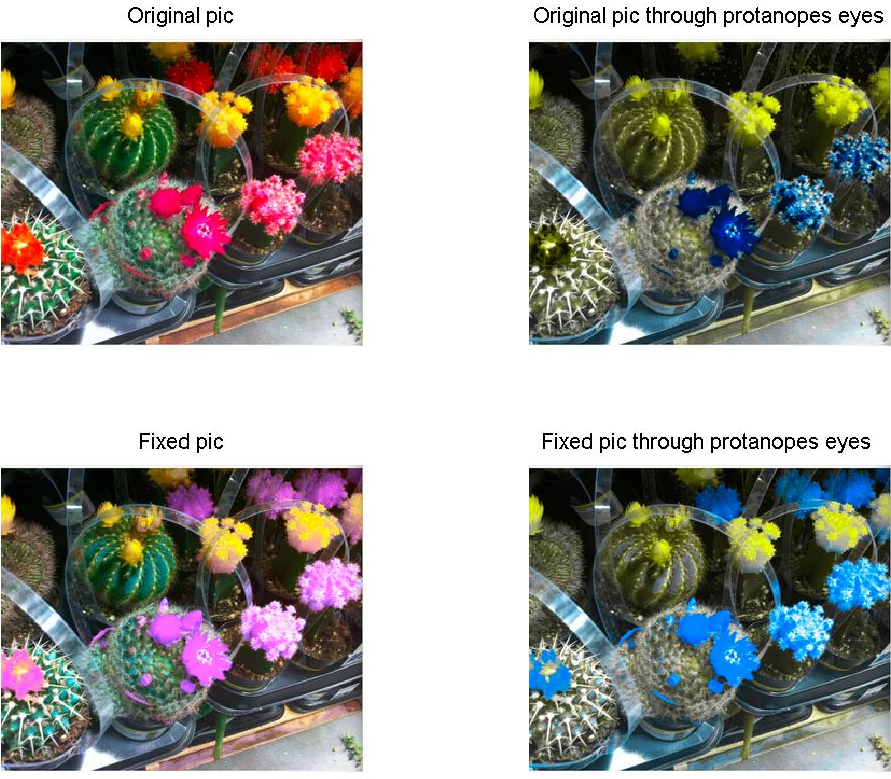
Project Documentation

Image Correction For Color Blind

****

Haim Shalelasvili

Zahi Kfir

Juli Dai

# Introduction

The human eye sees by light stimulating the retina (a neuro-membrane lining the inside back of the eye). The retina is made up of what are called Rods and Cones. The rods, located in the peripheral retina, give us our night vision, but cannot distinguish color. Cones, located in the center of the retina (called the macula), are not much good at night but do let us perceive color during daylight conditions.

The cones, each contain a light sensitive pigment which is sensitive over a range of wavelengths (each visible color is a different wavelength from approximately 400 to 700 nm). Genes contain the coding instructions for these pigments, and if the coding instructions are wrong, then the wrong pigments will be produced, and the cones will be sensitive to different wavelengths of light (resulting in a color deficiency). The colors that we see are completely dependent on the sensitivity ranges of those pigments.

Many people think anyone labeled as "colorblind" only sees black and white - like watching a black and white movie or television. This is a big misconception and not true. It is extremely rare to be totally color blind (monochromasy - complete absence of any color sensation). There are many different types and degrees of colorblindness - more correctly called color vision deficiencies**.**

People with normal cones and light sensitive pigment (trichromasy) are able to see all the different colors and subtle mixtures of them by using cones sensitive to one of three wavelength of light - red, green, and blue. A mild color deficiency is present when one or more of the three cones light sensitive pigments are not quite right and their peak sensitivity is shifted (anomalous trichromasy - includes protanomaly and deuteranomaly). A more severe color deficiency is present when one or more of the cones light sensitive pigments is really wrong (dichromasy - includes protanopia and deuteranopia).

5% to 8% (depending on the study you quote) of the men and 0.5% of the women of the world are born colorblind. That's as high as one out of twelve men and one out of two hundred women.

The protans (red weak) and deutans (green weak) make up 99% of this group.

Many research groups have conducted research on how to model the visually impaired vision, and published a number of papers presenting some algorithms to simulate what color-blind people see.

One such simulation program that has gained popularity is the "Vischeck" site ([www.vischeck.com](http://www.vischeck.com)).

This site also provides a model for daltonization, technique to modify a picture so that it is more visible to the visually impaired. In this project, we used the simulation data of [1,2] to generate a Matlab code that simulates how a color image is perceived by color blind people. Moreover, we generated a transformation code that daltonizes the digital image, base of [3,4,5,6].

We compare our results to those we get from the Vischeck site. The results we get turn out to be in reasonable agreement for both the color blindness simulation and daltonization. The algorithm was also verified by a fellow student who has color blindness. Hence we conclude our relatively simpler daltonization algorithm can be used to improve picture details visibility for the color blind.

