Online Submission ID: 4276

$\mathbb{C}^3$-palette: Co-saliency based Colorization for Comparing Categorical Visualizations

– Supplementary Material –

Category: n/a

This supplemental material file provides additional experimental results for our submitted paper titled "$\mathbb{C}^3$-palette: Co-saliency based Colorization for Comparing Categorical Visualizations".

Scatterplots with Significant Overlap. Since our contrast measures are built on the data-space nearest neighbour graphs, its produced palettes also work well for scatterplots with significant overlap between classes. For example, as shown in the results from *Random Assignment* in Fig. 1, the light orange class is mixed with the light purple class and the teal class has a strong overlap with the grey class, they are hard to distinguish. The *Alpha Blending* result is based on *Optimized Assignment* [3] with Tableau palette, however, due to the changed opacity, classes are hard to split from each other and generated different colors. The results generated from *Palettailor* and *C^3-Palette* both have a well separability since they are all built on the data-space nearest neighbour graphs. Specifically, *C^3-Palette* highlights the changed classes(classes with green, purple and red) while maintaining discrimination of strong overlapped classes, such as the blue and yellow, red and green classes.

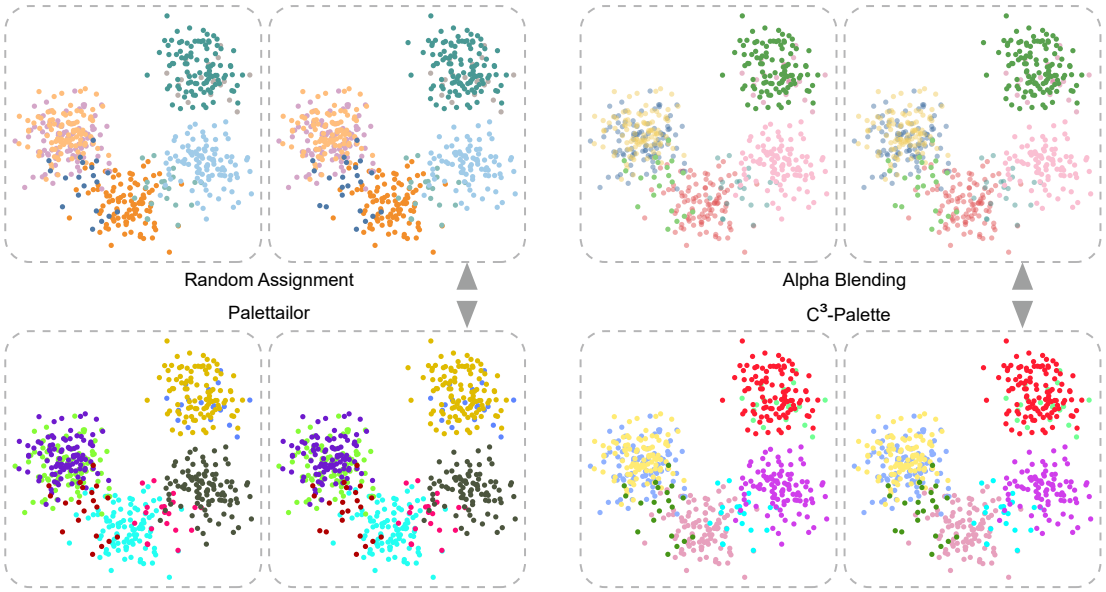

Figure 1: Results for scatterplots with significant overlap: (left) Tableau palette with random assignment versus Palettailor [2]; (right) Alpha Blending versus C^3-palette. Our co-saliency methods (right bottom) can highlight the changed classes while maintaining discrimination of strong overlapped classes.

Bad Case for Alpha Blending on Different Visualizations. Since scatterplot points would overlap, the alpha blending method will generate new colors. Here we show alpha blending is insufficient even for bar chart and line chart. Fig. 2 shows an example, where the left top two images show our auto-generated results and the left bottom are the alpha blending result based on Tableau 20 palette by maintaining the opacity of the bar with largest difference be 1.0 while changing others' opacity to 0.4. We still hard to find the largest change immediately. Similarly, this effect is existed in line chart as shown in Fig. 2 right. Due to the low contrast against the background, the yellow line is hard to pop out even the opacity of other classes set to 0.2.

