets' order of appearance in a trial (targets are twice their actual relative size for legibility purposes).

is addressed by coupling α_{FT} to the speed of the lens: the lens becomes increasingly translucent as it is moved faster, and conversely (see Section 5 for more detail).

5 User Study: Focus Targeting Performance

In order to verify the above assumptions about the properties of various lenses and their impact on user performance, we ran a series of experiments that evaluate lens usability in different navigation situations. We briefly present in this section the results of studies reported in detail in [13], that compare a set of lenses on an elementary action involved in any navigation task: *focus targeting*. The task consists in putting a given target in the flat-top of the lens and is one of the building blocks of many higher-level navigation tasks such as searching [35]. We then report in Section 6 the results of new experiments that further test the two most efficient lenses in a more complex navigation task that involves both *global* and *local* navigation actions.

5.1 Apparatus

In all our experiments, we used a Dell Precision 380 equipped with a 3 GHz Pentium D processor, an NVidia Quadro FX4500 graphics card, a 1600 x 1200 LCD monitor (21") and a Dell optical mouse. The program was written with the multi-scale 2D framework presented in Section 7.1 . The application was limited to a 1400 x 1000 window with a black padding of 100 pixels in order to accommodate instruction messages.

5.2 Task and Procedure

Our focus targeting task consisted in acquiring a target in the flat-top of the lens as quickly as possible. In our experimental setting, the lens was centered on the mouse cursor. The task ended when the participant clicked the left mouse button, provided that the target was fully contained within the flat-top. Each trial consisted in performing 24 successive focus targeting tasks in a row. As illustrated in Figure 8, the targets were laid out in a circular manner. The order of appearance forced participants to perform focus targeting tasks in every direction, as recommended by the ISO9241-9 standard [36]. We decided to have only one target visible at a time, as we noticed during a pilot experiment in which all targets were visible that some participants were often taking advantage of the layout pattern to acquire the current target object by positioning the lens relative to that object's neighbors.

5.3 Experiment 1: Lens type and focus targeting

We first compared the *focus targeting* performance and limits of five lenses described earlier: a plain Magnifying Glass, a simple distortion lens (Fisheye), and Blending, Speed-Coupled Flattening, Speed-Coupled Blending (see Section 4). Focus targeting performance was evaluated at five different magnification factors (MM). Higher magnification factors make the task increasingly difficult: (i) the transition area becomes harder to understand as it must integrate a larger part of the world in the same rendering area, and (ii) it becomes harder to precisely position the target in the flat-top of the lens, the latter being controlled in the motor space of the context window. To test the limits of each lens, we included factors up to 14x. Our experiment was a 5×5 within-participant design: each participant had to perform several trials using each of the five lenses with five different magnification factors ($MM \in \{2, 4, 6, 10, 14\}$). Ten volunteers (7 male, 3 female), from 23 to 40 year-old (average 26.4, median 25), all with normal or corrected to normal vision, served in the experiment and allowed us to collect data on 11,500 actual focus targeting tasks.

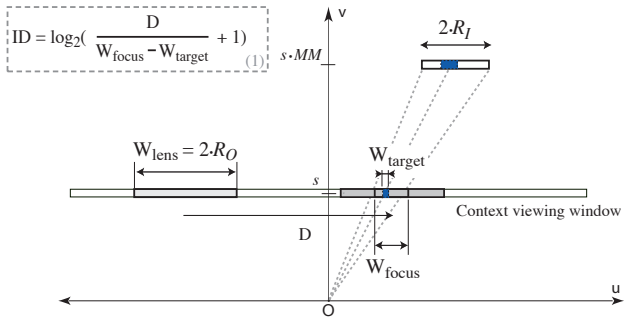


Fig. 9. Focus Targeting Task in a space scale diagram.

5.4 Results

Interestingly FISHEYE and BLENDING do not significantly differ in their performance. We initially thought that translucence could improve user performance by eliminating the space-based transition drawbacks. Transitioning through space indeed introduces distortion that makes objects move away from the approaching lens focus before moving toward it very fast, making focus targeting difficult [33]. BLENDING does not actually overcome this problem since it introduces a new one: the high cognitive effort required to comprehend transitions based on gradually increasing translucence which, as opposed to distortion-based transitions, do not rely on a familiar physical metaphor.

We expected speed-based lenses (SPEED-COUPLED FLATTENING and SPEED-COUPLED BLENDING) to outperform their static versions (FISHEYE and MAGNIFYING GLASS). Each focus targeting task can be divided into two phases: in the first phase, the user moves the lens quickly to reach the target's vicinity, while in the second phase, she moves it slowly to precisely position the target in the focus. In the first phase, the user is not interested in, and can actually be distracted by, information provided in the focus region since she is trying to reach a distant object in the context as quick as possible. By smoothly and automatically neutralizing the focus and transition regions during this phase, and then restoring them, speed-based lenses should help the user. Our results did actually support that this is the case for SPEED-COUPLED BLENDING and MAGNIFYING GLASS: smoothly neutralizing and restoring the focus of a MAGNIFYING GLASS by making it translucent does improve performance. However our participants were not significantly faster with SPEED-COUPLED FLATTENING than with FISHEYE. This was especially surprising since the study conducted in [33] showed a significant improvement in users performance with SPEED-COUPLED FLATTENING. We think this inconsistency is probably due to implementation differences: we implemented SPEED-COUPLED FLATTENING as a constrained lens while it was implemented as a full-screen lens by Gutwin. In full-screen lenses, distortion affects the whole representation, which thus benefits more from the neutralization effect than constrained lenses that only affect a limited area.

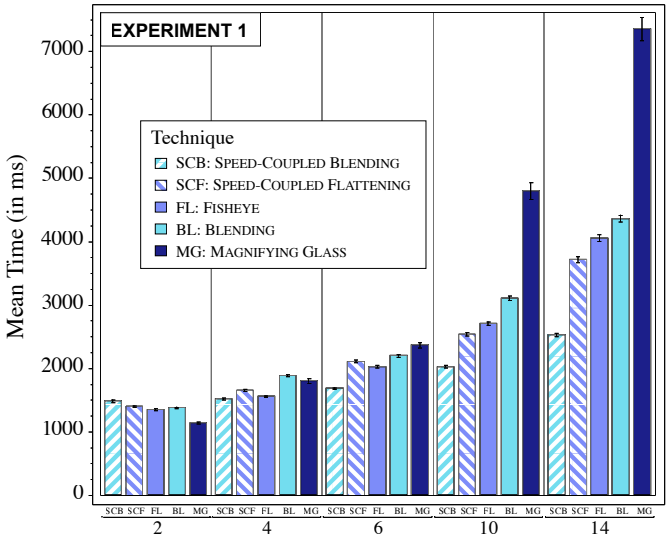


Fig. 10. Mean completion time per $Technique \times MM$ condition.