# Goal

The goal of this project is to analyze how color blind people perceive colors in the world, and to make a simulation so that people with normal vision can understand what they see.

We also would like to make certain modifications in digital pictures to make life easier for the color blind people who use them.

This software will simulate what color blind people will see in an RGB image and correct (Daltonize) the picture so that they see better.

# Chosen Algorithms

### LMS Based Simulation Algorithm:

Color vision is achieved through the L, M and S cones in the human retina. These photosensitive receptors are sensitive to the long, middle and short wavelength ranges of the visible spectrum, respectively. Color blindness is the result of a deficiency of one (or more) of these photoreceptors. There are three typical kinds of color-blindness: protanopic, deuteranopic, and tritanopic, which correspond to the deficiency of the L cone, M cone, and S cone. These people have problem perceiving the full spectrum of colors normal people can distinguish.

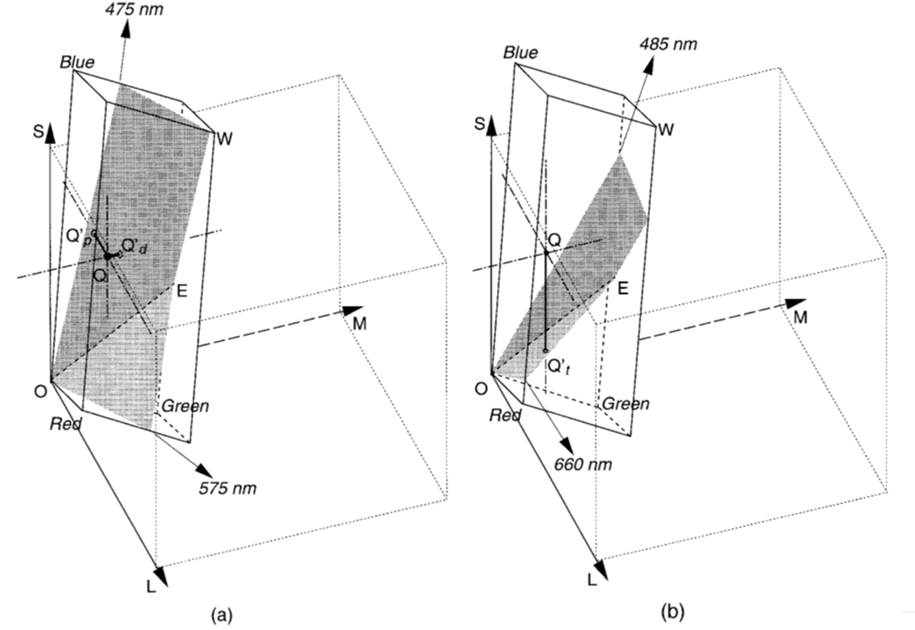
The basic idea of our algorithm is first to represent color stimuli as vectors in a three dimensional LMS space.

In the next step we make a conversion to delete the information associated with the loss of any of the cone types to get the modified LMS values LMS.

Finally, we make a reverse transformation on the LMS values to get the RGB values. RGB presumably represent how that specific color RGB is perceived by a color blind person. When this operation is done for all the pixels, the image is converted.

This involves a RGB to LMS transformation. This linear transformation can be achieved by a matrix multiplication, which is provided in [1].

The algorithm prepared in Matlab first takes in an image using the imread command and generates a matrix with the RGB values for each pixel. Then using the RGB-to-LMS matrix, this data is transformed into the LMS space. After obtaining the LMS value, the critical step comes, where we delete the information corresponding to one of the cone types. The transformation is done by considering the LMS color space.



B indicates blue, K black, W white, R red, G green, C cyan, M magenta, and Y for yellow.

For a normal person the color space spans over the KBMRGCWY parallelepiped.

For a protanope, all the colors which are on QpQ line will appear the same, which is the intersection color of QpQ and KBWY plane. Similarly, for a deuteranope, all the colors which are on QdQ line will appear the same, which is the intersection color of QdQ and KBWY plane. Clearly, information outside KBWY plane is lost for protanope and deuteranope.

The KBWY can be expressed as a plane equation: *αL+βM+γS = 0*

which passes through the points *(0,0,0), (LB, MB, SB), (LW, MW, SW)*. Solving *α,β,γ* using these three points, we can obtain

*α =MWSB - MBSW*

*β = SWLB - SBLW*

*γ = LWMB - LBMW*

Therefore, we can find *Lp* for protanope and *Md*for deuteranope:

*Lp = -(βM+γS)/α*

*Md =-(αL+γS)/β*

Which are the L cone response for a protanope and the M cone response for a deuteranope. The symbolic parameters are a function of the phosphor intensity functions of the specific CRT monitor used, and should be experimentally determined. In our simulations, we assumed a generic monitor.

