Color Me Surprised: An Interactive Color Picker for UI/UX Designers

EDGAR ONOFRE, University of Washington, USA
VINCENT VAN DER MEULEN, University of Washington, USA

At SIGGRAPH 2017, a team of Adobe researchers presented Playful Palette: a color picker for digital artists that takes cue from fine art. Playful Palette enables digital artists to mix colors in the same way they would using traditional media. To some, innovation in the creative tools industry might come as no surprise. After all, continued interest in artificial intelligence and machine learning has sparked conversations about the extent to which technology can help, or even replace, us. However, UI/UX design tools seem to be evolving at a slower rate than their digital media counterparts. This could be worrisome considering that the role of UI/UX designer is growing simultaneously in importance. In the following paper, we investigate whether we can use the ideas of Playful Palette to make UI designers more creative and effective. We implement a basic version of Playful Palette using web technologies and conduct user tests. Our results (N = 3) indicate that integrating Playful Palette into modern UI/UX design tools could benefit UI/UX designers. However, more user studies as well as explorations of alternatives are needed to conclude that our work could make designers more creative and effective.

CCS Concepts: • **Human-centered computing → Graphical user interfaces**; Gestural input; • **Applied computing → Media arts**;

Additional Key Words and Phrases: creativity support, user interface design, color, parametric color gamut representation, interactive interfaces, playful palette, sketch app

1 INTRODUCTION

SIGGRAPH 2017 saw the introduction of a new digital color picker by Adobe Research, namely Playful Palette [Shugrina et al. 2017]. Playful Palette draws on traditional media and lets digital designers explore colors by blending them on a mixing dish. On top of this, it provides artists with features unique to the digital medium, such as the ability to both view and manipulate their used colors.

A demo of Playful Palette was shown to the general public at the 2017 installment of Adobe's [2017] annual conference, Adobe Max. At that same conference, Adobe [[n. d.]] announced that Adobe Experience Design, or XD in short, was officially out of beta. Together with Sketch, Photoshop, Illustrator, Figma, and soon InVision Studio, Adobe XD is the leading tool for User Experience (UX) and User Interface (UI) design [UX Tools [n. d.]]. All of the aforementioned companies are rapidly iterating on their UI/UX design tools but, at least in the short-term, seem to be converging to two main features. These features are 1) the ability to design interfaces using an Adobe Photoshop-like interface and 2) the ability to make said interface into a clickable prototype, with various levels of fidelity.

For better or for worse, the developments in UI/UX design tools, which are primarily focused on enhancing existing features, appear to be less radical and future-facing than (Adobe's) developments in digital media tools. Considering that the role of UI/UX designer is growing in importance, it is worth asking ourselves if we can also make the lives of UI/UX designers easier by promoting their tools to assistants [Maeda 2018]. Explorations like this, that focus on taking work out of the hands of the designer, are a natural progression of the idea that computers can do certain arduous tasks for humans, which is regaining interest as artificial intelligence is becoming more popular. Notably, "machine intelligence" is also what design experts like former Rhode Island School of Design (RISD) president John Maeda [2018] think design tools are headed towards.

While Playful Palette likely does not qualify as machine intelligence, the concept is interesting because it takes the time intensive yet crucial task of color selection and makes it significantly easier and more fun. Picking colors is equally as hard and important for UI/UX designers. Hence, in the following paper, we investigate whether we can make UI/UX designers more effective and creative by integrating Playful Palette into modern design tools. Our short-term goal is to make the process of color selection simpler and more enjoyable. UI/UX designers might then be able to save both time and energy for other important tasks. Long-term, we hope that our work sparks a debate about the relationship between UI/UX designers and their tools, and about what UI/UX design tools currently are versus what they can be.