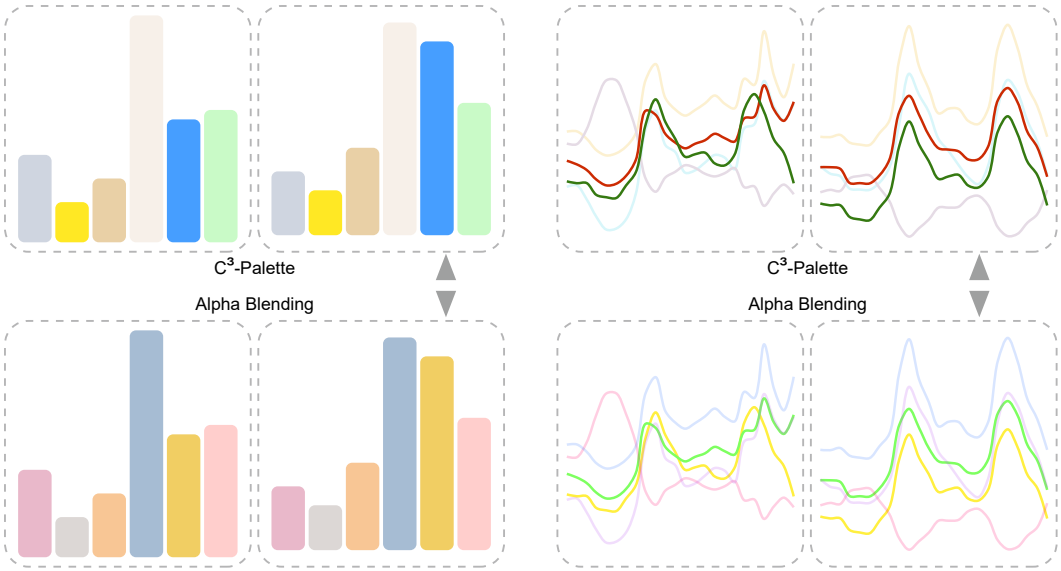

Figure 2: Illustrating the problem of alpha blending. (left) C^3-palette versus Tableau palette with alpha blending for bar charts; (right) C^3-palette versus palette contains low contrast color with alpha blending for line charts.

Pilot Study Details and Statistics. We conducted two pilot studies for scatterplot experiment, one for spotting the difference task and the other is for the counting classes number task. The statistics results are shown in Fig. 3. As for *spotting the difference* task, we recruited 28 people through the Amazon Mechanical Turk. The power analysis is executed between *C3-Palette Generation* and *Random Assignment*. With an effect size Cohen's d of 0.4, alpha level of 0.05 and beta level of 0.8, the power analysis suggested a minimum number of 100 participants for the spot-the-difference task. The procedure of *counting class number* task is similar to the previous one. We recruited 29 participants from AMT whose approval rate is larger than 97%. Since we mainly want to compare the class discriminability between *C3-Palette Generation* and *Alpha Blending*, we executed power analysis based on these two conditions. With an effect size Cohen's d of 0.6, the power analysis indicates that we need 50 participants for this task.

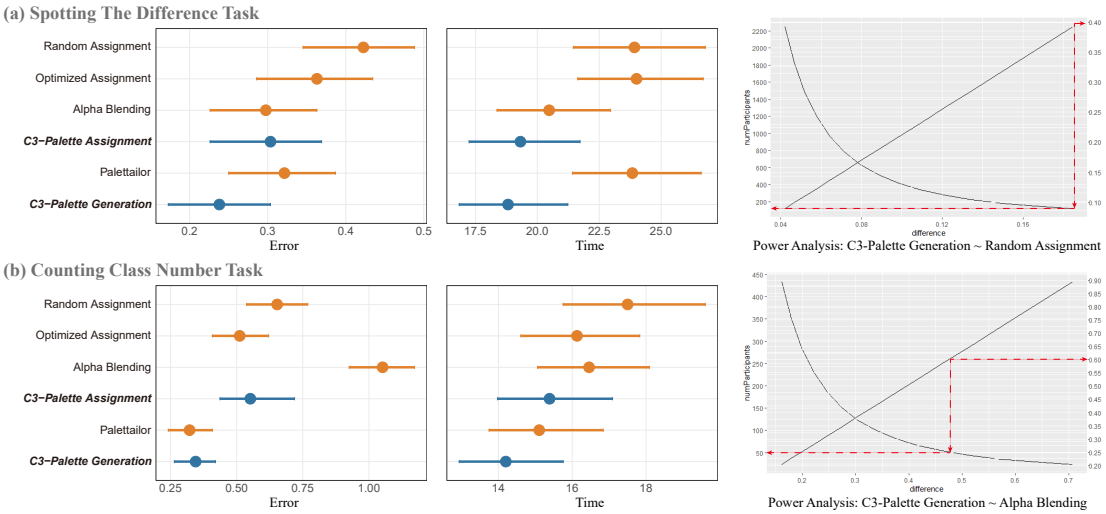

Figure 3: Confidence interval plots and power analysis for the two pilot studies.