The code finally converts the image back into the RGB domain and it could be saved as a jpeg file for review.

### Daltonize

Daltonization is a procedure for adapting colors in an image or a sequence of images for improving the color perception by a color-deficient viewer.

The basic idea behind daltonization is to calculate the error matrix, which is the image consisting of simulated image subtracted from the original image. This represents the information lost during the transformation. The error picture is what cannot be conveyed to a color blind person. We make a linear transformation on this picture so that it can be conveyed, and add this on the original picture to find the daltonized image.

For example, if the L cone is missing (protanope) the person will have difficulty in seeing the red part of the spectrum. Consequently, in the simulation, the error picture will consist of red shades mostly. Our transformation maps this information to the blue side of the spectrum. When this is added on the original picture we will get a daltonized version. The visibility of this image, therefore, is increased for a protanope.

#### Initial Transformation Matrices

Protanope :  Deuteranopic: Tritanopic: 

For each color blind type the problematic cone information is conveyed to the other two cones.

#### Why the current algorithm need improvement?

* Transformation matrix is manually chosen by the user until the resulting image is in adequate quality
* There is always the possibility of color matching between the original and the daltonized image
* Problem of unnatural appearance after the image processing

##### Solutions

* An automatic iteration technique for the selection of the transformation matrices
* A color checking module, which eliminates the possibility of color matching between the original and the daltonized image
* Add a set of rules for choosing the transformation matrix for the procedure to be more natural

#### The improved Daltonize Algorithm Steps

Our algorithm uses two groups of colors to find the optimal transformation matrix:

* Ccorrect – colors that the color blind person see correctly
* Cincorrect – colors that the color blind person see differently from a person with normal vision.

##### We needs to achieve 3 goals:

* For the Ccorrect colors – The daltoniztion procedure should do minor modifications to maintain naturalism.
* For the Cincorrect colors- The daltoniziation procedure must change them accordingly to the amount of information which lost.
* No color in Cincorrect should be daltonized to a color that the color blind people would perceive as similar to a color from Ccorrect

##### The algorithm:

1. Initialize a transformation matrix M.
2. Create quantized list of colors.
3. Classify each color from list of colors, as belonging to *C*correct or *C*incorrect .
4. Apply color daltonization to every color in *C*incorrect and name the resulting matrix *C*dalton
5. Run color blind simulation on every color in *C*dalton and name the resulting matrix *Csimulated*
6. If there is no color conflict between *C*correct and *Csimulated* go to step 7. Otherwise go back to step 2, after modifying M
7. Produce the result image by replacing, in the original image, every color in *C*incorrect with the corresponding color in *C*dalton

##### Modifying M:

On each iteration M is modified by conveying different amount of the missing cone information to the other two cones.

If we don't find a good transformation matrix we decrease the allowed distance between colors in the color checking module , which used in 6.

#### Known transformation matrix M:

There are two approaches after finding the transformation matrix:

* Modifying only Cincorrect using modification matrix M.

Advantages:

* Preserving naturalness.

Colors that color-deficient viewer have no problem seeing aren’t been modified.

Disadvantages:

* Unsmooth image colors.

There is predefined threshold to tell apart Cincorrect from Ccorrect, therefore, there might be two very close colors around that threshold, where one color mapped into Cincorrect and the other mapped into Ccorrect, the first color will be daltonize and the second will stay the same. Cause some noise pixels that are just above/below the threshold to be different from their surroundings.

* Modifying all of the original image using modification matrix M.

Advantages:

* smooth image colors.

Modify all pixels with a Linear daltonization matrix.

Disadvantages:

* Less Preserving naturalness

Daltonize Colors that color-deficient viewer have no problem seeing.

We use the second approach, Modifying all of the original image because the Naturalness is only Slightly damaged.

