Perception, 2014, volume 43, pages 873 – 880

doi:10.1068/p7708

Visual illusions on the Internet: 15 years of change in technology and user behaviour

Michael Bach
Eye Center, Freiburg University, Killianstrasse 5, 79106 Freiburg, Germany;
e-mail: michael.bach@uni-freiburg.de
Received 3 February 2014, in revised form 12 March 2014, published online 30 April 2014

Abstract. Looking back over 15 years of demonstrating visual phenomena and optical illusions on the Internet, I will discuss two major topics. The first concerns the methodology used to present interactive visual experiments on the web, with respect to (a) wide accessibility, ie independent of browser and platform, (b) capable and elegant graphic user interface, (c) sufficient computational power, (d) ease of development and, finally, (e) future-proofing in an ever-changing online environment. The second major topic addresses some aspects of user behaviour, mainly temporal patterns (eg changes over weeks, years, long term), which reveal that there are more visitors during office hours.

Keywords: visual illusions, computer animation, Internet

1 Introduction

The first vision science book I encountered was Tom Cornsweet's (1970) *Visual Perception*. Among the beautiful experiments featured in that book was the Craik–Cornsweet–O'Brien illusion. That enamoured me to vision research. While I pursued my PhD thesis, recording from the primate visual system, I kept an eye on illusions and wanted to utilize computers to create demo experiments—at that time hopeless. The first machine that could do it well was the Mac. Borrowing one, I wrote an interactive Müller-Lyer illusion program. But the machine was too expensive for me. Many years later—skipping several steps here—the arrival of simple animation techniques in Internet browsers finally allowed me to build an online collection (Bach, 2014), starting in 1997. The goal was to provide small 'online experiments' which combined entertainment with education. At present, the collection comprises 109 pages and draws 2000 visitors every day, allowing me to look back at my experiences over 15 years with animation techniques and at user behaviour and user responses to the 'Visual Phenomena and Optical Illusions' site.

2 Technology—computers and Internet

2.1 *Animation techniques*

To reach across a wide spectrum of visitors from the idle clicker to the scientist, some minimum requirements must be met:
(1) The visual effect must work on the user's browser.
(2) It must load rapidly—longer than 2 s is already too long for some.
(3) It must be immediately visually appealing.
(4) It should allow interesting interaction and immediately respond to user actions.
(5) It should be accompanied by at least minimal explanation—the correct name, the origin, the mechanism (if known, and explicable in that context).
On the Internet, everything is in flux. Requirement 1 above may have been met at a particular time, but then lost with progress. Examples include use of Java Applets (Java applet, 2014) (now discouraged and often completely switched off because of security concerns) and Flash (Adobe Flash, 2014) [now frowned upon and not available in iOS devices (eg iPad), and not in future Android versions; more below].

Requirements 1–3 narrow down the possible animation methods. Table 1 gives a simplified overview of techniques I have employed.

Many visual phenomena depend on precise timing of visual events—for example, masking. This is particularly difficult to control in browsers, where temporal aliasing just waits to annoy. At worst, three concurrent timelines run asynchronously: the frame rate of the monitor, the refresh timing of the browser, and the time scale of the internal model. These problems were markedly improved after Mozilla.org came up with 'window.requestAnimationFrame()', 2011). This instantiates a call-back, which can alert one's program that a display refresh is about to occur, and at what precise time (with a precision better than milliseconds). This still does not guarantee smooth animation, because other processes may intervene, but improves the situation substantially. This technique is now available in all major browsers and is standardized (W3C, 2013).

Table 1. Animation techniques employed. The examples refer to pages in http://michaelbach.de/ot/.

Type	Mechanism	Example	Pros	Cons
Static display	user moves (eyes or body)	snake illusion, Pinna–Brelstaff	simple	often not appealing
Image rollover	user places mouse over image	Poggendorff illusion	simple and effective	only trivial changes possible
Animated GIF	movie like	blind spot demo	fast loading	poor colour gamut, no interaction
Movie	animation	saccadic suppression	vivid	large file size
Flash/ActionScript (2.0, 3.0)	anything possible	White's illusion	anything possible	major coding requirements, not future-proof
JavaScript, directly	anything possible	Harmon–Julesz block face	future-proof	major coding requirements
JavaScript, with FrameWorks, eg Cappuccino	anything possible	motion-induced blindness	coding optimized, future-proof	loading time