2 RELATED WORK

Perhaps the closest that one can come to an "existing" UI/UX design tool integration of Playful Palette, is an abandoned patent application by Figma from 2014 [Wallace and Field 2014]. Presently, Figma has a standard color picker. However, three years before Adobe published its research, Figma was exploring what is essentially the UI/UX design equivalent of Playful Palette. For the sake of time and the experiment, we decided not to deviate too much from Playful Palette. Moreover, we only found out about Figma's patent long after we had finished conducting our user study. Nonetheless, it is a perfect example of how one could elegantly integrate Playful Palette into a modern design tool. Figma's color picker consist out of three components: a color preview, an HSV picker, and a color palette (see Figure 1). Its color palette section makes it similar to Playful Palette. Here, Figma automatically generates a color palette by interpolating between a set of anchors that each denote a color. Users can add, remove, and move around anchors, after which new interpolated colors are generated. In other words, the color palette in Figma's patent application is essentially Playful Palette's mixing dish.

Besides Figma's patent application, there appears to exist little research that explores alternative, and more creative ways of selecting colors in UI/UX design. The Playful Palette authors cite Kita and Miyata [2016] and Wijffelaars et al. [2008], among others. Kita and Miyata investigate why we prefer certain colors over others. They use the resulting color rating model to automatically expand color palettes. Wijffelaars et al. enable users to generate color palettes using intuitive parameters, such as number of colors and hue range.

Authors' addresses: Edgar Onofre, University of Washington, 1410 NE Campus Parkway, Seattle, WA, 98195, USA, edgaro@uw.edu; Vincent van der Meulen, University of Washington, 1410 NE Campus Parkway, Seattle, WA, 98195, USA, meulen@uw.edu.

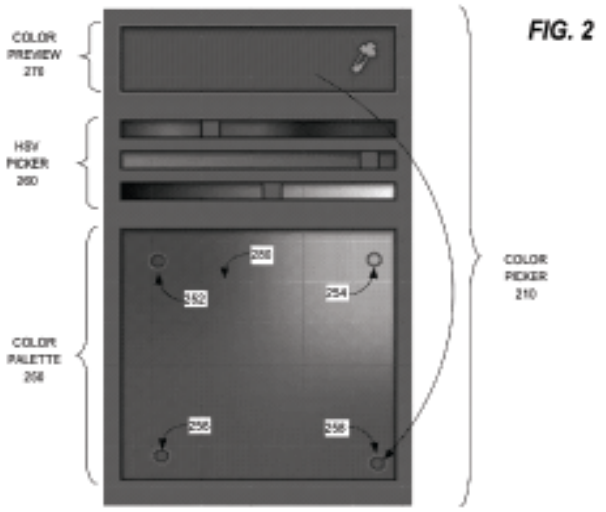

Fig. 1. Figma's patent application: Automatically generating a multi-color palette and picker.

Both the work of Kita and Miyata and Wijffelaars et al. is impressive, but does not deal directly with color selection in UI/UX design. Furthermore, it reduces the extent to which color selection is about fun and exploration, which is the opposite of what Playful Palette aims to do. Perhaps a closer approximation to our work is a color picker by Guosheng et al [2012]. Using this color picker, users can easily create harmonious color palettes. Yet again, there is no explicit focus on UI/UX design however. One could also argue that while important, harmonious color palettes are rather trivial to generate.

Because Sketch has the most open ecosystem of all the popular UI/UX design tools, we expect any existing Playful Palette implementations to be Sketch plugins. While no Sketch plugin is comparable to Playful Palette, it is worth our time to briefly discuss the Sketch color plugins that do exist. After all, they represent the ways in which plugin authors have tried to make color selection easier for UI/UX designers. Chain [2017] and Bjango Color Creator Templates [2018], both popular among Sketch users, enable users to generate palettes by enforcing relationships such as brightness between colors. Chromatic, which also appears among the first results on GitHub, takes a blob-like approach by providing the user with a color scale for any two colors [petterheterjag 2017]. It is clear that when it comes to Sketch plugins, existing solutions also focus less on the manual exploration of colors. Notably, none of the aforementioned Sketch plugins are standalone color pickers either.