From a pure motor perspective, the difficulty of a focus targeting task can be evaluated as a view pointing task in a fixed-scale interface [37]. We can thus use Formula (1) in Figure 9 to quantify the difficulty of moving the lens' flat-top, of size W_{focus}, to a position where it will contain the target, of size W_{target}, initially located at a distance D from the lens' center. Formula (1) computes the Index of Difficulty, ID, of our focus targeting task. The lens' position in the context window is controlled in the visual and motor space of that window. W_{target}, W_{focus} and D are thus expressed in context pixels: in our experiment, $W_{target} = 8$ pixels and $D = 800$ pixels, while W_{focus} depends on a given $Lens \times MM$ condition: $W_{focus} = (2 \times R_I)/MM$. As MM increases, the size of W_{focus} decreases, making the task more difficult. For lenses of equal size (R_O), the size of the flat-top (R_I), and thus W_{focus}, vary depending on the lens type. MAGNIFYING GLASS and SPEED-COUPLED BLENDING lenses are made of a flat-top only: $W_{focus} = W_{lens} = 200$, while other lenses have to accommodate the transition within the same overall area: $W_{focus} = W_{lens}/2 = 100$ in our implementation. MAGNIFYING GLASS and SPEED-COUPLED BLENDING thus feature a larger flat-top than other lenses with the same overall size, consequently making focus targeting easier from a motor perspective: ID ranges from 3.2 to 6.3 for MAGNIFYING GLASS and SPEED-COUPLED BLENDING while it ranges from 4.2 to 8 for FISHEYE, SPEED-COUPLED FLATTENING and BLENDING. Our data showed that SPEED-COUPLED BLENDING is actually faster than all other lenses starting at $MM = 4$. However MAGNIFYING GLASS becomes the worst lens at $MM = 6$: its large opaque flat-top causes occlusion that makes the second phase of the task (precise positioning) too difficult to make users benefit from a larger flat-top.

Figure 10 summarizes these results. Our analyses only provide a partial order of performance between the five lenses but strongly support that SPEED-COUPLED BLENDING lens is the most efficient lens.

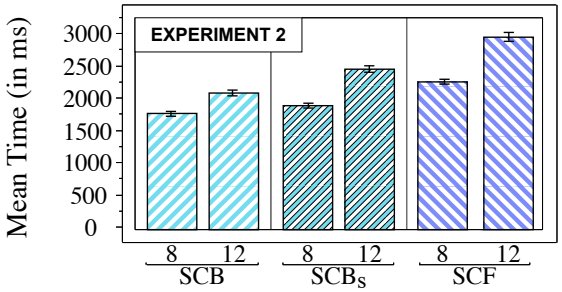


Fig. 11. Mean completion time per $MM \times Technique$ condition.

5.5 Experiment 2: Flat top size and focus targeting

Experiment 1 compared lenses with the same size (R_O). We found that SPEED-COUPLED BLENDING lenses outperform SPEED-COUPLED FLATTENING lenses, and attributed this performance gain (i) to the large flat-top of the SPEED-COUPLED BLENDING lens, which makes focus targeting easier from a motor perspective, and (ii) to the absence of distortion and reduction of occlusion effects through the coupling of focus translucence with lens speed. Experiment 2 aimed at better understanding the results of the previous experiment by identifying the contribution of both properties to this performance gain. We studied how SPEED-COUPLED BLENDING performed at two "extreme" sizes: 1) the lens has the same size as other lenses (W_{lens} = 200), and 2) the lens has the same size as the flat-top of lenses which accommodate a transition area and thus feature a smaller flat-top (W_{lens} = 100), making focus targeting harder from a motor perspective as explained earlier. We called the latter SPEED-COUPLED BLENDING$_{small}$ and compared it to SPEED-COUPLED BLENDING and SPEED-COUPLED FLATTENING, both from the previous experiment.

Two representative magnification factors were selected: $MM \in \{8, 12\}$. This experiment was thus a 3×2 within-participant design. Six volunteers (5 male, 1 female), from 23 to 40 year-old (average 27.8, median 25.5), all with normal or corrected to normal vision, served in the experiment. Figure 11 illustrates our results: even at the same level of motor difficulty, the SPEED-COUPLED BLENDING lens still performs better than the SPEED-COUPLED FLATTENING lens. This means that interface designers are given several options to improve upon a classical lens such as FISHEYE: 1) they can either get a smaller but more efficient lens (in terms of focus targeting performance), saving screen real-estate for the context, 2) if the latter is not critical they can make the SPEED-COUPLED BLENDING lens occupy the same space as a FISHEYE would, further improving focus targeting performance, or 3) find a balance between these solutions.

6 EXPERIMENT 3: TRANSLUCENCE AND SPEED-DEPENDENCE

Lieberman used translucence in Powers of Ten Thousands [38], a bifocal display technique that makes the focus and context views share the same physical screen space, by using multiple translucent layers. Even though it has been shown to be usable in exploratory studies [39], [40], this type of representation based on transparent or translucent layers is cognitively demanding, causing visual interferences that are the source of serious legibility problems, and requiring additional mental effort from the user to relate focus and context. Translucence can hence affect targeting performance, especially when targets are superimposed on a complex background such as a map or photograph. Speed-dependent properties can also be confusing as they affect the lens' appearance depending on cursor movements. As the simple abstract world we used in the first two experiments might have hidden potential negative effects caused by translucence, we present here further controlled experiments aimed at verifying that our comparative lens performance ordering is still valid in more realistic environments.