2.2 *Animation implementation and the user interface*

Simple animations with image rollovers have a trivial user interface: one just needs to advise "place your mouse over the image". But this is neither obvious nor elegant. Ideally, the full gamut of current universal graphical user interfaces should be usable where needed—for example, buttons, text fields, check boxes, radio buttons, sliders, steppers, or pop-up menus. This need was met with Flash, which I use as the basis for the Freiburg Vision Test "FrACT" (Bach, 1996, 2007). Using Flash, combined with its internal programming language ActionScript, everything was possible, and consequently, over 60 demonstrations on my site were built in Flash. Disadvantages include (1) inelegant programming environment (in my opinion), (2) a major language overhaul every 4 years or so—necessitating extensive revision—and (3) the killer: it is not future-proof. Flash has never been available on iOS devices, is flaky on Android and will lose support on that platform. Since there is a major paradigm shift from PCs to mobile devices under way (see section "User behaviour"), Flash-based content will lose most of its audience.

For a long time I considered various alternatives to Flash, and the one solution that solved nearly all problems is the 'Cappuccino Framework' (Modern App Development for the Web, 2010). This is a framework, entirely in JavaScript and thus running on all modern browsers and

platforms, that embodies a programming language 'Objective-J' and a graphical user interface modelled after Apple's Mac OS. Objective-J is virtually identical to Objective-C, the language for Mac OS and iOS, which I am used to anyway. One of its major advantages for me is its use of named parameters (Named parameter, 2014), which allows self-documenting programs. As an example, the method I wrote to draw an outline star with 5 vertices is called like this:

```
[self strokeStarAtX: 0 y: 0 nPoints: 5 radiusInner: 10 radiusOuter: 30];
```

where the brackets [] enclose the method call, the colons delineate the parameter names (eg 'y:'), followed by the parameter values, delimited by blanks.

In C, C++, Java, or JavaScript the same call would be coded thus:

```
strokeStar(0, 0, 5, 10, 30);
```

leaving the code maintainer lost as to the meaning of the parameters without extensive comments.

A major advantage is that Xcode (2014) with its Interface Builder can be used as the development environment for Cappuccino. Thus one can design the visuals of the program interactively; place views, buttons, and all other interface elements; and then connect them with their code partners. Finally, 'bindings' are supported. This means that—by making one connection entry between, say, a slider and an internal variable—the two are bound both ways: moving the slider automatically changes the corresponding variable, and changes of the variable changes position of the slider. These automatic relations remove a lot of clutter from the code, sometimes reducing it by a factor of 3. Cappuccino is open source and free. Of course, there are also disadvantages. (1) Slow load times, because much of the entire framework needs to be downloaded (this has been alleviated with server-side compression, but is still slower than Flash). (2) Small developer community: the core group is rather small, thus progress, while steady, is slow. There is great camaraderie in such a small group, and my occasional problems have been met with rapid support; the parts of Cappuccino I currently use now seem to be free of bugs. (3) Installation was difficult in 2012—marked improvements have been made since; but judging from questions on the mailing list, newcomers still have a hard time. A major overhaul of Cappuccino will improve on this.

All in all, I am very happy with Cappuccino. At the time of writing this paper, 40 pages on my site employ Cappuccino, most converted from Flash. Work effort is about half compared with Flash, and the results run on tablets (with a few speed limitations). Cappuccino is future-proof in the sense that it uses only JavaScript and HTML5 capabilities. Thus, I expect at least 10 years' lifetime. But it does need continuous effort by the open-source community to improve, and keep in step with Xcode changes.

3 User behaviour

3.1 *Visitor feedback*

From early on I received lots of kind, supportive, and helpful emails about these pages. By now the number of emails exceeds 3500, not counting my replies. Visitor feedback has also been highly positive over the 15 years of experience. The audience ranges widely: there are school children or students seeking advice for presentations. Then there are enthused laypeople, who wish to thank me, offer explanations, or suggest language corrections. Finally, there are vision scientists suggesting improvements and additions. One of these stands out especially for me: on 22 May 2005 I received an email from Jeremy L Hinton asking "Do you accept submissions for your Visual Illusions webpage?", including an animated GIF of what is now known as 'Hinton's Lilac Chaser'. I was immediately galvanized, and added a page with a modified version of his animated GIF and a description with explanation. It took only three days for this GIF to be replicated all over the Internet, nearly always without appropriate attribution.