Additional Case Study for Scatterplots. We conducted a case study with a real world data, which is well-known for the use in Gapminder [1], to evaluate the usability of our system.

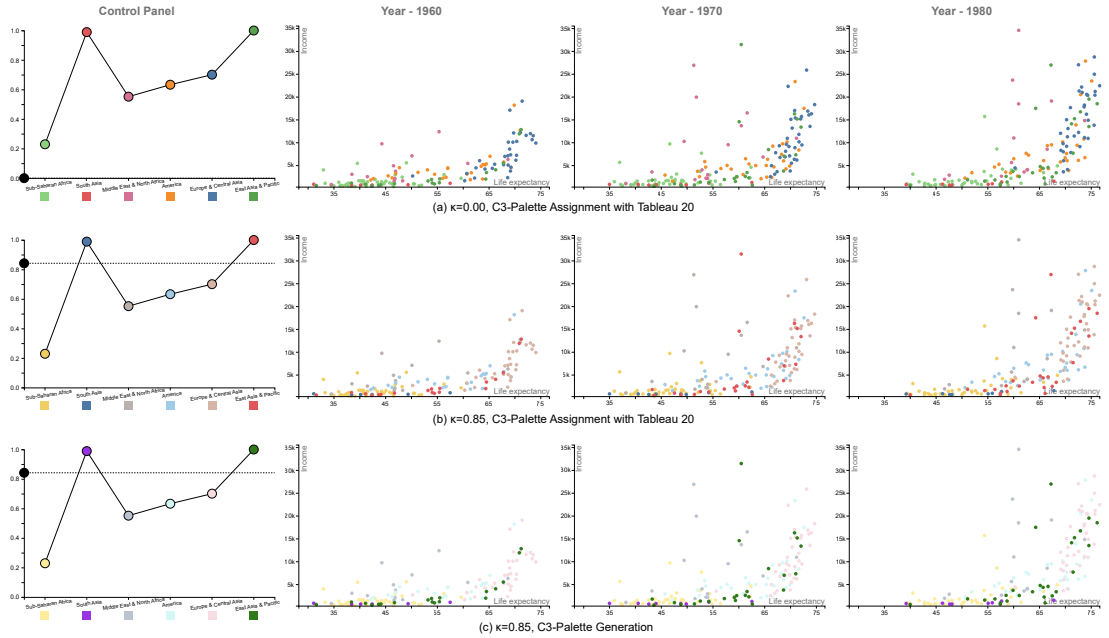

Figure 4: Gapminder dataset: (a) Result generated by default setting for given palette; (b) User-specified κ value for popping out classes; (c) Automatic palette generation for achieving a better discriminablity.

We choose life expectancy and income as the x axis and y axis, respectively. And we use world regions as the class label. As shown in Fig. 4, due to the limit space, we only show three years. And to make it easy to read, we removed the points with a much larger x value or y value. First, we used the default settings of our system to automatically produce a color assignment result based on Tableau 20 palette for assigning colors to different objects in the dataset, see Fig. 4(a). Since κ is 0 and all the classes are changed, each class is assigned with a salient color to make it more distinguishable. This result is similar to *Optimized Assignment* [3] while our result considers the different importance of classes, i.e., larger importance value has a more salient color. Then we want to explore the two classes with the largest change degree, thus we move the κ control point(the black circle in control panel) to a larger value, as shown in Fig. 4(b). Now we can see the largest changed classes more clearly. But the visual separability between the classes with lower κ value is small, such as the color of *Middle East & North Africa* and *Europe & Central Asia*. We further generate the result by our palette generation method which has a better performance on discriminability, see Fig. 4(c). Through our exploration, we found that *South Asia* should not have a large change degree. This result is caused by our default class importance measure which sets point number change a larger weight, this is done due to the previous evaluation result that point number change is harder to distinguish than point position change.

Our system also supports manually class importance adjustment, we illustrate this in Fig. 5. For example, we are interested in *America*, thus we can increase the importance value of the corresponding circle and meanwhile, decrease other classes' importance value until lower than κ. We show both assignment result for user provided palette and automatic palette generation result. It's obvious that both results highlight the interested class while palette generation method leads to a much better visual separability between different classes.

