COLOR PALETTE PROCESSING FOR THE VISUALIZATION THE DOMINANT COLORS

Modrekelidze Dali
Georgian Technical University

Summary

The article discusses the color search algorithm color compatible with graphic images. The algorithm eliminates the output of each pixel: R, G, B (red, blue, green) and summarizes the final color to which the alpha indicator is added. Finally, we get 5 color combinations from a variety of color spectra, which are very dominant, and as a result they are visualized.

Keywords: Dominant colors. Palette processing. Pixel iteration.

1. Introduction

Manipulation on graphic imagery is an important topic in today's computer science. The article discusses the extraction of colors from the graphic image and finding the dominant colors/visual display. By means of this algorithm, you can quickly capture the individual color and insert it into the appropriate color. It can be used for any photograph or graphic representation. The processing is done on vibrant.js library based on which we can handle fast manipulations on any site.

2. Main part

The main task is to find out the dominant colors of picture. To resolve it, I use the solution of iterating picture pixel by pixel with javascript canvas and vibrant library.

The algorithm includes in itself deploying the image as the array of RGB (red, green, blue) indexes and sorting them by the most usable ones.

First, we need to pass a picture to canvas, clone and then construct it with the same dimensions.

```
// // -- listing_1 --- Dominant colors extractor ------
function init() {
  var image = document.getElementById('mainimg');
  effectButton = document.getElementById('EffectButton');
  paintButton = document.getElementById('PaintButton');
  canvas = document.getElementById('Canvas');
  context = canvas.getContext('2d');
  canvas.width = image.width;
  canvas.height = image.height;
```

```
        drawImage(image);
        addEffect();
}
```

For the sake of apparently we use Canvas through which we print the original image. Step by step we iterate through the object data and save the prominent color codes with their types and compare to swatches pallete hexademical characters included in the library of vibrantjs:

```
// -- listing_2 --- Dominant colors extractor ------
vibrant = new Vibrant(image)
    swatches = vibrant.swatches(
    for swatch of swatches
      if swatches.hasOwnProperty(swatch) and swatches[swatch]
        for el in document.querySelectorAll ".color#{swatch}"
          el.style.backgroundColor = swatches[swatch].getHex()
  parseFile = (theFile) -> progres
  reader.onload = parseFile(f)
  data = reader.readAsDataURL(f) [1]
```

By example, if input picture looks like (Figure 1).

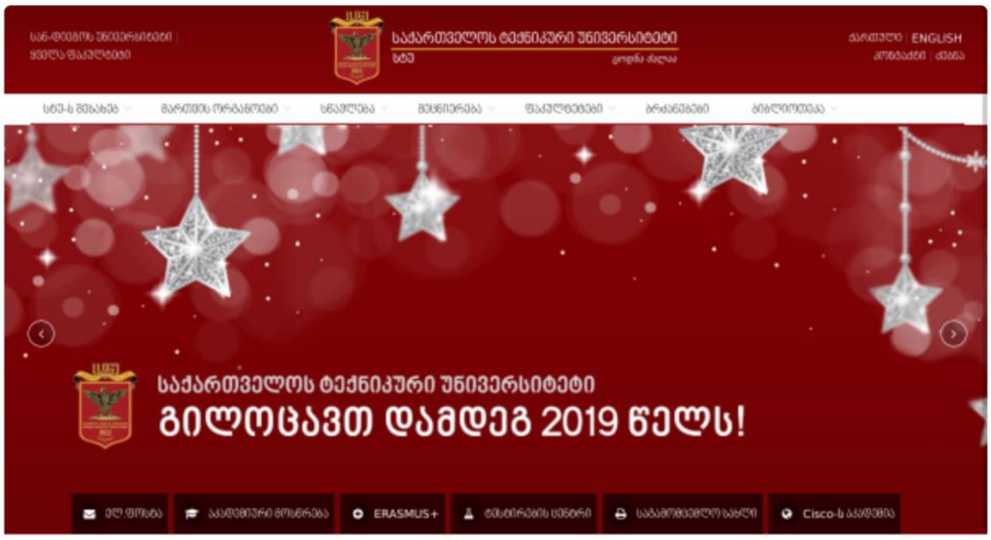
საქართველოს ტექნიკური უნივერსიტეტი
ENGLISH
საქართველოს ტექნიკური უნივერსიტეტი
გილოცავთ დამდეგ 2019 წელს!
ERASMUS+
Cisco-ს აკადემია

Fig.1. Screenshot of www.gtu.ge

And display the result:

```
/* LightMuted (population: 253) *\/
.lightmuted{background-color:#cbc0a2;color:#000000;}
```

```
/* DarkMuted (population: 11069) *\/
.darkmuted{background-color:#5b553f;color:#ffffff;}
/* Vibrant (population: 108) *\/
.vibrant{background-color:#dfd013;color:#000000;}
/* LightVibrant (population: 87) *\/
.lightvibrant{background-color:#f4ed7d;color:#000000;}
/* DarkVibrant (population: 2932) *\/
.darkvibrant{background-color:#917606;color:#ffffff;}
/* Muted (population: 4098) *\/
.muted{background-color:#a58850;color:#000000;} [2]
```

Our result visually looks like (Figure 2).