Now there is a Wikipedia entry on this (Lilac chaser, 2014). [The example on Wikipedia uses unnecessarily desaturated colours, as can be verified by Bach (2005), where, thanks to Cappuccino as detailed above, parameters can be interactively changed.]

3.2 *Raw data: time courses*

Figure 1 shows the number of pages added to the 'Visual Phenomena and Optical Illusions' site over time. While starting slow, the net effect is a cumulative rise of available pages.

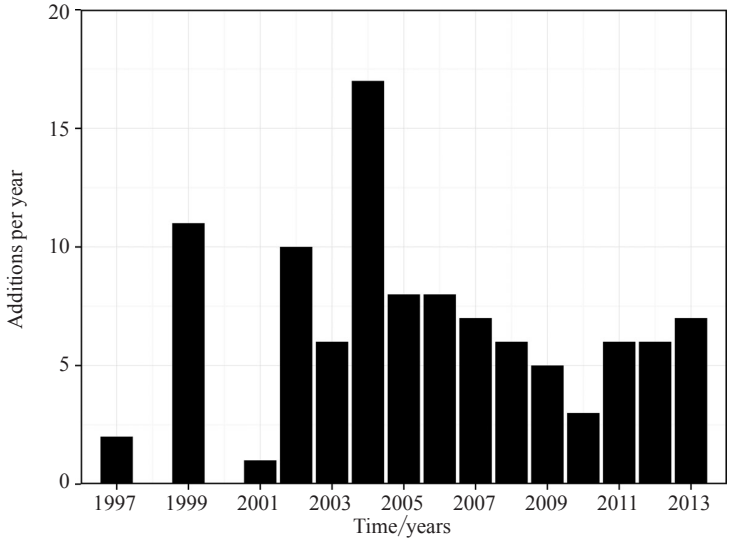

Figure 1. Pages added per year to the site 'Visual Phenomena and Optical Illusions'. It started slowly with two pages in 1997, with an average of a new page added every two months since.

Unfortunately, usage statistics from the early years are no longer available. Visitor statistics from Google Analytics, starting in November 2005, are depicted in figure 2 (top). By that time the site was already widely known and had around 10 000 unique visitors per day, or about half a million hits per day. As a 'hit' counts any access to any item, text, picture, etc, on any page, hits always suggest an excessive number, especially as the average visitor stays for over 5 pages. I will concentrate on the statistic 'unique visits' from here on. The time course in figure 2 has structure on various levels. Most obvious are the 'spikes', the highest one in autumn 2011 with 170 000 visitors on a single day (which caused my provider to shut down the site for 3 days). These spikes occur when a page goes viral on a 'social' site. The first, in mid-2006, occurred when the site was mentioned at digg.com. The more recent ones are accesses from Chinese social sites. These spikes typically subside fully within 3 days. The other major temporal structure is a marked decrease in visitors, by a factor of 3, starting in 2012, reaching a trough in summer 2013. What has caused this sizable loss of visitors? Personal communications with people running similar sites suggest that this is not a quality issue of my site, since they observe the same phenomenon. Lacking unequivocal data, I can only hypothesize that two factors play a role.

The first factor may be the rise of Facebook (suggestion by Priscilla Heard). Indeed, it is possible that 'idle clickers', who might have swerved widely years ago, now stay within their 'friends'' circles. Furthermore, traffic is usually in one direction: a picture is grabbed from outside, and then circulated within Facebook.