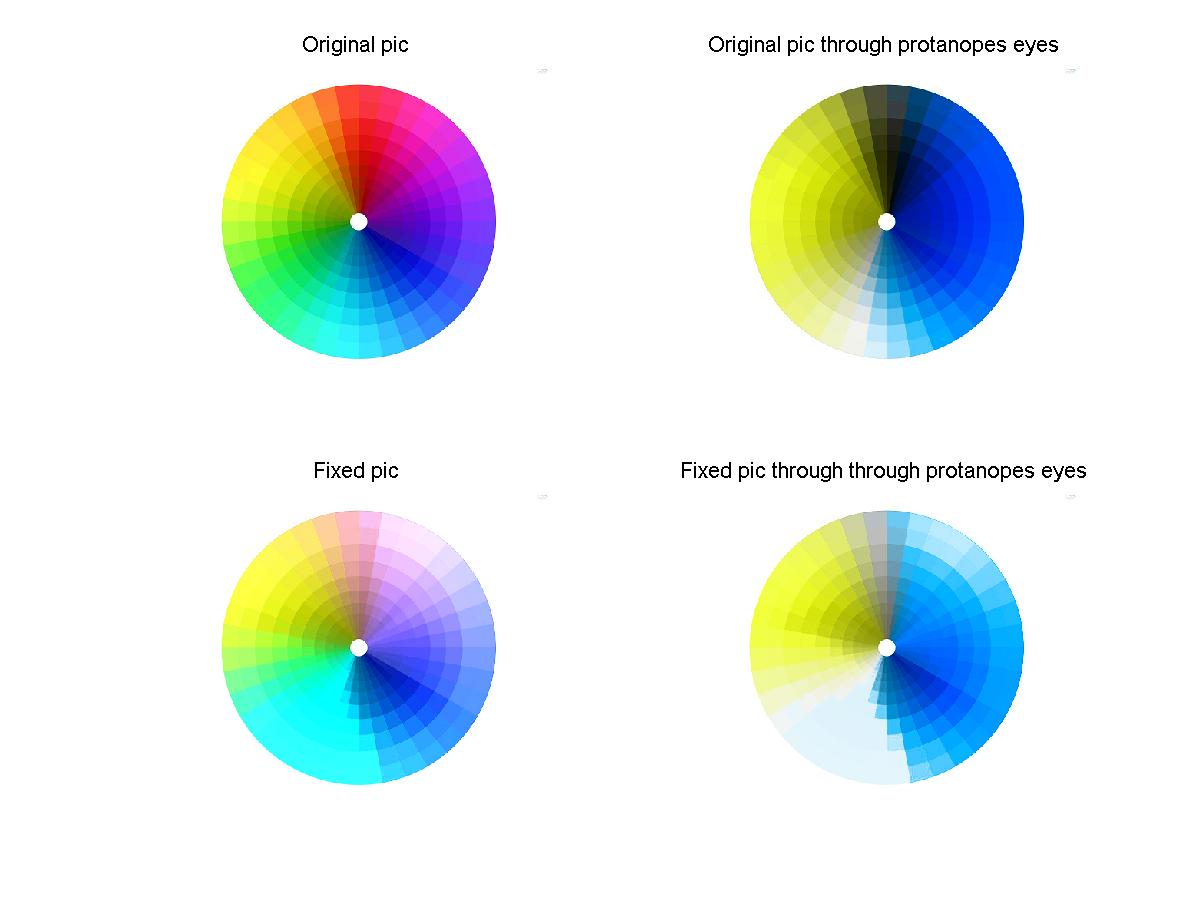
Ccorrect are colors that the color blind person see correctly (with very small error), since we daltonize correspondingly to the Error, the colors in Ccorrect are hardly changed.

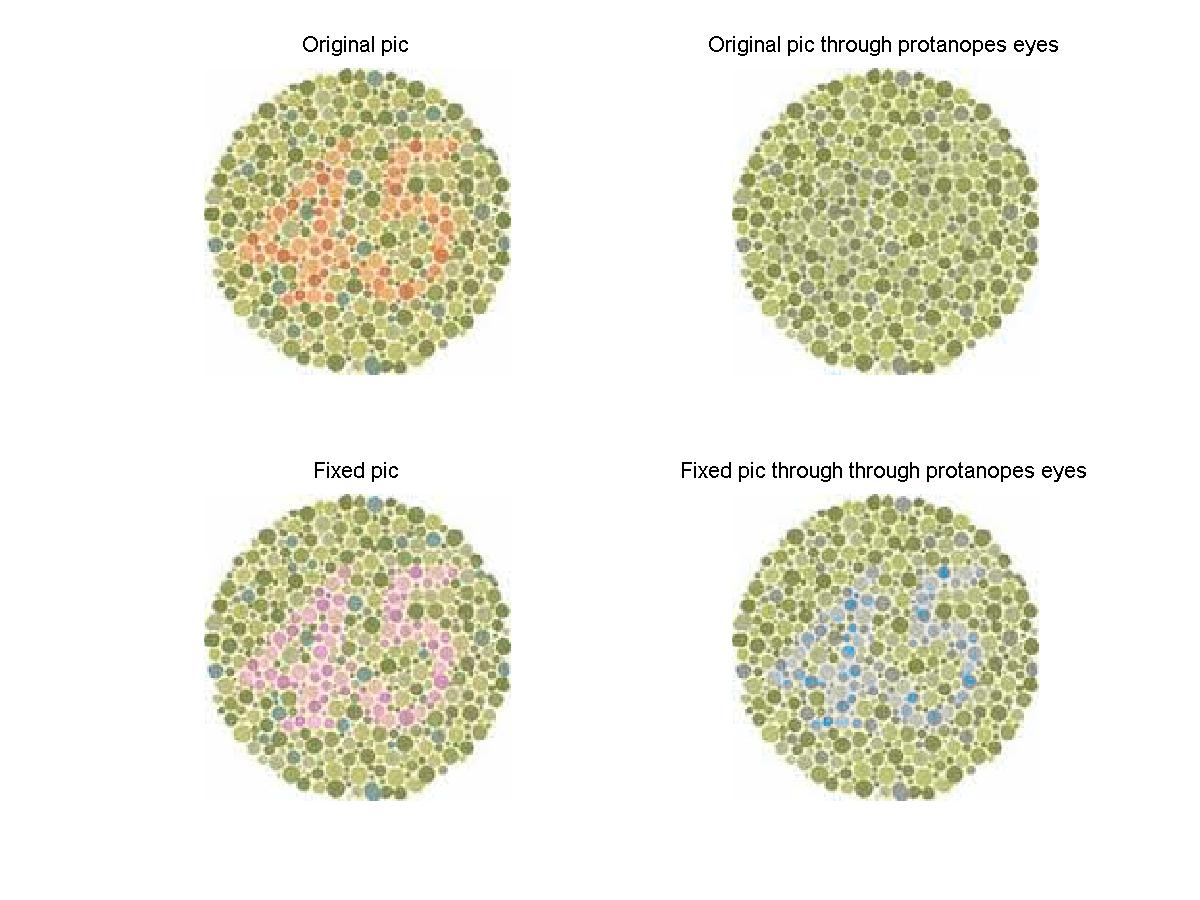
We gain much smoother image with Slightly Naturalness damage.