6.1 Preliminary experiment

In [13], we had already conducted a preliminary study to assess the potential effects of translucence on targeting performance. The task was the same as described earlier, but the 24 targets were laid out on a satellite photograph, and could either be filled with a fully opaque red color or with a translucent red, in which case they blended into the background and were less easily identifiable. The satellite photograph was a 7000×5000 pixels portion of NASA's Blue Marble world map [34], providing an appropriate level of detail in both the focus and context regions. This experiment yielded a performance ordering consistent with that observed in the first two. Target opacity had a significant effect on performance only for BLENDING. This result is not unexpected as the BLENDING lens can be prone to visual interference between focus and context in the transition region depending on the nature of the representation, especially when non-contrasted objects are targeted. No matter how aesthetically pleasing (several participants noted that it produced very nice graphical renderings), the BLENDING lens suffers from a lack of reliance on a familiar physical metaphor, and proneness to visual interference in the transition region. The SPEED-COUPLED BLENDING lens, however, does not seem to suffer from these problems, as its use of translucence is very different: it can be seen as a magnifying glass whose content smoothly fades out to prevent occlusion at focus targeting time.

In the remainder of this section, we further evaluate speed-dependent lenses by introducing a task that implies more speed variations when operating the lens by forcing both *global navigation* (large movement followed by a stop) and *local navigation* (small movements for fine-grain positioning), as the preliminary experiment described above suggests that SPEED-COUPLED BLENDING could cause some legibility problems since it is neither opaque nor fully transparent during local navigation phases.

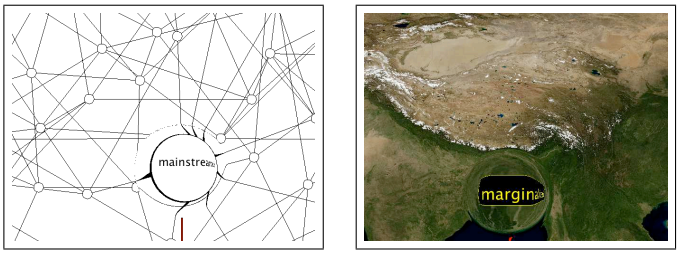


Fig. 12. (a) Exp. 4_{graph} (labels displayed in black) and (b) Exp. 4_{map} (labels displayed in yellow over a black background). Viewports are cropped to show details.

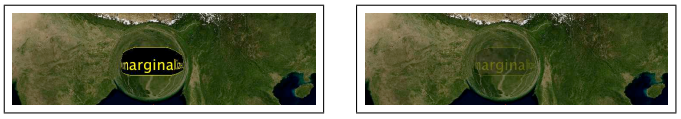


Fig. 13. Exp. 4_{map}: Opaque target vs. Translucent target

6.2 Task and procedure

Transitions based on distortion or transparency add complexity to the representation, and can affect usability differently depending on the nature of the objects displayed. To test lens usability in a wide range of realistic situations, we used different representations. We conducted two experiments illustrated in Figure 12 based on two different types of representation: a network (vector graphics) for Experiment 4_{graph} ($Bg = graph$), and a high-resolution satellite map (bitmap) for Experiment 4_{map} ($Bg = map$). In both cases, the task is the same. A word is displayed to the user in three locations: top, center and bottom of the screen. Participants are instructed to memorize this word as they will have to search for it in the representation. Once this target word is memorized, participants put the cursor on a red square (20×20 pixels) located at the center of the screen and press the space bar to start the trial. The red square disappears, as does the target word displayed at the center (the ones displayed in the top and bottom margins remain displayed throughout the trial in case participants forget it). Words appear successively in the same locations as the circular targets did in previous experiments. When a word appears, the participant has to move the lens over it to be able to read it (a word represents about 27×8 pixels in the context and is thus unreadable at this scale). If the word is not the target word, she proceeds to the next one by pressing the space bar. This is repeated until she finds the target word, in which case she clicks the left mouse button to end the current trial and start a new one.

In both cases (mouse click or space bar), the word must be *in focus*. For a word to be considered in focus, our software uses the following criterion: the intersection area between the word and the lens' flat-top should be at least 66% of the flat-top area (thus ensuring that the word can be read). A word can however never be fully displayed in the flat-top. This is to force participants to perform local navigation as described earlier. Here again to compare lenses both in usual and extreme conditions, we use two magnification factors ($MM \in \{8, 12\}$). Font size is set to 42 pts (at context scale) for $MM = 8$ and 28 pts for $MM = 12$, so that the lens' flat-top can display at most 6 letters at full magnification. We use two word lengths to test the effect of the amount of local navigation on lens performance ($LabLength \in \{8, 12\}$). To decrease the probability of a participant recognizing a word only based on a few specific letters (which would reduce the amount of local navigation), we have chosen words that are similar to a certain degree: all 8-letter words start with 'a' and end by 'ed'; all 12-letter words start with 'm' and end by 'ed'. We also had to make participants believe that the target word could appear at any time so they don't turn the task into a pure routine motor task. To that end, we introduced a secondary factor, $Rank$, that was used to control how many objects participants had to inspect before finding the target word for a given trial. $Rank$ could take four different values: {2, 4, 6, 8}.

The graph of Experiment 4_{graph} contained 76 nodes that were laid out in a semi-random manner so as to provide a uniform density and to make sure that a node would coincide with each of the potential target locations (a word was always displayed inside a node – see Figure 12-a). For Experiment 4_{map}, we used the NASA world map mentioned earlier (Figure 12-b). In this latter experiment, we introduced an additional factor: in half of the trials, words were opaque (O) while in the other half, they were translucent[2] (T) ($Opacity \in \{O, T\}$), as shown in Figure 13. We hypothesized that background and focus might be perceptually interpreted as one illegible image if contrast is not strong enough, especially for SPEED-COUPLED BLENDING as it itself makes use of translucence. $Opacity$ was not included as a factor in Experiment 4_{graph} because sharp edges displayed on a uniform background are strongly contrasted.

Twelve volunteers (9 male, 3 female), from 23-33 years (avg. 26.7, med. 26.5), all with normal or corrected to normal vision, no color blindness, served in the experiment. Each of them was involved in both experiments. Experiment 4_{graph} lasted around 40 minutes, 4_{map} lasted around 1 hour (instructions were shorter since all participants were already familiar with the task). The two experiments were performed on two different days to minimize fatigue and boredom. Each experiment was composed of four blocks, one per $Lens \times MM$ condition. Successive changes of $Lens$ values would have been too disturbing. To counterbalance $Lens$ presentation order, 6 participants saw the 2 SPEED-COUPLED FLATTENING blocks first, while the 6 other participants started with SPEED-COUPLED BLENDING. We always presented block $MM = 8$ first for a given $Lens$, so as to avoid harder conditions being presented first to participants.