The other, possibly concurrent, explanation concerns the shift to mobile devices. This was started by the iPhone (2007), which was widely imitated, with marked impact beginning in 2010 to form the category 'smartphone'. The next shift involved tablet computers, pioneered by the iPad (appearing January 2010), again mimicked with about 2 years delay with competing

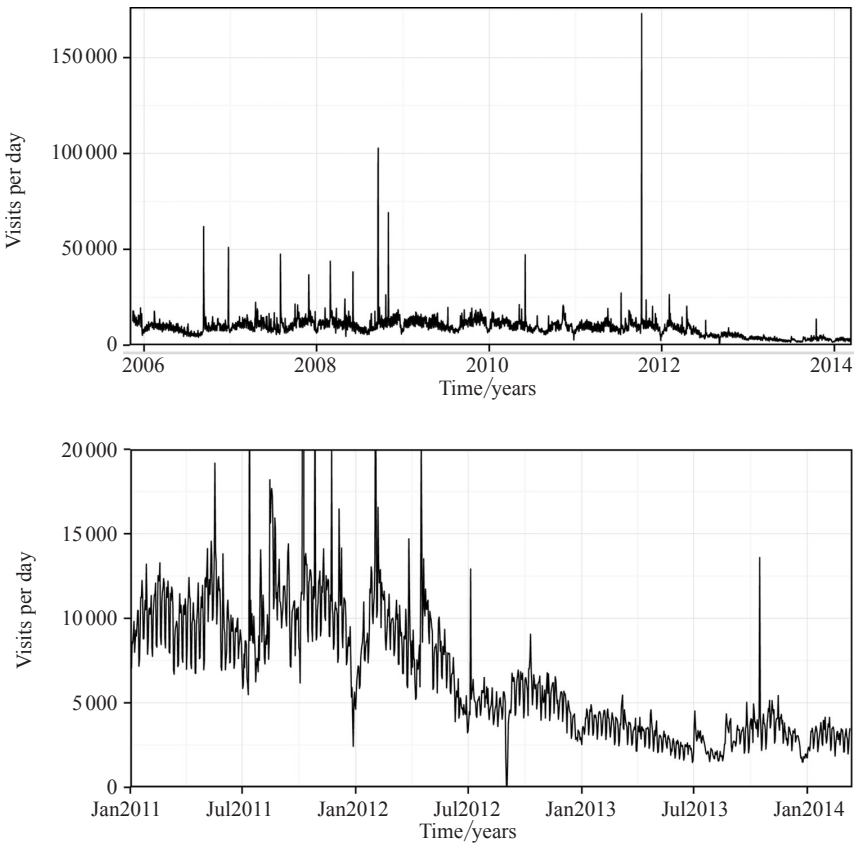

Figure 2. Visits per day. Top: covering the last 8 years; bottom: last 2 years, enlarged. The spikes represent short-lived access maxima. There is further fine structure per year (dips in the summer and in December) and per week (visible in the enlarged version; see also figure 5), and a marked downward trend since 2012, possibly reversed over the last half year.

systems, based mainly on Android. The global cumulative number of tablets is as follows: end of 2010: 16×10^6; end of 2011: 76×10^6; end of 2012: 184×10^6; and end of 2013: 327×10^6 (http://techcrunch.com/2013/08/06/forrester-tablets/). The iPhone/iPad operating system, now called iOS, never rendered Flash content. On Android devices Flash access was flaky and will be discontinued. Thus on mobile devices, especially tablets, most pages of my site did not show those demonstrations at all. This could explain part of the decline in visitor numbers beginning in 2012 (figure 2, bottom). My ongoing migration from Flash to Cappuccino, which works on smartphones and iPads, might explain the possibility of a slow recovery in the last half year and thus confirm the above hypothesis. Remaining cyclic fine structures of the time course will be considered in the next section.

3.3 *Temporal access details*

To gain further insight into the fine structure of the visitors' time series, a discrete Fourier analysis was applied [all analysis was performed with R (R Development Core Team, 2006)]. To represent yearly and weekly cycles without overspill (Bach & Meigen, 1999), the end was set at 16 November 2013, so exactly 8 years are covered. Clear peaks emerge at the frequencies corresponding to 1, 0.5, and 0.25 years; none to months; a weekly peak is the most prominent feature (figure 3).