# Results

We used sample pictures to test the effectiveness of our algorithm.

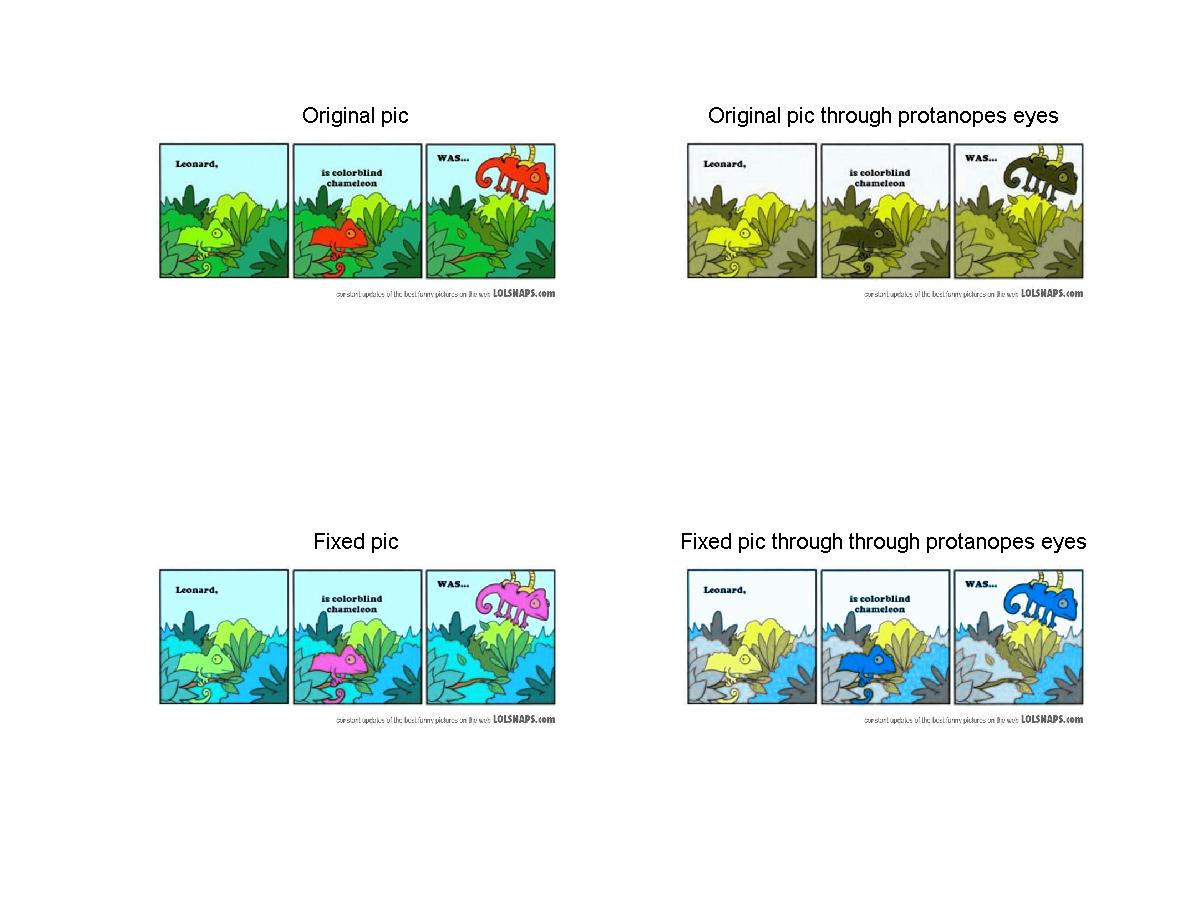
A few results:

1.