A $Lens \times MM$ block contained 4 series of 4 trials in

2. When $Opacity = T$, words and their black background are rendered with an alpha channel set to 0.2, thus blending into the satellite image, which makes them more difficult to identify (Figure 13).

Exp. 4_{graph}	Exp. 4_{map}
2 $Lens$	2 $Lens$
x 2 MM	x 2 MM
x 2 $LabLength$	x 2 $LabLength$
	x 2 $Opacity$
x 4 $Rank$	x 4 $Rank$
x 2 replications	x 2 replications
x 12 participants	x 12 participants
= 768 trials	= 1536 trials

TABLE 1
Summary of design for Experiment 3

	Exp. 4_{graph}	Exp. 4_{map}
$Lens$	$F_{1,11} = 54$, $p < .0001$	$F_{1,11} = 150$, $p < .0001$
MM	$F_{1,11} = 32$, $p < .0001$	$F_{1,11} = 48$, $p < .0001$
$Lens \times MM$	$F_{1,11} = 20$, $p < .0001$	$F_{1,11} = 29$, $p < .0001$

TABLE 2
Significant effects revealed by analysis of variance of MM and $Lens$ factors on $Acquisition$ errors

Experiment 4_{graph} and 8 series of 4 trials in Experiment 4_{map}. A series contained words of the same length and presented the four different $Rank$ values in a pseudo-random order to ensure that the overall difficulty was the same for all participants. In Experiment 4_{graph}, a participant saw alternatively a series of 8-letter words and a series of 12-letter words, twice. In Experiment 4_{map}, a participant saw alternatively two series of 8-letter words and two series of 12-letter words, twice; one series per opacity value ($Opacity = O$ then $Opacity = T$). Table 1 summarizes this experimental design.

6.3 Results and Discussion

We collected three main measures for our analyses: (i) completion $Time$, i.e., the time interval between the appearance of the first word and the click on the target word; (ii) the number of $Reading$ errors, i.e., when the participant notices that she has pressed the space bar instead of clicking on the target word or if she has visited an abnormally large number of words[3]; (iii) the number of $Acquisition$ errors, i.e., when the participant presses the space bar or clicks while the word is *not in focus*. In this case, the message "target not in focus" would flash and the participant would have to adjust the lens (the trial continues, the timer is not reset).

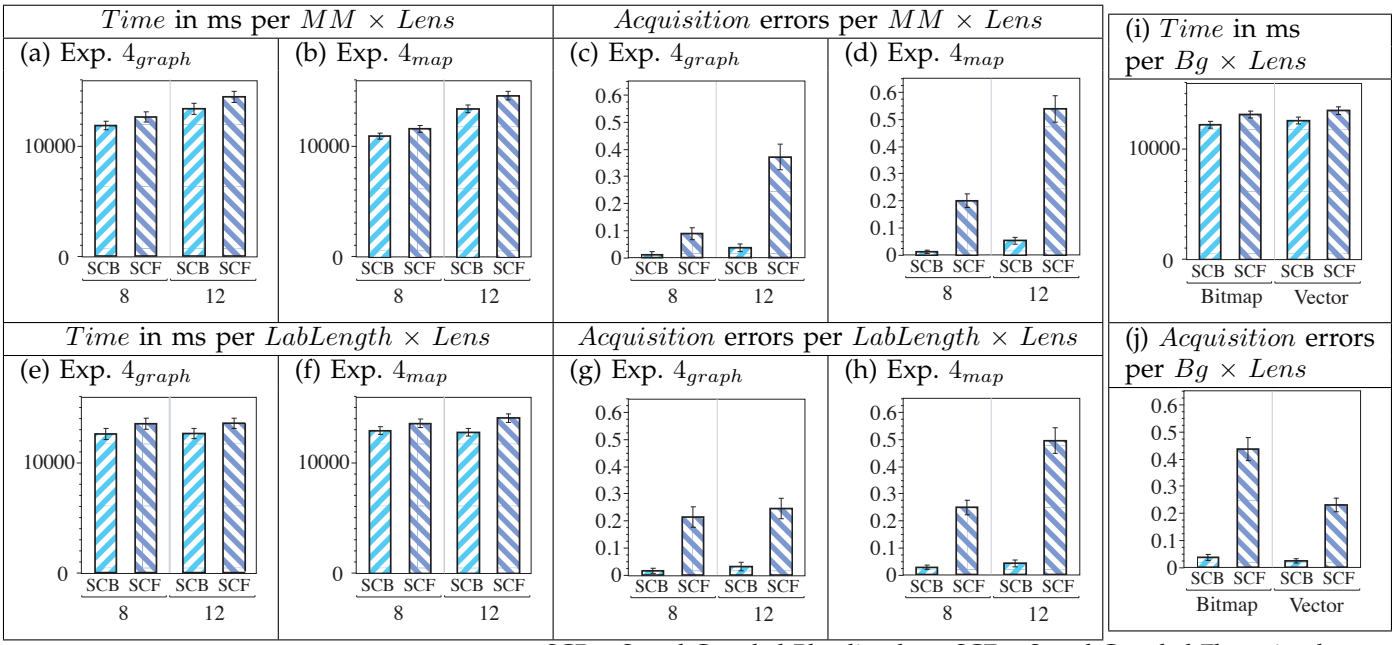
A total of 62 trials among 768 were restarted because of $Reading$ errors in Experiment 4_{graph} ($\sim$ 8%) and 128 trials among 1536 in Experiment 4_{map} ($\sim$ 9%). Note that the recorded time for these particular trials was potentially biased, as participants could avoid some cognitive aspects of the task and turn it into a simple motor task. For instance, if a participant remembered the rank at which the target word appeared when realizing her failure, she could avoid having to carefully read the intermediary words the second time, simply pointing at them. To avoid analyzing data with an unbalanced number of measures per factor, we left these trials in our data after having checked that (i) $Reading$ errors were uniformly distributed among the primary factors and (ii) that neither primary factors nor $Rank$ had a significant effect on the number of $Reading$ errors.

3. Instructors told participants that if they had visited about 15 words without having seen the target word this meant they had missed it. In this case, the participants could press the Escape key to skip the current trial and restart it (the timer was reset).