3 IMPLEMENTATION

3.1 Creating Colored Metaballs

There are a variety of ways to render blobs, or metaballs. We use HTML5 Canvas and a technique described in blog posts by Loktar [2012] and Hoffman [2012]. The idea behind this technique is to make use of a hidden, temporary canvas and a visible canvas. Every time the user clicks on the visible canvas, we draw a radial gradient at the corresponding position of the user's mouse click on the temporary canvas. This radial gradient starts with the appropriate (rgba) color in its center, and slowly fades out towards its edges. Because the gradient transitions from a solid color to transparency, the circle's alpha value can be thought of as a function that produces lower values as the distance to the circle's center increases. With only one circle on the temporary canvas, this fact is of little importance. However, it means that when two radial gradients partly overlap, the shared pixels will have higher alpha values than they would have had if only one circle had existed. This observation is the key to rendering blobs.

Once we have drawn a radial gradient on the temporary canvas in response to the user, we take our temporary canvas, that at any given point in time is comprised of n radial gradients, and process it pixel by pixel. If a pixel's alpha value is at least 200 on a 0-255 scale, we transfer it to the exact same location on the visible canvas. Since the inclusion in other circle's areas can give pixels a higher alpha value, causing them to show up on the visible canvas, colored metaballs are created naturally. As can be seen in Figure 2 and Figure 3, this approach produces similar results to Playful Palette's.

3.2 Using HTML5 Canvas with React and Redux

The aforementioned technique for rendering colored metaballs only requires a HTML5 Canvas object. Nonetheless, we chose to use React and Redux to be able to a) develop our interface as a collection of components and b) easily share state across different parts of our UI. We found that there especially was a need for point b), as more than one component need to know information such as the currently selected color.

We render a Sketch-like color picker using react-color [casesandberg 2018]. The flow of information is as one would expect from a typical React/Redux app. When the user clicks on the visible canvas, we dispatch an action, a plain JavaScript object, that contains the click's coordinates and the color that was selected. A reducer then adds this point to the piece of state that describes all of our app's points, without mutating this piece. The Playful Palette component, or mixing dish, makes use of the renderProps pattern to listen for changes in the app's state [Facebook [n. d.]]. Every time the global collection of points change, Redux informs the Playful Palette component about the new set of points. The temporary canvas, and subsequently the visible canvas, rerender using the technique described in the previous subsection and the user sees new blobs. Because the Sketch color picker is also listening for changes in the points state, the user's color history gets updated automatically.

If one knows how to render a blob on user click, enabling the user to dissect and move around blobs is fairly straightforward. When the user clicks anywhere on the visible canvas, we first determine whether they are clicking on an existing blob. We know the user is clicking on an existing blob if the pixel's alpha value is greater than or equal to our predetermined alpha threshold of 200. If that is the case, all we need to do is loop over our collection of points as described by Redux, and find the point (circle center) that is closest to the mouse coordinates. Once we have determined the nearest point, we tell Redux it needs to move the point to a new

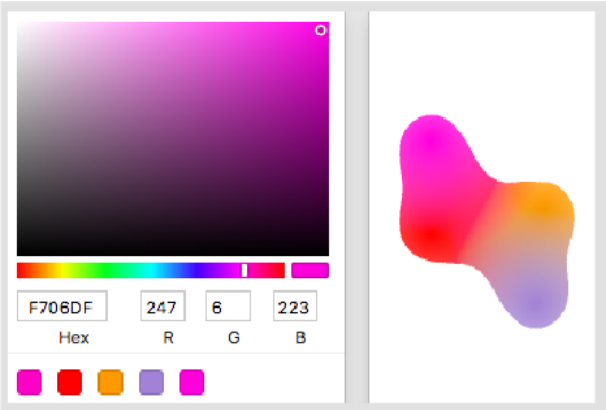

Fig. 2. Color Me Surprised's blobs.

Fig. 3. Playful Palette's blobs.

location every time the user moves their mouse, thus simulating dragging a blob. As soon as the user releases their mouse click, we stop informing Redux about changes in the mouse's position as the user is no longer dragging the blob.