****

2.

3.



4.

# User-Guide

### Project structure:

This code is divided into couple of folders:

1. Color Blind simulation - implements the "LMS Based Simulation algorithm" which described in the project documentation file.
2. Daltonize - implements the chosen "Daltonize" algorithm which described in the project documentation file.
3. Misc - A service functions which supports the previous algorithms.
4. The main folder - Holds the graphical user interface for this project and a demo script.

For documentation and function guides for folders 1 , 2 and 3, please see the "[Chosen Algorithms](#_Chosen_Algorithms)" section and the readme file in each code folder.

The project was built and tested only on the "MATLAB R2013a" Ide

### Main folder guide ( The graphical user interface and the demo scripts):

#### Graphical user interface :

To run the graphical user interface please enter in the command window the command: "ColorCorrectionForColorBlind".

A graphical user interface should appear and the required folders will be added automatically to the matlab included folders path.

To get more information about the graphical user interface please check the "[Graphical user interface - User Guide"](#_Graphical_user_interface) section.

#### Demo script:

To run the main demo script, enter in the command window the command "ColorCorrectionDemo".

This script first will add the required folders to the matlab included folders path.

Then it will load 3 images from the Data folder: "Rainbow Fishes.bmp","Gaugin.bmp" and "Flowers.bmp".

For each image it will plot:

1. Color blind simulation image for Protanopes , Deuteranopes and Tritanopic.
2. Daltonize image for Protanopes , Deuteranopes and Tritanopic.
3. Daltonized simulate image for Protanopes , Deuteranopes and Tritanopic.

### Using the main functions manually:

First step is to manually add the algorithm folders to the included path of matlab - You can do it by selecting the folders in the "current folder" window in matlab, right click on them and select the option: "Add to Path" -> "Selected folders and subfolders".

Then you can call:

#### SimulateColorBlindImage

[ SimulatedImage ] = SimulateColorBlindImage( ColorBlindType , RGBImage )

A generic function which gets a color blind type and an image

It simulates what the color blind person sees

* Input:
  + ColorBlindType - 1 for protanopes, 2 for deuteranopes, 3 for tritanopic
  + RGBImage - A 3-dimensional matrix representing the RGB values of the original image
* Output:
  + SimulatedImage - A 3-dimensional matrix representing the RGB values of the simulated image

#### ColorBlindFix

[ Ifinal , ~ , ~ ] = ColorBlindFix( ColorBlindType, OriginalPic\_RGB )

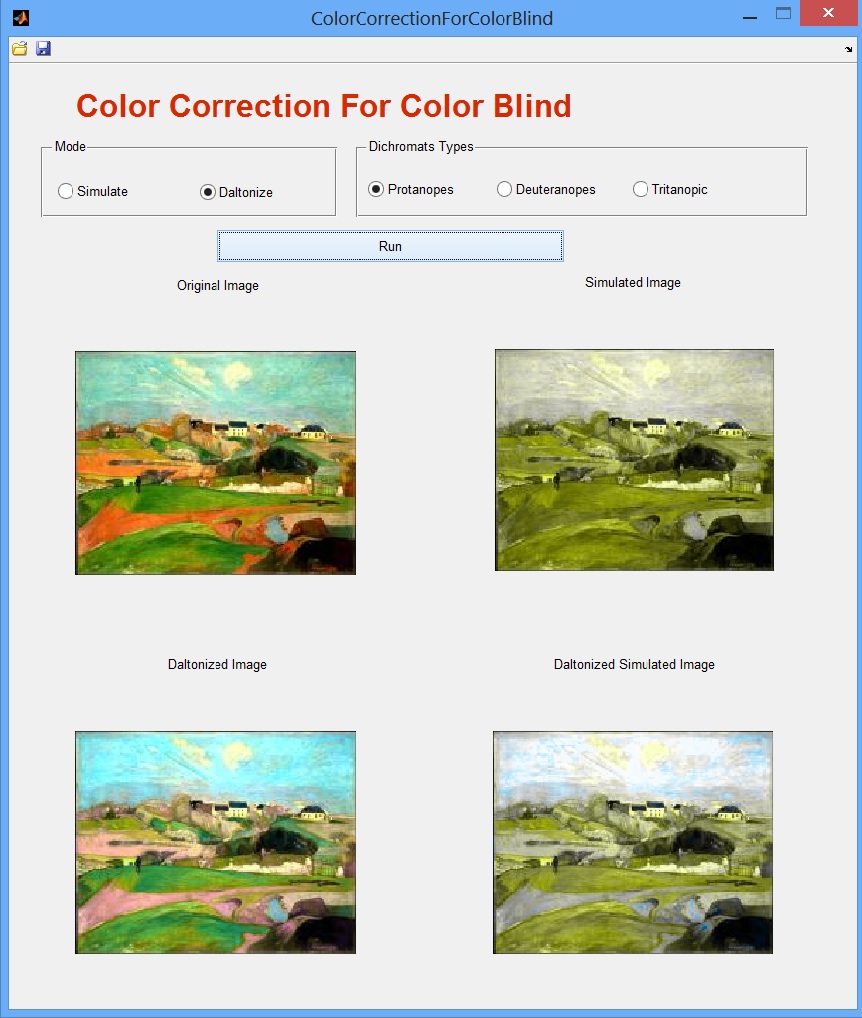
A generic function which gets a color blind type and an image.

