ABSTRACT

It is widely held that sometime around 2006, the World Wide Web as we knew it mutated into Web 2.0. This colloquial label signaled a shift from an Internet designed for us to an Internet designed by us. Nowhere was this more explicitly stated than in Time Magazine’s 2006 Person of the Year selection: You.

More than a decade later, Internet browsers have evolved into ubiquitous interfaces accessible from mobile devices, tablet computers, public kiosks, workstations, laptops, etc. It would, therefore, not be an overstatement to say that the browser is the most wide-spread content canvas in the world. Designers frequently use web browsers for their ability to exhibit and organize content. They are the sites for portfolios, announcements, magazines, and at times, discussions. But despite its flexibility and rich infrastructure, rarely is the browser used to generate design elements.

Thanks to advanced web development languages like JavaScript and open-source code libraries, such as p5.JS, Matter.JS, and Three.JS, browsers now support interactive and spatial content. Typically, these tools are used to generate gimmicks or visual effects, such as the parallax illusion or the infinite scroll. But if we perceive the browser as a time-based picture plane, we can immediately recognize its architectonic potential. This paper puts forth a method for engaging the creative potential of web-based media and Internet browsers. Through example projects, I argue that the Internet browser is a highly complex spatial plane that warrants more architectural analysis and experimentation.
INTRODUCTION

"...the Internet is not something that you just dump something on. It's not a big truck. It's a series of tubes."

In 2006, United States Senator Ted Stevens coined the above phrase in an address to Congress on the topic of net neutrality (Wired 2006). His comments were followed by a tidal wave of cultural mockery. Magazines and news outlets were quick to point out that “a series of tubes” was perhaps the most ridiculous (and inaccurate) way to describe the Internet. And yet, all jokes aside, this desire to metaphorically visualize the underlying circuitry facilitating our immaterial networks remains common practice to this day. Software and the Internet rely so heavily on metaphors that conversations on their makeup often revolve around clouds, platforms, libraries, architectures, gateways, bridges, factories, and windows. On one hand, this reliance on metaphors elevates computing to a kind of literary art, where rhetorical devices enhance its perceptual effects. But on the other hand, it often leads to dull and easily outdated clichés, such as web surfing, homepages, and information superhighways.

Is it possible, given the triteness of metaphors, to perceive these complex systems in more abstract terms? And if less metaphorical models of the Internet emerge, is it possible for designers to take advantage of this system as a medium?

We can begin to answer these questions by looking at other disciplines that have been historically well-versed in the language of abstraction. Painting, for instance, underwent a radical shift at the turn of the 20th century with artists privileging less representational modes and styles, such as Impressionism, Cubism, Suprematism, and Purism. These experiments sought to exploit the flatness of the picture plane and the material effects of paint to communicate something other than legible or accurate images of the real world. Narrowing in on one movement, specifically Russian Suprematism, we can identify a specific mode of abstraction that ties together themes of infinity, objectlessness, and flatness; immaterial ideas that are quite prevalent in the world of Internet browsers.

Because the Suprematists were fundamentally concerned with depicting immaterial ideas, we can place the Internet browser among a lineage of investigations into aesthetic problems such as the task of representing infinite space or time. In other words, because the browser operates at both the planetary and the personal scale, collapsing distance and time, it is the ideal site to address some of the cosmic questions posed by the Suprematists in the early 20th
century. Using Suprematism as a conceptual model that
eschews mimetic and perspectival techniques can help
designers reimagine the Internet as an abstract flat plane
well suited for isometric, graphic, real-time, or interactive
experiments.

In short, this paper examines the potential for Internet
space to communicate visual and spatial abstraction.
I posit that the Internet browser is a contemporary
dynamic picture plane that conflates time, space, visual
effects, and user input. If we regard it as a canvas not
for mimetic representation, but for abstract qualities,
architects and designers can use it as a new medium for
experimentation.

Along with this conceptual framework, two projects will
be described at length, which incorporate user interac-
tion, simulation, and flatness. The projects are web-based
experiments, part simulations, part drawing-apps that
exploit the mechanisms of the browser and suggest how
designers might incorporate this new canvas into their
toolset. Though these techniques make use of prevalent
open-source libraries and specific programming languages,
this is not a technical paper. Some aspects of the code will
be shared, but the goal is to look at the technical infra-
structure as something that facilitates aesthetically and
conceptually rich results. Framing this against art-his-
torical dialogues, the discussion becomes an oscillation
between engineering and aesthetics.