Figures 14-a to 14-h show the data collected along the $Lens$, MM and $LabLength$ factors. Regarding $Time$, we did not observe any significant effect of any condition. We can see in the first two columns that participants were faster using SPEED-COUPLED BLENDING than SPEED-COUPLED FLATTENING. This difference was however not statistically significant.

Differences in accuracy were stronger. In both experiments, participants were more accurate using SPEED-COUPLED BLENDING than SPEED-COUPLED FLATTENING. $Lens$ and MM had a significant effect on both $Time$ and $Acquisition$ errors (Table 2). Furthermore, differences between lenses in terms of accuracy increased with the magnification factor (Figures 14-c and 14-d): $Lens \times MM$ had a significant effect on $Acquisition$ errors.

Participants were more accurate with shorter words: $LabLength$ had a significant effect on $Acquisition$ errors in Experiment 4_{map} ($F_{1,11} = 22$, $p < .0001$), see Figure 14-h. However, participants were not significantly faster with shorter words. We did not expect this observation since the task is supposed to be harder from a motor perspective. The local navigation required by longer words probably penalizes overall task performance more than the motor aspects involved in a simple focus targeting task. In addition, lenses seem to be unequally affected by word length: $Lens \times LabLength$ was significant on $Acquisition$ errors ($F_{1,11} = 17$, $p < .0001$) in Experiment 4_{map}. These results tend to show that SPEED-COUPLED BLENDING better supports local navigation than SPEED-COUPLED FLATTENING (Figures 14-f and 14-h). This latter effect, observed only in Experiment 4_{map}, reinforces our intuition that lens usability is affected by the type of representation. To have a closer look at this effect, we built a table resulting from the concatenation of trials of Experiment 4_{graph} ($Bg = Graph$) and trials where $Opacity = O$ of Experiment 4_{map} ($Bg = Map$). Analysis of variance revealed a significant effect of Bg on $Acquisition$ errors ($F_{1,11} = 17$, $p < .0001$) but not on $Time$ (Figure 14-i and 14-j). Participants were more accurate on a vector-based representation (the network) probably because the white background helped perceive the limits of the label. We also found one interaction effect $Lens \times Bg$ on $Acquisition$ errors ($F_{1,11} = 14$, $p = 0.0002$): surprisingly SPEED-COUPLED FLATTENING is more penalized by background type than SPEED-COUPLED BLENDING is (left of Figure 14-j). Transparency does not hinder performance on complex representations, and seems more robust than distortion for high magnification factors.



SCB = Speed-Coupled Blending lens, SCF = Speed-Coupled Flattening lens.

Fig. 14. Exp. 4_{graph} and 4_{map}: *Time* and *Acquisition* errors.

Hence, our hypotheses about usability problems due to the use of transparency are not supported by this experiment. Analysis of the *Opacity* factor in Experiment 4_{map} also supports SPEED-COUPLED BLENDING's robustness. The effect of *Opacity* on *Acquisition* errors was significant ($F_{1,11} = 6$, $p = 0.01$), which is consistent with participants remarks about the reading difficulty they had when labels were translucent. Interaction effect *Lens* × *Opacity* was also significant on *Acquisition* errors ($F_{1,11} = 5$, $p = 0.02$) revealing that users were more strongly affected by label opacity with SPEED-COUPLED FLATTENING than with SPEED-COUPLED BLENDING. As a summary, speed-coupled translucence does not have a negative impact on reading performance. The dynamically varying translucence of the SPEED-COUPLED BLENDING lens does not cause significant visual interference, even on complex scenes featuring a low level of contrast. The SPEED-COUPLED BLENDING lens remains more efficient than the SPEED-COUPLED FLATTENING lens.

7 IMPLEMENTATIONS

According to the previous evaluations, Sigma Lenses should prove useful in various types of graphical user interfaces. The next step consists in investigating how to add support for the framework in existing graphics environments. Though implementation is not straightforward, the approach is generic enough that it can be implemented in different environments with only a few requirements. The underlying graphics library must allow i) for the scene to be rendered at different levels of detail, and ii) for the pixels that constitute the two rendered images (the context region and the lens region) to be manipulated and composited before the actual rendering to the screen occurs. The following sections describe two different implementations, one for multi-scale 2D graphics, the other for OpenGL scenes.

7.1 Multi-scale 2D Framework

Pad [1] was one of the first toolkits designed for the implementation of multi-scale interfaces. Since then, several other zoomable user interface toolkits have been developed, including Piccolo [2] and ZVTM [3]. Our implementation is based on the latter, an open source toolkit built on top of Java2D. It allows for the same scene to be rendered from different viewpoints. Furthermore, as the toolkit does not rely on Java's internal double buffering mechanism but implements its own and makes the offscreen rendering buffers publicly accessible, adding support for Sigma Lenses was easy. Our extension works directly on these offscreen images, and is independent of their content. It is thus readily compatible with all graphical objects that can be displayed by the toolkit, including arbitrary vector-based shapes, bitmap images and text rendered with any font.

A call to the extension is simply inserted in the rendering loop, between the rendering of the main offscreen buffer, which corresponds to the context region, and its copy to the screen. The extension asks for a second rendering from the viewpoint of a camera set to observe the lens region. Clipping algorithms internal to the toolkit make sure that only objects visible through each camera get projected and drawn in the associated offscreen buffer. Our extension then creates the focus+context representation. As a significant part of the final rendering will match the context buffer exactly, the latter serves as the target of the transformation to prevent unnecessary copy operations. For each pixel (x, y) in the subregion corresponding to the lens, we

simply read a pixel in the raster associated with the lens buffer using an index computed through function $\mathcal{R}$, thus achieving magnification as well as distortion depending on the drop-off defined by $\mathcal{G}_{scale}$. This pixel gets composited with the original pixel (x, y) from the context buffer, using alpha blending if necessary with a value of α computed through function $\mathcal{R}$ (more specifically $\mathcal{G}_{comp}$). Pixels outside the lens region are left untouched.

The core classes of the extension represent about 700 lines of actual code, supporting three different `BufferedImage` types (type varies depending on operating system and color depth). Taking advantage of the inheritance mechanism, a new lens is then typically written with less than 100 lines of code: most of those lines are dedicated to constructors, getters, and setters. Lines that actually modify the rendering of the transition, i.e., implementations of $\mathcal{G}_{scale}$ and $\mathcal{G}_{comp}$ specific to each lens, only account for about 10% of those 100 lines except for lenses with speed-dependent properties, in which case the coupling with the interpolated low-pass filter adds an extra 80 lines of code.