Returns a daltonize image for the specific color blind person.

* Input:
  + ColorBlindType - 1 for protanopes, 2 for deuteranopes, 3 for tritanopic
  + OriginalPic\_RGB - A 3-dimensional matrix representing the RGB values of the original image
* Output:
  + Ifinal - A 3-dimensional matrix representing the RGB values of the daltonized image.
  + fileInfo - string which contains the info about the result (Only for debug
  + folderInfo - string which contains the info about the result (Only for debug)

! A normal user should only use the first output parameter, ignoring the other output parameters

# Graphical user interface guide:



**(3)**

**(7)**

**(6)**

**(4)**

**(5)**

**(1)**

**(2)**

**(8)**

**(9)**

**(10)**

**(11)**

**(1) -** Mode checkbox, choose between 2 modes:

Simulate mode - simulate image through color blinds eyes.

Daltonize mode - fix image for color blind.

**(2) -** Run button,

In Simulate mode - start simulation.

In Daltonize mode - start daltonization.

**(3) -** Dichromats type checkbox, choose between 3 types of color blinds.

**(4) -** Original image.

**(5) -** protanopes simulate Image / original simulate Image,

In Simulate mode - simulate image through protanopes eyes.

In Daltonize mode - simulate image through dichromats type

Checkbox (3) eyes.

**(6) -** deuteranopes simulate Image / daltonized image,

In Simulate mode - simulate image through deuteranopes eyes.

In Daltonize mode - daltonized image.

**(7) -** tritanopic simulate Image / daltonized simulate image,

In Simulate mode - simulate image through tritanopic eyes.

In Daltonize mode - simulate daltonized image through dichromats type checkbox (3) eyes.

**(8) -** Open image button,choose local image to work with.

**(9) -** Save results button, saves 4 images:

In Simulate mode - screenshot,protanopes simulation (5),

deuteranopes simulation (6),tritanopic simulation (7).

In Daltonize mode - screenshot, dichromats type checkbox simulation (5),daltonized (6),dichromats type checkbox daltonize simulation (7).

**(10) -** Minimize.

**(11) -** Close.

# Appendix

### CIE xy color space based simulation implementation:

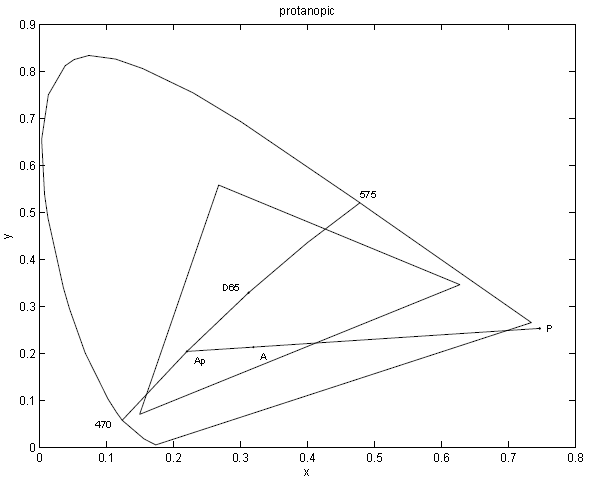
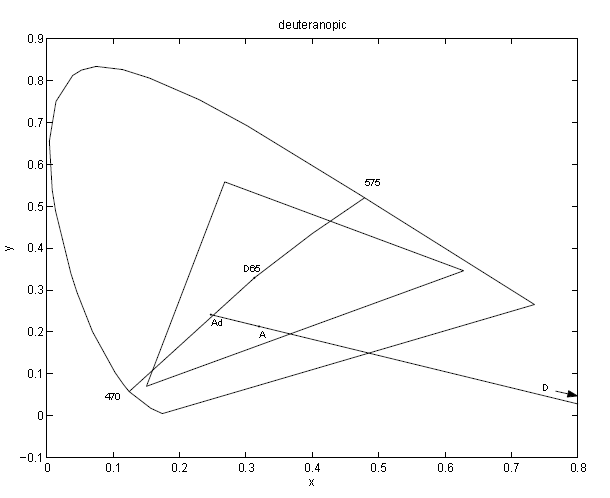
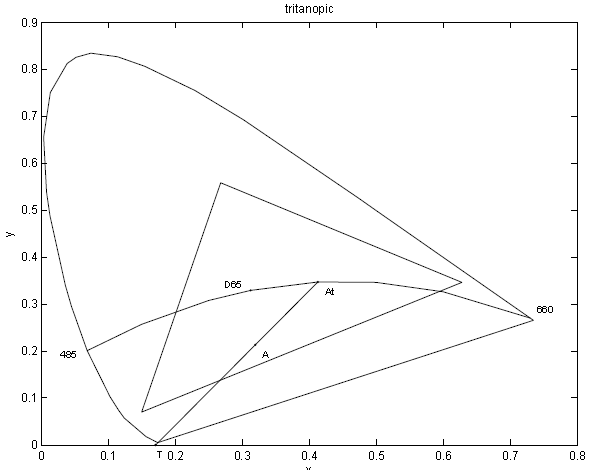
The CIE XYZ color space is based on direct measurements of the human eye, and serves as the basis from which many other color spaces are defined.