BACKGROUND
In his essay, “A. and Pangeometry,” Suprematist artist El
Lissitzky puts forth a concept he calls irrational space. This
space, largely derived from parallel projection drawing
methods, constitutes for Lissitzky a new medium for repre-
senting abstract and immaterial qualities. He states,

“Suprematist space may be formed not only forward from
the plane but also backward in depth. If we indicate the flat
surface of the picture as 0, we can describe the direction in
depth by – (negative) and the forward direction by + (posi-
tive), or the other way around. We see that Suprematism
has swept away from the plane the illusions of two-dimen-
sional planimetric space, the illusions of three-dimensional
perspective space, and has created the ultimate illusion of
irrational space, with its infinite extensibility into the back-
ground and foreground” (Lissitzky 1993).

This can be seen in the simultaneous effects of objects both
receding and protruding from the canvas or the illusion of
an isometric corner being both an interior and an exterior
condition (Figure 2).

Axonometric drawing was for Lissitzky the most appro-
priate mode of representing infinite space as it allowed
elements to show depth without succumbing to the distor-
tion of vanishing points. It was this relationship between
non-converging lines and the lack of a single viewpoint that
made his space irrational and ambiguous. This ambiguity
would then “force the spectator to make constant decisions
about how to interpret what he or she sees” (Bois 1988).
In a way, it established an artistic proto-interface between the viewer and the space of the canvas. The indeterminacy of background and foreground elements gave compositions a conceptual dynamism unachievable through perspectival means, which always represented the literal. As a result of Suprematism, abstraction as a means to represent that which is hard to represent was codified and tied to the orthographic picture plane.

Similarly for Kazimir Malevich, Lissitzky’s mentor and founder of Suprematism, orthographic and flat representations reduced painting to a “degree zero” from which artists could begin again to ask larger questions about the world around us (Dumpelmann 2014). His primary concern was representing what he called objectlessness, ambiguous forms that prioritized movement, dynamic states, colors, hues, fractures, or even sounds. By privileging qualities and geometric forms over literal imagery, Suprematist compositions dismantled the hegemony of traditional painting and introduced a new graphic vocabulary. Color, for instance, was not used explicitly for a stable meaning, but used to further confuse the spatial logic of the painting, as in Malevich’s White on White (Figure 3). Suprematist paintings often implied ambiguous relationships between objects and their shadows or created indeterminate depths by clashing geometry and color, foreground and background.

The Suprematists’ struggles to represent ineffable qualities such as infinity and time serve as lessons for the medium that is the Internet browser. While the browser has historically relied on a series of metaphors to describe its position as the end node of a highly intricate World Wide Web, the language and techniques of the Suprematists can suggest a more abstract, dynamic, and perhaps more architectonic interpretation of both its structure and visual presence. For instance, the browser, as an ambiguous space in which an endless variety of media exist, can fit quite well within Lissitzky’s concept of irrational space. As we’ve seen, Lissitzky proposed this abstraction of space as an evolution of Malevich’s work on the painterly surface, pushing for Suprematist space to be “formed not only forward from the plane but also backward in depth” (Bois 1988). Browsers operate in a similar kind of shallow space generated by Hypertext Markup Language (HTML) that organizes flat media in layers. Users navigate this space by scrolling, swiping, bringing to the front, sending to the back, or even going back or forward in time to a previous page. These gestures can be likened to Lissitzky’s axonometry: the forward and backward buttons move one in the positive and negative directions, scroll bars move orthographically (and infinitely in some cases), and content is stacked on a Cascading Style Sheet (CSS) canvas.

These orthographic qualities have at times been exploited by artists to create surreal and novel effects (Connor and Dea 2019), but rarely have they been used to explore architectonic qualities, that is: volumetric or spatial effects, 3D assemblages, or material imagery. With the introduction of HTML5 and embedded languages like JavaScript, browsers now have a built-in infrastructure for handling 3D data and time-based applications. Open-source JavaScript libraries, such as p5.JS and Three.JS, are also actively being
developed for embedding dynamic 2D and 3D elements into browsers. Combining both this rich infrastructure, open-source JavaScript libraries, and the lessons of the Suprematists, it is clear that the browser can be used for architectonic experiments.