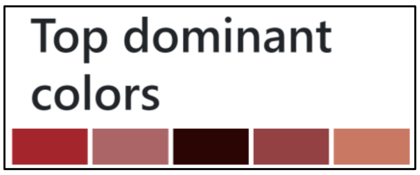

Fig.2. Application gui result

With the returned collection we can make lot of best combinations – make borders, shadows, reflections, bevel, emboss, etc. The usage is valuable. We have list of our top colors, which we can use in further development of our product, website or design. The code is dynamic and works with any type and size of pictures.

In the most cases the result gives us the right answer on the question – what are the most usable colors of your website? It may become one of main tools of web developers as it works fast and perfectly performs manipulation on images even without big resources of device (Figure 3).

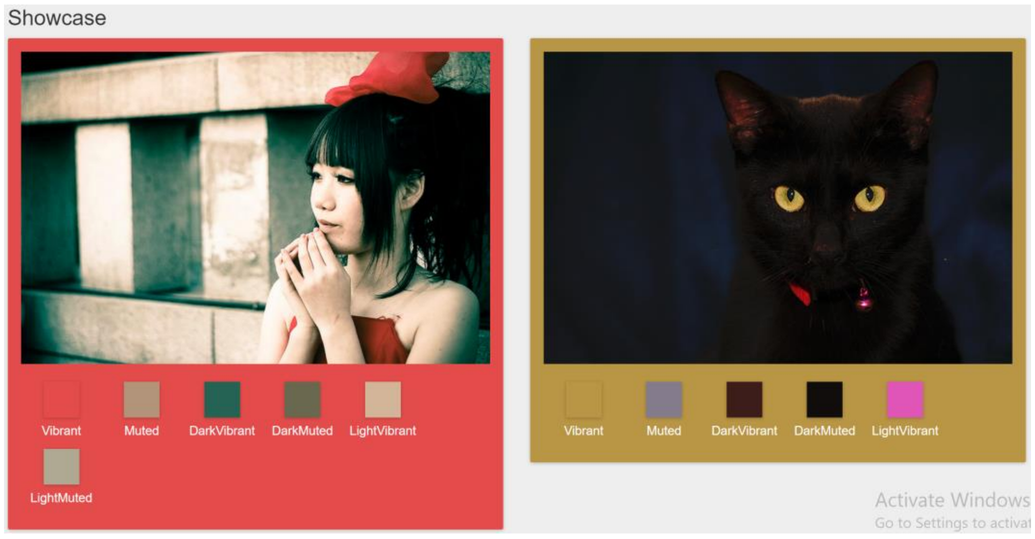

Fig.3

3. Conclusion

The algorithm is looking for the most usable color from the graphic image and keeps it in the appropriate array in the end 6 colors: Vibrant, Vibrant Dark, Vibrant Light, Muted, Muted Dark, Muted Light. [3] The color is displayed as a coded form, and its visualization is also perceived to be the same as for people who are aware of technologies, as well as ordinary users.

ლიტერატურა – References – Литература:

1. https://codepen.io/norcaljohnny/pen/MbOoMK
2. https://embed.plnkr.co/plunk/DGLrkj
3. https://www.npmjs.com/package/node-vibrant

ფერთა სპექტრის დამუშავება დომინანტური ფერების მისაღებად

დალი მოდრეკელიძე
საქართველოს ტექნიკური უნივერსიტეტი

რეზიუმე

განხილულია ფერების ავტომატური ამოღების ალგორითმი, რომელიც თავსებადია გრაფიკულ გამოსახულებებზე. ალგორითმი ახდენს თითოეული პიქსელის მახასიათებლის ამოღებას: R, G, B (red, blue, green) და მათ შეჯამებას საბოლოო ფერის მისაღებად, რომელსაც ბოლოს ემატება alpha მაჩვენებელი. ფერთა სპექტრის მრავალი მახასიათებლიდან კი საბოლოოდ ვიღებთ 5 ფერის კომბინაციას, რომელიც მეტად დომინანტურია, შედეგად კი ვახდენთ მათ ვიზუალიზაციას.

ОБРАБОТКА ЦВЕТОВОЙ ПАЛИТРЫ ДЛЯ ВИЗУАЛИЗАЦИИ ДОМИНИРУЮЩИХ ЦВЕТОВ

Модрекелидзе Д.
Грузинский Технический Университет

Резюме

Рассматривается цветовой алгоритм поиска цвета, совместимый с графическими изображениями. Алгоритм исключает вывод каждого пикселя: R, G, B (красный, синий, зеленый) и суммирует окончательный цвет, к которому добавляется альфа-индикатор. Наконец, мы получаем 5 цветовых комбинаций из множества цветовых спектров, которые являются очень доминирующими, и в результате они визуализируются.