4 USER STUDY

We conducted three user tests with three student designers. We asked each participant to color in a premade interface in Sketch (see Figure 4) using their choice of colors with Sketch's color picker (Task 1). We then had the participants design the same interface using the Color Me Surprised color picker (see Figure 5), which features mixing color blobs to produce the user's desired color and records a history of colors used (Task 2). The participant would create a color in Color Me Surprised using the blobs, then use the eyedropper function in Sketch to select the color and copy the hex code into Sketch's color picker for the color to take effect. Approximately 10 minutes were allocated for each task, which we switched after each participant. We asked the designers to verbalize their process for each task, point out any strengths or weaknesses, to better

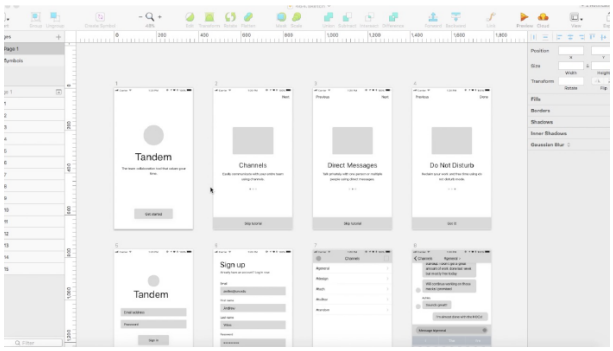

Fig. 4. Blank UI used during user testing.

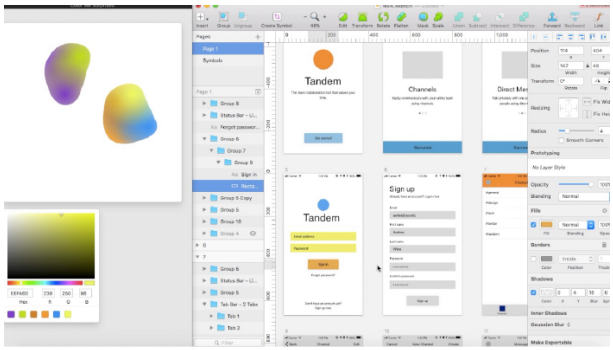

Fig. 5. UI colored in during user testing.

understand our product. We recorded some of the screens during the studies for further investigation and took notes of what we were observing.

The verbal feedback during our studies was very informative. One participant said "It was much more like mixing colors on a paint palette. I like how it keeps a history of what color I am using. It was nice being able to move around the blobs." Another participant said they like "the interactiveness instead of monochromatic of clicking." The participants had many suggestions. For example, they mentioned adding a delete function to get rid of unwanted blobs on the canvas, by either dragging it off the screen or dragging it into a corner that has a trash can icon. Next, a participant suggested a feature that would allow you to increase or decrease the size of the blob to have a stronger or weaker gradient in the mixing of colors. Additionally, another participant mentioned the idea of having a smart color picker that would be able to suggest color palettes that go well together when a user clicks on a color. Overall, all the participants said they would like to see the Color Me Surprised color picker in a modern UI/UX design tool.

5 CONCLUSION

Based on assumed designer needs, we designed Color Me Surprised, an interactive color picker interface. We introduced an effective and exciting tool for UI/UX designers that allows for color editing and provides a color history. Color Me Surprised is an application

that is designed to make it easier for UI/UX designers to combine colors in a way that is novel and visually appealing. Our user studies confirmed our interface was efficient at assisting the designer's goals and creativity.

6 DISCUSSION

While our users were excited about our color picker, there are a variety of problems with our methodology that could negatively impact our study's generalizability. First, contrary to the Playful Palette team, we did not verify that the perceived problem, color selection in UI/UX design, indeed is a problem. Second, we recruited research subjects by asking fellow design students to participate in our user study instead of reaching out to a diverse selection of participants. As a result, we introduced selection bias into our study and worked with participants that were more likely to give us favorable answers. Third, we should have worked with a higher number of participants than we did (N = 3) because our results are currently not statistically significant. Fourth, the setup of the user testing was inconvenient because users were not fond of having two windows open at the same with our color picker and the program Sketch. They would have preferred a Sketch plugin. Last, it would have been wise to follow Adobe Research in collecting quantitative feedback. That would have allowed us to perform t-tests and get a sense of our results' significance.