The following sections describe two architectonic projects designed primarily for Internet browsers.

METHODS

1 Design

Code is, of course, an abstraction. It is immaterial and relies heavily on mathematical approximations. But what this paper suggests is that designers can use code in conjunction with visual abstraction to produce new effects and projects for the Internet browser. In order to put into practice the aforementioned concepts, two projects were developed that combine web-based physics simulations with interactive graphics. To preserve anonymity, the projects will be referred to as Project 1 and Project 2.

Project 1 is a 2D drawing app that exists solely on the web. Conceptually, it draws heavily from early computation experiments, such as Ivan Sutherland’s Sketchpad. (Sutherland 1963). Aesthetically, it recalls the minimal compositions of the Suprematists and the software of MOS Architects (Meredith and Sample 2012). The goal was to produce an app in which users could use simple gestures to generate primitive shapes. The shapes could then interact with each other through simulated physics. Because it is browser-based, it works on both traditional and mobile operating systems.

The design of Project 1 relied heavily on user interactions and gestural logics. A big concern was how should shapes be drawn. It was decided that one simple gesture could be used to define all the primitive shapes: rectangles, circles, and triangles. Referring back to Sutherland’s logic for Sketchpad (Figure 6), the principal mechanism for Project 1 is a distance formula that calculates the stretch from an initial origin point of the shape to the shape’s radius (Figures 4 and 10). This allows users to quickly familiarize themselves with the interface and easily generate a wide range of compositions.

Once a shape is generated, it can be moved around manually by selecting any part of it, or the user can trigger a gravitational force by clicking a button labeled FORCE. Enabling gravity immediately causes all objects in the scene to fall to the bottom of the browser and stack on one another. If gravity is disabled, the objects may still be moved around and can still collide with one another. Users can get rid of shapes by grabbing them and throwing them off the top of the screen. These interactions were crucial to the project as they directly illustrate the potential of the browser as a dynamic and graphic medium (Figure 5).

Project 2 is a 3D drawing app built on the logic of Project 1. Instead of generating primitives, however, this app only allows the user to model walls of different sizes. Again, it
depends on a single gesture for generating objects. But as it is in 3D, the effects are quite different. Project 2 was designed primarily to test the qualities of web-based 3D orthographic space. Like Lissitzky’s Proun series, it plays with scale and rectilinear forms. Unlike those static pieces, however, Project 2 was designed to take advantage of the ways in which users zoom in and out of space on screens. It was important to allow users to either pinch or scroll to scale the world up or down.

Project 2 also incorporates simulated gravity, which adds an element of entropy to the environment. It is modeled after traditional 3D modeling suites such as Rhinoceros or 3D Studio Max.

2 Programming
While each project’s structure will not be covered extensively here, the source code is available on GitHub. This paper prioritizes the methods of abstraction and how they correlate to graphic and spatial effects. As such, only the code principles that facilitate this are discussed.

Project 1 is built using the p5.JS library, a lightweight collection of functions similar to Processing that allow programmers to generate graphics on a browser canvas. Its goal, as stated on the developers’ website is to “make coding accessible for artists, designers, educators, and beginners, and reinterprets this [Processing] for today’s web” (McCarthy 2019). Physics are facilitated by Matter.JS, a 2D physics engine for browsers. Both p5 and Matter work together in real-time to update the canvas with each object’s position and rotation.

Objects are defined following typical JavaScript syntax (Figure 7). Custom functions allow the app to listen for behaviors happening as a result of simulations as well as user input, such as onMousePressed. It is also important that some objects share parameters for both the graphics and the physics libraries to work in conjunction. For instance, each shape’s position and rotation must be accessible to both libraries as they are working separately, although simultaneously. Thus, each graphic object has functions that update its position and render it in real-time.

Project 2 is built using the Three.JS library, perhaps the most widespread library for rendering 3D data to a WebGL canvas. Typically used for interactive video games or dynamic illustrations, Three.JS is a robust collection of functions, classes, and objects that enable users to generate meshes, textures, lights, and scenes simulating 3D space.