Palette Generation for Color Vision Deficiency. To help people with a color vision deficiency, we allow users to generate palettes that can be used for different types of vision problem, such as protanomaly and deuteranomaly which result in poor red-green hue discrimination. This is achieved by adopting a color blindness simulator(the source code can be find at github `https://github.com/MaPePeR/jsColorblindSimulator`) and then used our matrix for palette evaluation. Fig. 6 shows an example, where the left two images show the auto-generated results and the right are the simulated results perceived by people with protanomaly. That is, the purple and red

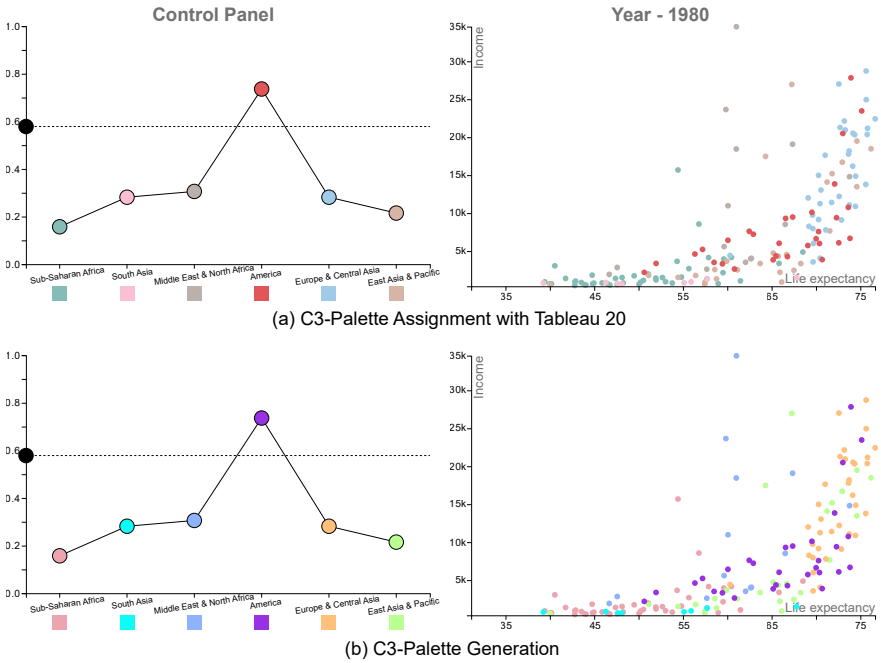

Figure 5: Manually define the class importance in the control panel: (a) Result generated based on given palette; (b) Automatic palette generation.

classes seen by normal vision will be perceived as dark blue and dark purple by protanomaly. We can see that our results are easy to find changes between scatterplots for people with color vision deficiency and it is still can be distinguished by normal vision. This preliminary result proves that our method can integrates start-of-the-art color blindness simulation algorithm to generate palettes for color vision deficiency.

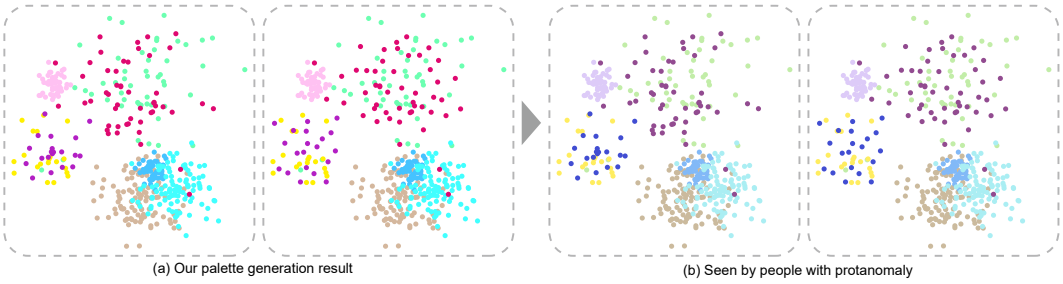

Figure 6: Exploring the ability of our system to generate palettes for both people with normal vision and color blindness. (a) The automatic generated palette makes the two changed classes with large saliency while maintains good separability between other classes. (b) Simulated results which is seen by people with protanomaly. We can see our results maintain a good performance for both people with normal vision and color vision deficiency.

REFERENCES

[1] Gapminder. The gapminder visualization system. `https://www.gapminder.org/data/`.

[2] K. Lu, M. Feng, X. Chen, M. Sedlmair, O. Deussen, D. Lischinski, Z. Cheng, and Y. Wang. Palettailor: discriminable colorization for categorical data. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):475–484, 2021. doi: 10.1109/TVCG.2020.3030406

[3] Y. Wang, X. Chen, T. Ge, C. Bao, M. Sedlmair, C.-W. Fu, O. Deussen, and B. Chen. Optimizing color assignment for perception of class separability in multiclass scatterplots. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):820–829, 2019. doi: 10.1109/TVCG.2018.2864912