Performance. We measured the performance of this implementation on a representation consisting of a multi-scale version of the high-resolution (86400×43200 pixels) Blue Marble Next Generation world map from NASA [34]. The pyramid consists of 2,728 tiles, each 1350×1350 pixels in size. We overlaid shapefile data representing country boundaries as vector graphics on top of the map, amounting to 23,715 segments forming 1,375 polygons of various shapes and sizes. Performance tests were run on a Windows XP PC, equipped with a 3 GHz Pentium D processor, 2 GB of RAM and an NVIDIA Quadro FX 4500 graphics card with 512 MB of memory, using Java 1.6 with the default DirectX rendering pipeline enabled. The application ran in a 1280×640 window. All lenses were 200px wide, occupying about 5% of the rendering area, and were tested at various magnification factors ranging from 2x to 8x. Rendering the content of the context buffer took an average of 23ms (base condition). Average time spent rendering and compositing the content of the lens buffer ranged from 24ms in the best case (2x distortion lens) to 74ms in the worst case (8x hovering lens), corresponding to overall frame rates varying between 21fps and 10fps. While not extremely high, the measured frame rates show that our framework can be implemented in graphics environments that do not benefit from significant hardware acceleration such as plain Java2D, and still achieve interactive frame rates for relatively complex representations.

7.2 3D Framework

The second implementation takes advantage of programmable graphics hardware. Lenses are written with the OpenGL Shading Language (GLSL). The rendering process is as follows. First, the lens region is rendered to a texture, thanks to a framebuffer object that makes it possible to render to other destinations than those provided by the window system. The projection frustum is set to match the lens region, so that only objects visible in that region get rendered in this first pass. Only the projection frustum is changed, not the camera position, so as not to introduce any discontinuity between the context and lens regions. The viewport's dimensions, and thus those of the target texture are set[4] according to the lens buffer's size: $2 \cdot MM \cdot R_O \times 2 \cdot MM \cdot R_O$. Second, the viewport and perspective projections are set according to the dimensions of the application's window to render the scene as it would look like in the absence of any lens. Third, the texture generated during the first step is mapped on a plane matching the lens region using normalized texture coordinates. If the scene were to be rendered through the standard fixed function pipeline, this higher-resolution texture would perfectly blend into the context. Instead, we use a fragment shader that implements the lens' displacement and compositing functions. The data contained in the texture is accessed through this fragment shader. Context fragment color data is accessed from the lens texture through a texture sampler by using the texel coordinates calculated by the standard fixed function pipeline in previous stages of the rendering process:

```
//focus color data access with
texture2D(lensTex,vec2(x*1.0/texWidth,y*1.0/texHeight));

//context color data access with
texture2D(lensTex,gl_TexCoord[texNum].st);
```

The fragment shader is independent of the content of the scene observed. It is an implementation of the specific instance of function $\mathcal{R}$ defining one particular lens, nothing more. External data controlled by the application, such as the position of the lens (x_c, y_c), its magnification factor MM, or the cursor speed, are sent to the shader using GLSL `uniform` variables. The source code of a standalone shader is thus very small, typically 60 to 80 lines, instructions related to a specific instance of $\mathcal{G}_{scale}$ or $\mathcal{G}_{comp}$ typically amounting to two to four lines.

Performance. We used the same hardware configuration as in the 2D implementation section, except that the software was running under Linux, with version 169.07 of the NVIDIA driver. The application ran in a 1050×750 window and displayed 3D objects of varying complexity. We tested lenses 200px to 800px wide, at magnification factors ranging from 2x to 8x. Measures indicate that, in these ranges, the values of MM and R_O have negligible influence on performance. In all cases, rendering the content of the lens region takes approximately the same amount of time as rendering the context in the frame buffer. For instance, a 3D model made of 69,451 facets is displayed at 123fps with a lens vs. 231fps with no lens; another model made of 345,944 facets

4. If the dimensions exceed the maximum texture size permitted by the graphics card, the lens region can be rendered in smaller tiles that are then read using the multi-texture access capabilities of fragment shaders.

is displayed at 28fps vs. 53fps. Performance could be improved if necessary by implementing manual frustum culling, taking the different sizes of the context and lens regions into account (lens rendering time would depend on the lens region's dimensions). Time spent distorting and compositing in the fragment shader is not significant compared to scene rendering time.

8 Discussion and Future Work

As opposed to most techniques described in the literature, we wanted to develop a lightweight, representation-independent approach to the problem of focus+context navigation based on constrained lenses. This choice has implications in terms of implementation effort, expressive power and graphics performance.

Graphics performance will obviously vary depending on the complexity of the representation and on the underlying graphics framework's capabilities, such as clipping algorithms and level of support for hardware acceleration; but even in cases where most of the computations are done on the software side, acceptable frame rates can be achieved for most lenses on reasonably complex scenes such as the one described in Section 7.1. Performance can however be an issue when highly magnifying lenses defined by complex drop-off functions for both scale and translucence are used in a non hardware-accelerated framework. In that case, values for $\mathcal{G}_{scale}$ (displacement function) and $\mathcal{G}_{comp}$ (blending gradient) can be pre-computed for each target pixel in the lens region and stored in a data structure such as an array or texture for fast lookup. Memory consumption will amount to a maximum of two structures storing $(2R_O)^2$ floating point numbers, and can be lowered to a quarter of that value by taking advantage of vertical and horizontal symmetry for lenses based on $L(P)$-metrics.

The shape of a lens does not necessarily have to feature the above-mentioned symmetry. While most constrained lenses have been restricted to regular shapes such as circles and squares (Magic Lenses [41] excepted), irregular perimeters can be obtained fairly easily by defining them using parametric equations that make R_I and R_O angle-dependent. We are planning to better integrate such perimeters in the framework, our final goal being to build *adaptive lenses* whose shape changes to provide more relevant magnifications of the objects in focus by accessing information about their geometry.

One strong point of our approach is its good expressive power which comes at a relatively small implementation cost. The two implementations described in Section 7 show that it can be implemented with minimal effort in various graphics frameworks, only requiring offscreen drawing capabilities, a feature commonly available in graphics APIs, and the possibility to draw the representation at two different scales. Once the core lens programming elements are in place (when required at all), writing a new lens typically takes only a few lines, most of the code being the same from one lens to another, and differences residing mainly in the definition of transitions.