Three.JS syntax is modular and relatively intuitive. For instance, to create scenes, lights, or shapes, we need only instantiate each of aptly-named classes, such as THREE.Scene() or THREE.DirectionalLight() (Figure 8). This instantiation allows us to focus on interaction, as the creation of objects is fairly straightforward. Interactive functions are custom-built to receive data from the user and parse it through built-in operators from Three.JS.
In the case of Project 2, for example, we can access the physics library’s gravity and turn it off if a user presses a key, or we can make a line disappear after a user lets go of a button (Figure 9).

Both Project 1 and Project 2 are rendered to an HTML canvas using the React.JS framework. React provides an efficient and lightweight workflow for packaging and referencing external open-source libraries in JavaScript projects. React also allows JavaScript modules to be reused easily. This is a key principle of object-oriented programming of which designers can make extensive use as they begin to operate on the Internet browser.

RESULTS

Project 1 can be considered a figure-ground generator, an elevation-based stacking game, or anything in between. It directly engages the real-time calculations enabled by JavaScript as well as the user inputs of touch and mouse gestures. Neither solely an animation nor a drawing, Project 1 illustrates the new interactions afforded by today’s browser space.

Project 2 is decidedly more conceptually driven than its predecessor. For example, it can produce neither a clear figure-ground nor a fixed composition. Instead it prioritizes orthographic effects such as layering and visual ambiguity. Nevertheless, as an interface it operates like traditional 3D software and, at the same time, exaggerates some of its properties such as scale and physics.

Both projects should be understood as software instruments. Following the tradition of architects building drawing machines or custom robots, these incredibly limited tools can influence both fabrication of artifacts and intellectual reflection on those artifacts. Fabrication could be engaged by regarding browsers as potential interfaces for real-time 3D model updates, such as those used in BIM construction management. Cloud-based BIM software uses similar technology, often relying on WebGL applications, to communicate 3D architectural/mechanical/structural design coordination. Instead of developing proprietary apps, our techniques allow the industry to use browser-based interfaces for this coordination.

On a deeper level, browser-based software can also enable spectators to reflect on the gestures used to generate forms and shapes. While the visual artifacts can be compelling, immaterial effects—such as gravitational forces or the fact that each software can be opened in multiple tabs and devices at once—may suggest new, deeply digital, design behaviors and effects (Figure 12). For instance, opening multiple instances of Project 1 or Project 2 forces the user’s graphics card and CPU to work harder, increasing fan speed, and thus engendering very physical environmental repercussions. The effect of hundreds of open tabs in an Internet browser can be comical, and yet it is a new paradigm of flat visual effects and physical resources that are rarely addressed in architectural discussions. Not only does this support the theories of the Suprematists by illustrating how data can be simultaneously material and immaterial, but also it foregrounds ideas of multitasking, device communication, and real-time feedback—behaviors inextricably linked to the digital realm.

REFLECTION

In his book, The Interface Effect, Alexander Galloway describes object-oriented programming as a “metaphysico-Platonic logic.” He states that “objects are instances of classes, they are created in the image of a class, they persist for finite amounts of time and eventually are destroyed” (Galloway 2012). This quasi-ontological premise, while metaphorical, is the overarching logic of networked systems. That software enables an infinite amount of copies of any data to be quickly dispersed brings up questions of authority, authenticity, and a slew of other philosophical quandaries.

In the case of Project 1 and Project 2 (Figures 1 and 11), modularity and packaging of code were crucial to the development of their structure. Without these libraries, the graphic and physics functions would have to be developed from scratch—an incredibly difficult and
time-consuming task. In other words, the open-source nature of these systems allows code to be used by a much wider demographic. Architects need only follow online tutorials or basic coding principles to be able to engage with the drawing mechanisms of p5.JS or Three.JS. As such, JavaScript libraries can open up the world of Internet browsers to designers.

Beyond accessibility, the browser today offers many more potential uses as a medium. If the introduction of digital video in the 1980s was considered a radical paradigm shift in the visual arts, the browser is clearly following suit. Browsers allow for interaction with a variety of input devices from the mouse to the finger, which can be used to practical and impractical ends. The projects presented here focus largely on the latter, as they are not designed as specific problem-solving tools but rather as design experiments. Like most experiments in computational design, they prioritize accidents that might be scrutinized and dissected afterwards.