7 FUTURE WORK

Our work can be extended and improved upon by addressing the issues outlined in the "Discussion" section. It is our hope that, once these problems have been addressed, it turns out that our conclusions were sound and the ideas of Playful Palette can indeed be used to make UI/UX designers more creative and effective. If that is the case, a logical next step would be to turn our tool into an actual Sketch plugin. The existence of a Sketch plugin that radically changes the way designers work with color will hopefully prompt the makers of the most popular UI/UX design tools to revisit their traditional color picker. Hence, in the most favorable scenario, extending our work could be a first step towards a more helpful and supporting, new generation of UI/UX design tools.

ACKNOWLEDGMENTS

The authors would like to thank Alex Colburn, Barbara Mones, Deepali Aneja, and Gary Faigin of the University of Washington, Seattle for their help and guidance.

REFERENCES

Adobe. [n. d.]. Closure of the Adobe XD Beta program. ([n. d.]). https://helpx.adobe.com/xd/kb/closing-adobe-experience-design-beta-program.html

Adobe. 2017. PlayfulPalette: Adobe MAX 2017 (Sneak Peeks) (Sneak Peeks) | Adobe Creative Cloud. (Oct 2017). https://www.youtube.com/watch?v=bo5MM0gD6cM

Bjango. 2018. bjango/Color-Creator. (Mar 2018). https://github.com/bjango/Color-Creator

casesandberg. 2018. casesandberg/react-color. (Feb 2018). https://github.com/casesandberg/react-color

Facebook. [n. d.]. Render Props. ([n. d.]). https://reactjs.org/docs/render-props.html

Hoffman. 2012. 2D Metaballs in XNA. (Jun 2012). http://nullcandy.com/2d-metaballs-in-xna/

Guosheng Hu, Zhigeng Pan, Mingmin Zhang, De Chen, Wenzhen Yang, and Jian Chen. 2012. An interactive method for generating harmonious color schemes. *Color Research and Application* 39, 1 (Dec 2012), 70âĂŞ78. https://doi.org/10.1002/col.21762

N. Kita and K. Miyata. 2016. Aesthetic Rating and Color Suggestion for Color Palettes. *Computer Graphics Forum* 35, 7 (Oct 2016), 127âĂŞ136. https://doi.org/10.1111/cgf.13010

LaloMrtnz. 2017. LaloMrtnz/Chain. (Aug 2017). https://github.com/LaloMrtnz/Chain

Loktar. 2012. 2d Metaballs with canvas! (Jun 2012). http://www.somethinghitme.com/2012/06/06/2d-metaballs-with-canvas/

John Maeda. 2018. Design in Tech Report 2018. (Mar 2018). https://designintech.report/

petterheterjag. 2017. petterheterjag/chromatic-sketch. (Nov 2017). https://github.com/petterheterjag/chromatic-sketch

Maria Shugrina, Jingwan Lu, and Stephen Diverdi. 2017. Playful palette. *ACM Transactions on Graphics* 36, 4 (2017), 1âĂŞ10. https://doi.org/10.1145/3072959.3073690

UX Tools. [n. d.]. Design Tools Survey. ([n. d.]). https://uxtools.co/survey-2017

Evan Wallace and Dylan Field. 2014. Automatically generating a multi-color palette and picker. (Feb 2014).

Martijn Wijffelaars, Roel Vliegen, Jarke J. Van Wijk, and Erik-Jan Van Der Linden. 2008. Generating Color Palettes using Intuitive Parameters. *Computer Graphics Forum* 27, 3 (May 2008), 743âĂŞ750. https://doi.org/10.1111/j.1467-8659.2008.01203.x

Received ; revised ; final version ; accepted