Working on the rendered scene, distorting and compositing the focus and context regions by manipulating pixels makes our technique fully independent of the actual 2D or 3D objects that constitute the representation. However, this approach also has some limitations. One limitation is that performance depends on the magnification factor and size of the lens, as a consequence of the increased level of detail in the focus. Another issue is related to the quality of the rendering in the transition area for lenses that make use of distortion: as points get compressed in the transition region, small objects such as points may be lost, and lines cut if they fall between pixels, especially for high magnification factors. This problem can be partially addressed by specifying an expression of $\mathcal{G}_{comp}$ that matches $\mathcal{G}_{scale}$, but this, in turn, can introduce unwanted rendering artefacts. Solutions based on MIP maps [42] can help address this issue, but will be computationally expensive if the content of the representation in the magnified region varies continuously.

9 Summary

We have presented an approach to the implementation of constrained magnification lenses for navigation in large workspaces, based on the displacement and compositing of pixels from two renderings at different scales. This technique has several practical advantages such as its independence with respect to the type of representation and its low cost of implementation. But more importantly, it allows for a variety of dynamic transitions between the focus and context regions, including – but not limited to – those covered in the Sigma Lens framework. This framework enables the design of new lenses, such as the Speed-Coupled Blending lens, which has been shown to perform very efficiently through a series of controlled experiments that cover cognitive aspects of navigation tasks ranging from motor performance to legibility issues on various types of representations.

References

[1] K. Perlin and D. Fox, "Pad: an alternative approach to the computer interface," in *SIGGRAPH '93: Proc. of the Conf. on Computer Graphics and Interactive Techniques*. ACM, 1993, pp. 57–64.

[2] B. B. Bederson, J. Grosjean, and J. Meyer, "Toolkit design for interactive structured graphics," *IEEE Transactions on Software Engineering*, vol. 30, no. 8, pp. 535–546, 2004.

[3] E. Pietriga, "A Toolkit for Addressing HCI Issues in Visual Language Environments," in *VL/HCC'05: IEEE Symp. on Visual Languages and Human-Centric Computing*. IEEE Comput. Soc., 2005, pp. 145–152.

[4] C. Ware and S. Osborne, "Exploration and virtual camera control in virtual three dimensional environments," in *SI3D '90: Proc. of the 1990 Symp. on Interactive 3D graphics*. ACM, 1990, pp. 175–183.

[5] W. C. Donelson, "Spatial management of information," in *SIGGRAPH '78: Proc. of the 5th Conf. on Computer Graphics and Interactive Techniques*. ACM, 1978, pp. 203–209.

[6] J. Lamping, R. Rao, and P. Pirolli, "A focus+context technique based on hyperbolic geometry for visualizing large hierarchies," in *CHI '95: Proc. of the SIGCHI Conf. on Human Factors in Comp. Syst.* ACM/Addison-Wesley Publishing Co., 1995, pp. 401–408.

[7] T. Munzner, F. Guimbretière, S. Tasiran, L. Zhang, and Y. Zhou, "Treejuxtaposer: scalable tree comparison using focus+context with guaranteed visibility," in *SIGGRAPH 2003: Proc. of the Conf. on Comp. Graphics and Interactive Tech.* ACM, 2003, pp. 453–462.

[8] E. R. Gansner, Y. Koren, and S. C. North, "Topological fisheye views for visualizing large graphs," *IEEE Transactions on Visualization and Computer Graphics*, vol. 11, no. 4, pp. 457–468, 2005.

[9] M. S. T. Carpendale, J. Ligh, and E. Pattison, "Achieving higher magnification in context," in *UIST '04: Proc. of the ACM Symp. on User Interface Software and Technology*. ACM, 2004, pp. 71–80.

[10] G. Shoemaker and C. Gutwin, "Supporting multi-point interaction in visual workspaces," in *CHI '07: Proc. of the SIGCHI Conf. on Human Factors in Comp. Systems*. ACM, 2007, pp. 999–1008.

[11] G. Ramos, A. Cockburn, R. Balakrishnan, and M. Beaudouin-Lafon, "Pointing lenses: facilitating stylus input through visual- and motor-space magnification," in *CHI '07: Proc. of the SIGCHI Conf. on Human Factors in Comp. Syst.* ACM, 2007, pp. 757–766.

[12] P. Blenkhorn, G. Evans, A. King, S. H. Kurniawan, and A. Sutcliffe, "Screen magnifiers: Evolution and evaluation," *IEEE Computer Graphics and Applications*, vol. 23, no. 5, pp. 54–61, 2003.

[13] E. Pietriga and C. Appert, "Sigma lenses: focus-context transitions combining space, time and translucence," in *CHI '08: Proceeding of the twenty-sixth CHI Conf. on Human factors in computing systems*. ACM, 2008, pp. 1343–1352.

[14] G. G. Robertson and J. D. Mackinlay, "The document lens," in *UIST '93: Proc. of the ACM Symp. on User Interface Software and Technology*. ACM, 1993, pp. 101–108.

[15] M. Sarkar, S. S. Snibbe, O. J. Tversky, and S. P. Reiss, "Stretching the rubber sheet: a metaphor for viewing large layouts on small screens," in *UIST '93: Proc. of the ACM Symp. on User Interface Software and Technology*. ACM, 1993, pp. 81–91.

[16] M. S. T. Carpendale, D. J. Cowperthwaite, and F. D. Fracchia, "3-dimensional pliable surfaces: for the effective presentation of visual information," in *UIST '95: Proc. of the ACM Symp. on User Interface Software and Technology*. ACM, 1995, pp. 217–226.

[17] T. A. Keahey and E. L. Robertson, "Nonlinear magnification fields," in *INFOVIS '97: Proc. of the 1997 IEEE Symp. on Information Visualization*. IEEE Comp. Soc., 1997, pp. 51–58.

[18] M. S. T. Carpendale and C. Montagnese, "A framework for unifying presentation space," in *UIST '01: Proc. of the ACM Symp. on User Interface Software and Technology*. ACM, 2001, pp. 61–70.

[19] C. Gutwin and A. Skopik, "Fisheyes are good for large steering tasks," in *CHI '03: Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*. ACM, 2003, pp. 201–208.