The yearly and half-yearly peaks are better understood when the time series is averaged in monthly bins (figure 4). Here it becomes obvious that there are two marked dips in visitors per year: one during northern hemisphere summer (vacation time) and another around the

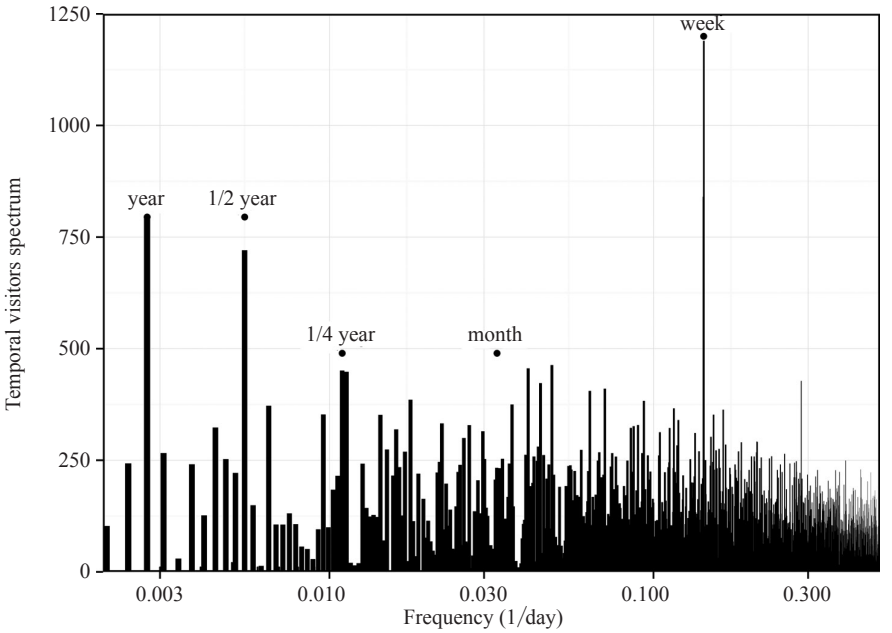

Figure 3. The frequency spectrum of daily visitors as derived via discrete Fourier transform of the time series depicted in figure 2. The spectrum displays a marked peak at 1 and 0.5 years; there is no structure at monthly intervals, but a most pronounced peak at weekly intervals.

Christmas holidays. This structure is highly significant, as obvious from the fact that the notches (representing the 95% confidence interval for the median) do not overlap between, for example, May and July. This structure suggests that there are more visitors during northern nonvacation time. This interpretation is supported by access analysis over a weekly cycle (figure 5).

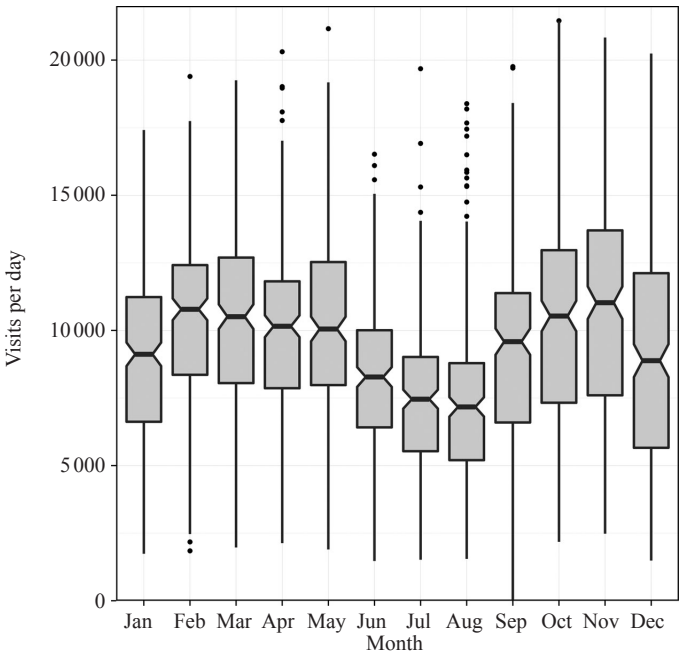

Figure 4. Mean visits per day arranged by month. (Box-plot details: the median is indicated by the thick horizontal lines, the notches represent a 95% confidence interval for the medians, the box covers the 25%–75% percentile range, the 'antennas' indicate the range, and outliers are indicated by circles.) There is a significant drop in access twice per year: one during northern summer vacations and a second one around Christmas holidays.