Simulation of dichromatic vision can be done directly on CIE x, y chromaticity values.

CIE chromaticity space of normal color vision collapses to a curve connecting the anchor points.

The anchor points for this curve are 470 nm and 575 nm for protanopes and deuteranopes, and 485 nm and 660 nm for tritanopes.

We also assume that D6500 white stays as the same hue in normal and dichromatic vision.

So the color space will collapse into a curve passing through the anchor points and D6500. We use the Lagrange interpolation to get the other points on this hue curve.

Basing on these observations, (Meyer and Greenberg, 1988) assume that color space of normal vision collapses to a line called ”major axis” on the uniform chromaticity diagram for each of the three types of dichromat.

All the loci (straight lines) representing stimuli of the same dichromatic chromaticity will converge to the same point in the chromaticity diagram.

That point is called confusion point and the loci passing specified stimuli and confusion point is called confusion line.

Our algorithm replaces color seen by dichromats by calculating the intersection between the confusion line and the hue curve.

#### Our algorithm :

For any chromaticity point A(x,y) in chromaticity space, the chromaticity point Ap (x’,y’) actually seen by dichromats is found by intersecting the confusion line passing A with the hue curve we get by the interpolation.

#### Main Disadvantage:

Very slow, it takes relativity long time to find the intersection points instead of just running linear transformation with the "LMS base color blind simulation" algorithm.

#### How to use this simulation algorithm:

Source Code for the implementation of this algorithm can be found in:

Code/Appendix/Color Blind Simulation (XYZ Based) folder.

To use it, simply replace the LMS based simulation code by the XYZ based simulation code.

The interface stays exactly the same as the other LMS simulation algorithm code and a readme file which explain each function available in the folder.

It’s highly recommended, not to use this algorithm when you running the daltonize algorithm. Color blind simulation is used in every step of the daltonization algorithm, and using the xyz simulation can take a long time.

Please stick to the LMS color space simulation algorithm that explained above when you are running daltonize.

# Conclusion

In this project, we developed an algorithm to simulate how color blind people see the world. With the algorithm, an RGB image is converted into the LMS space, where the information corresponding to the missing cone type is deleted. Converting this image back to the RGB plane, we obtain the modified picture.

We developed another code to daltonize images so that more information can be conveyed to a color-blind person. Also make the picture look more natural for him. In this, the error picture, which is obtained by subtracting the modified picture from the original, is further processed and the colors are mapped to some other part of the spectrum so that they can be perceived by color blind people.

The error -transform algorithm we developed for daltonization can be further modified such that the color mapping is determined according to the color content of the original picture. This can be done by simply trying different values for the parameters we use in the transformation matrix, and find out how the values affect the spectrum

# References:

1. H. Brettel, F. Vi´enot, and J. D. Mollon*. "Computerized simulation of color appearance for dichromats"* J. Opt. Soc. Am., 14(10):2647–2655, 1997.
2. Yinghua Hu. *"Visual Simulating Dichromatic Vision In CIE Space".*
3. Analysis of Color Blindness
4. Christos-Nikolaos Anagnostopoulos, George Tsekouras, Ioannis Anagnostopoulos Christos Kalloniatis. *"Intelligent modification for the daltonization process of digitized paintings"* Cultural Technology & Communication Dpt., University of the Aegean, Mytilene, Lesvos, Greece, 2007.
5. Paul Doliotis, George Tsekouras, Christos-Nikolaos Anagnostopoulos, and Vassilis Athitsos. *"Intelligent Modification of Colors in Digitized Paintings for Enhancing the Visual Perception of Color-blind Viewers"*
6. William Woods, "*Modifying Images for Color Blind Viewers"* ,Electrical Engineering Department,Stanford University,USA