[20] A. Cockburn, A. Karlson, and B. B. Bederson, "A review of overview+detail, zooming, and focus+context interfaces," *ACM Comput. Surv.*, vol. 41, no. 1, pp. 1–31, 2008.

[21] E. LaMar, B. Hamann, and K. I. Joy, "A magnification lens for interactive volume visualization," in *PG '01: Proc. of the 9th Pacific Conf. on Comp. graphics and Applications*. IEEE Comp. Soc., 2001.

[22] Y. Yang, J. X. Chen, and M. Beheshti, "Nonlinear Perspective Projections and Magic Lenses: 3D View Deformation," *IEEE Computer Graphics and Applications*, vol. 25, no. 1, pp. 76–84, 2005.

[23] J. Brosz, F. F. Samavati, M. T. C. Sheelagh, and M. C. Sousa, "Single camera flexible projection," in *NPAR '07: Proc. of the 5th International Symp. on Non-photorealistic Animation and Rendering*. ACM, 2007, pp. 33–42.

[24] A. Angelidis and K. Singh, "Space deformations and their application to shape modeling," in *ACM SIGGRAPH 2006 Courses*. ACM, 2006, pp. 10–29.

[25] A. H. Barr, "Global and local deformations of solid primitives," *SIGGRAPH '84: Proc. of the 11th Conf. on Computer Graphics and Interactive Techniques*, vol. 18, no. 3, pp. 21–30, 1984.

[26] T. W. Sederberg and S. R. Parry, "Free-form deformation of solid geometric models," *SIGGRAPH '86: Proc. of the Conf. on Computer Graphics and Interactive Techniques*, vol. 20, no. 4, pp. 151–160, 1986.

[27] P. Rademacher, "View-dependent geometry," in *SIGGRAPH '99: Proc. of the 26th Conf. on Computer Graphics and Interactive Techniques*. ACM/Addison-Wesley Publishing Co., 1999, pp. 439–446.

[28] C. D. Correa and D. Silver, "Programmable shaders for deformation rendering," in *GH '07: Proc. of the 22nd ACM SIGGRAPH/EUROGRAPHICS Symp. on Graphics hardware*. Eurographics Association, 2007, pp. 89–96.

[29] S. Schein, E. Karpen, and G. Elber, "Real-time geometric deformation displacement maps using programmable hardware," *The Visual Computer*, vol. 21, no. 8, pp. 791–800, 2005.

[30] Y. Kurzion and R. Yagel, "Interactive space deformation with hardware-assisted rendering," *IEEE Computer Graphics and Applications*, vol. 17, no. 5, pp. 66–77, 1997.

[31] M. Spindler, M. Bubke, T. Germer, and T. Strothotte, "Camera textures," in *GRAPHITE '06: Proc. of the 4th international Conf. on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia*. ACM, 2006, pp. 295–302.

[32] T. Porter and T. Duff, "Compositing digital images," in *SIGGRAPH '84: Proc. of the 11th Conf. on Computer Graphics and Interactive Techniques*. ACM, 1984, pp. 253–259.

[33] C. Gutwin, "Improving focus targeting in interactive fisheye views," in *CHI '02: Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*. ACM, 2002, pp. 267–274.

[34] R. Stockli, E. Vermote, N. Saleous, R. Simmon, and D. Herring, "The Blue Marble Next Generation - A true color earth dataset including seasonal dynamics from MODIS," Published by the NASA Earth Observatory, 2005.

[35] E. Pietriga, C. Appert, and M. Beaudouin-Lafon, "Pointing and Beyond: an Operationalization and Preliminary Evaluation of Multi-scale Searching," in *CHI '07: Proc. Human Factors in Computing Systems*. ACM, 2007, pp. 1215–1224.

[36] ISO, "9241-9 Ergonomic requirements for office work with visual display terminals (VDTs)-Part 9: Requirements for non-keyboard input devices," *International Organization for Standardization*, 2000.

[37] Y. Guiard and M. Beaudouin-Lafon, "Target acquisition in multiscale electronic worlds," *Int. J. Hum.-Comput. Stud.*, vol. 61, no. 6, pp. 875–905, Dec. 2004.

[38] H. Lieberman, "Powers of ten thousand: navigating in large information spaces," in *UIST '94: Proc. ACM Symp. on User Interface Software and Technology*. ACM, 1994, pp. 15–16.

[39] D. A. Cox, J. S. Chugh, C. Gutwin, and S. Greenberg, "The usability of transparent overview layers," in *CHI 98 Conf. summary on Human factors in computing systems*. ACM, 1998, pp. 301–302.

[40] B. L. Harrison, H. Ishii, K. J. Vicente, and W. A. S. Buxton, "Transparent layered user interfaces: an evaluation of a display design to enhance focused and divided attention," in *CHI '95: Proc. Human Factors in Computing Systems*. ACM/Addison-Wesley Publishing Co., 1995, pp. 317–324.

[41] E. A. Bier, M. C. Stone, K. Pier, W. Buxton, and T. D. DeRose, "Toolglass and magic lenses: the see-through interface," in *SIGGRAPH '93: Proc. of the 20th Conf. on Computer Graphics and Interactive Techniques*. ACM, 1993, pp. 73–80.

[42] L. Williams, "Pyramidal parametrics," *SIGGRAPH Comput. Graph.*, vol. 17, no. 3, pp. 1–11, 1983.

Emmanuel Pietriga received a PhD degree in computer science from Institut National Polytechnique de Grenoble (France) in 2002. He worked for INRIA and Xerox Research Centre Europe, did his postdoctoral research at MIT (Boston, MA) as a team member of the World Wide Web Consortium, and is now back at INRIA as a research scientist. He works on interaction techniques and tools for interactive structured graphics environments such as multi-scale user interfaces.

Olivier Bau received a M.S. degree in computer science from Université Joseph Fournier (Grenoble, France) in 2006. He is currently a PhD candidate in the Department of Computer Science at Université Paris-Sud (France). His research interests include human-computer interaction, computer graphics and design.

Caroline Appert received a PhD degree in computer science from Université Paris-Sud (France) in 2007 and did her postdoctoral research at IBM Almaden (CA, USA). She is now a permanent research scientist at CNRS (France). Her past and current work focuses on the design of tools for programming and evaluating advanced interaction techniques.