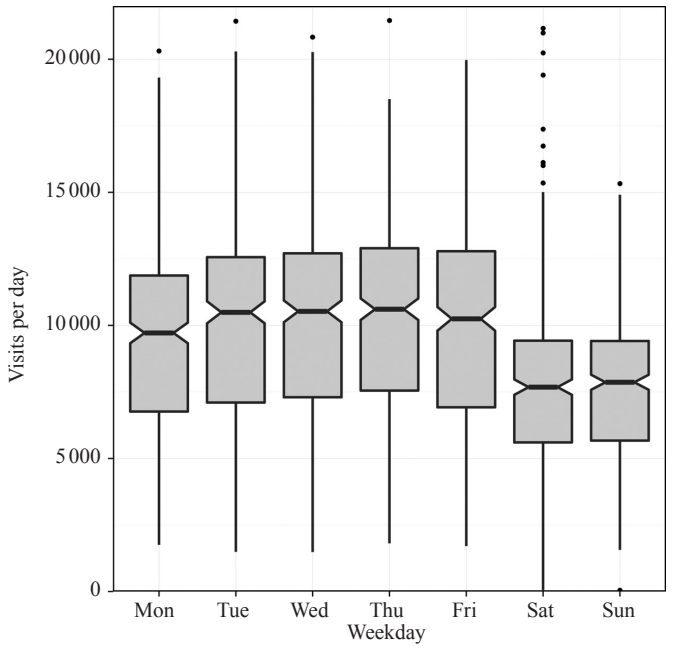

Figure 5. Mean visits per day arranged by weekday. There is a significant drop during the weekend.

4 Conclusions

User behaviour over the past 15 years (with currently around 2000 visitors every day) shows the following features: (1) short-lived (1–3 days) access spikes, multiplying normal access by a factor of 100 or more, (2) a marked decline starting in 2012 and possibly reversed since moving away from Flash, (3) yearly cycles with 2 dips of about 30% at summer and winter vacation times, and (4) a 30% dip at weekends. Visitor feedback has been highly positive, with the audience reaching from school children seeking advice for presentations, to enthused laypeople, to vision scientists.

The goal to provide 'online experiments' demands constant adaptation to the evolution in technology, given the dynamic nature of the Internet. The move from Flash-based animation to JavaScript frameworks was described in some detail, and should lead to a more successful user experience on desktop computers and mobile devices alike.

References

Adobe Flash. (2014). In *Wikipedia, the free encyclopedia*. Retrieved from http://en.wikipedia.org/w/index.php?title=Adobe_Flash&oldid=593171996

Bach, M. (1996). The Freiburg Visual Acuity Test—Automatic measurement of visual acuity. *Optometry & Vision Science*, **73**, 49–53.

Bach, M. (2005). Hinton's "Lilac Chaser". *Visual phenomena and optical illusions*. Retrieved from http://www.michaelbach.de/ot/col-lilacChaser/

Bach, M. (2007). The Freiburg Visual Acuity Test—Variability unchanged by post-hoc re-analysis. *Graefes Archive for Clinical and Experimental Ophthalmology*, **245**, 965–971.

Bach, M. (2014). *109 Visual Phenomena & Optical Illusions*. Retrieved from http://www.michaelbach.de/ot/

Bach, M., & Meigen, T. (1999). Do's and don'ts in Fourier analysis of steady-state potentials. *Documenta Ophthalmologica*, **99**, 69–82.

Cornsweet, T. N. (1970). *Visual perception*. New York: Academic Press.

Java applet. (2014). In *Wikipedia, the free encyclopedia*. Retrieved from http://en.wikipedia.org/w/index.php?title=Java_applet

Lilac chaser. (2014). In *Wikipedia, the free encyclopedia*. Retrieved from http://en.wikipedia.org/w/index.php?title=Lilac_chaser

Modern App Development for the Web. (2010). *Cappuccino*. Retrieved from http://www.cappuccino-project.org/

Named parameter. (2014). In *Wikipedia, the free encyclopedia*. Retrieved from http://en.wikipedia.org/w/index.php?title=Named_parameter

R Development Core Team. (2006). R: A Language and Environment for Statistical Computing. Retrieved from http://www.R-project.org

W3C. (2013). Timing control for script-based animations. *W3C Standards for an Open Web Platform*. Retrieved from http://www.w3.org/TR/animation-timing/

window.requestAnimationFrame( ). (2011). *Mozilla Developer Network*. Retrieved from https://developer.mozilla.org/en-US/docs/Web/API/window.requestAnimationFrame

Xcode. (2014). *Mac Dev Center*. Retrieved from https://developer.apple.com/xcode/