One of those effects could be the radical change of scale from a desktop interface to a mobile screen. Having designed each project for both, we are able to gauge the difference in drawing on a large tablet with the same gestures as drawing on a handheld phone. This also brings up the issue of computing power and its variation across different devices. After showing this app to different users, we became aware of certain limitations, such as the inability to download screenshots on some mobile devices, incompatible browsers, etc. This was not something that could have been explored if the focus was solely making a desktop application.

What JavaScript libraries allow for is an accessible way to explore the intricacies of the Internet browser. Code is not simply a way of putting content on the web, but it can be a vehicle for dissecting the new behaviors engendered by it. Project 1 and Project 2 are in effect diagrams of these behaviors, drawing apps for touch culture or passive games for distracted users.

CONCLUSION
In “Lessons of the Russian Avant-Garde,” historian Catherine Cooke critiques the architectural Deconstructionist movement for its purely visual reading and appropriation of Suprematist compositions. Deconstruction was, of course, the proto-digital movement concerned with visual instability that branded itself as a metonymic offspring of deconstructionist literary theory and constructivist art. But in her critique, Cooke points out that much of the architectural work missed the underlying themes of the Russian avant-garde: mysticism, infinity, dematerialization, and objectlessness. Had architects studied the movements closer, they would have realized that, “[Suprematism’s] very otherness...provides a paradigm of a space-time universe, which is, naturally and logically, appropriate to the new perceptions of how the cognitive and phenomenal world of the late 20th century is operating” (Cooke 1988). And so for Cooke, the lessons that should be
taken away from the Russian avant-garde are, therefore, not primarily visual at all, but rather deeply cosmological.

If we extend Cooke’s thesis to the digital world, arguably a cosmos in itself, we can recognize similar immaterial properties in the ubiquitous computing environments that surround us; not necessarily as graphic visual elements, but as gestures and instincts tied to software interactions. The pervasiveness of software has engendered a new consciousness that blurs the distinction between virtual and physical objects and creates a tension similar to that of off-white and pure-white in Malevich’s White on White; or what Anna Neimark refers to as the real-time oscillation of figure-ground, figure-figure, and ground-ground (Neimark 2014).

This instability between the virtual and the actual should come as no surprise to architects and designers, whose task is primarily negotiating between the two realms (from design to building, between representations and instrumental directives). But for some, the digital world is still regarded as a virtual analog to the real world. The methodologies and projects outlined here seek to counter this claim and introduce skepticism of any digital technologies claiming to be purely immaterial. Furthermore, the Internet today has evolved into a much more complex and metaphysical environment than it was just a decade ago. This warrants new spatial conceptions. It has enabled a new digital consciousness that manifests itself as an extension of ourselves either through physical gestures that manipulate virtual things or psychological reactions to those virtual objects. Instantly accessible web-based canvases could be used to represent simulations of existing spaces, instructions for creating new purely ephemeral spaces or even non-Euclidean spaces. It is thus possible that engaging this medium might teach us something about space-making that we have yet to explore. Conversely, it is also possible to use architectonic thinking to reinterpret how the space of the web functions, not as metaphors but as a form of knowledge. By bringing in historical discourses on flatness and pictorial space, we might develop a more aesthetic understanding of the Internet as a medium (in addition to its social, and technical presence). We can understand infinite voids, abstract environments, and geometric behaviors not only as terms related to software and computing, but also as Suprematist modes of depicting space put forth by Malevich and El Lissitzky. Using the mysticism of the Suprematists and conceiving of browser space as an ever-changing, infinitely deep orthographic space can be a productive way to advance our perception of these virtual realities.

12 ThreeJS “after-image” effect, an example of a borne-digital visual effect
REFERENCES


IMAGE CREDITS
All drawings and images by the author.

Galo Cañizares is a designer, writer, and a 2018-19 Christos Yessios Visiting Professor at the Knowlton School where he was the school’s 2016-17 Howard E. LeFevre ’29 Emerging Practitioner Fellow. He holds a Master of Architecture from the Massachusetts Institute of Technology and a Bachelor of Environmental Design from the College of Architecture and Planning at the University of Colorado, Boulder. In 2014, Cañizares founded office ca, a research and design collaborative that investigates alternate methods for architectural practice. Some recurring themes of the practice include absurdity, genre fiction, conspiracy theories, world-making, storytelling, simulation, and most recently, the parafictional. He is the author of Digital Fabrications: Designer Stories for a Software-Based Planet, a collection of essays on software and design forthcoming from Applied Research + Design Publishing.