

ON IMAGE RECOLORING – PART 1: CORRECT GRAYSCALE CONVERSION

Diana-Maria Popa^{1*}

ABSTRACT

This paper aims to analyze and develop the algorithm of the grayscale conversion by performing a reduction of the three dimensional input color set into a single dimension with a minimal loss of information. In the meantime, the results obtained are desired to be as perceptually plausible as possible. The related software application uses as input a colored image, three varying parameters and outputs a greyed-out image.

KEYWORDS: color conversion, greyscale to color, color to greyscale

1. INTRODUCTION

The focus of this paper is to develop an algorithm for obtaining greyscale images [1]. For bringing into focus the usefulness of this algorithm, we can mention the situation where a color document undergoes printing on a grayscale device and how indecipherable it can become after doing so. This is the main problem with greyscale transformation: reproducing as accurately as possible the intent, the contrasts and the salience of the original colored image. As most processing operations on images are done in single channel mode, ex: [27] [29], a channel that contains logical information from all colors is useful.

This paper presents part of the work done in the license thesis presented at the faculty of Automatics and Computers from the “Politehnica” University of Bucharest by the author of the paper [28].

2. RELATED WORK

Greyscale conversion

Algorithms that decrease the dimensional color components (ex: 24 bit color images to 8 bit greyscale) are referred to in the specialty books as dimensional reduction algorithms [12]. The main challenge is to optimally preserve and make use of the chrominance component of the image so that the initial appearance of the image is transferred to the resultant picture as accurately as possible. The focus lies on making use of the color differences of the initial image in order to retain feature discriminability in the grayscale appearance of the output.

The conversion methods that regard achromatizing an image can obtain their desired results by using either a local mapping or global mapping. The main difference between the two mentioned types of mapping is that the global mapping preserves a more homogenous feel on the output image, as it ensures that same pixels values will get

^{1*} corresponding author, Engineer, diana.popa@cti.pub.ro, “Politehnica” University of Bucharest, 060042 Bucharest, Romania

mapped to the same value every time and everywhere. Meanwhile, the locally mapping algorithms can be more accurate in preserving local characteristics of the image but can produce inhomogeneous results as the mapping function changes. Moreover, the latter ones are slower.

It is a wide known fact that color is represented by a three-dimensional concept, the RGB color space, which basically refers to the three main attributes of color: lightness, hue (tone), Chroma (pigment, coloration). In the converted image the one attribute that is preserved is the luminance, the other two (i.e. Chroma and hue) are discarded. The lightness (i.e. luminance) is the achromatic reaction to a color incentive. This is exactly what the challenge is, to find an algorithm that better combines the initial 3D information into a 1D information to optimally reflect initial appearance and dynamism.

A much discussed local mapping approach was introduced by [12]. The authors described a method of infiltrating the chrominance information into the luminance component of the output image by applying a high-pass filter onto the chrominance information based on the spatial information of the pixels. They aim to prove that by making use of the location of the pixels, the problem of different pixels having similar luminance values would be greatly improved.

Another local mapping algorithm [13] developed a linearly complex technique of preserving the chromatic contrasts based on a research of the Coloroid color system observations, which is a tool to express the relational phenomena and psychometric characteristics that relate each color to another. In addition to this tool, the authors [13] also present their extension to the Color2Gray [1] method. They introduce a new function for computing the delta gray difference which greatly reduces computational cost and also achieves better results.

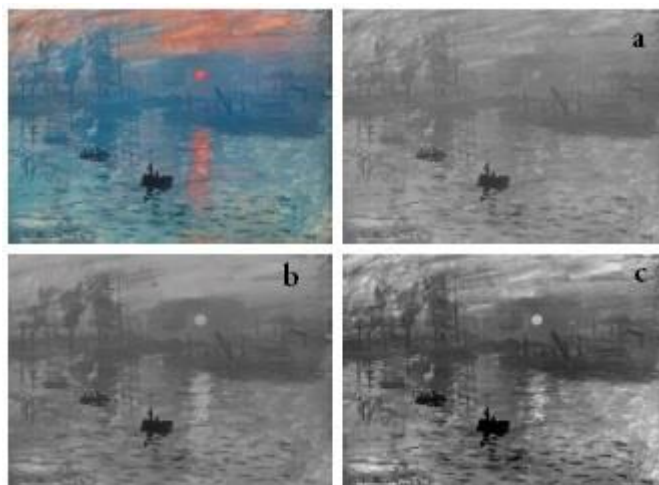


Figure 1. Comparison of simple CIE Y method (a), Gooch et al. method (b), Neumann (c) et al. method (image taken from [13])

One interesting greyscale transformation technique is that of Smith et al. [14]. Their work proposes the development of an algorithm that combines global as well as local concepts. The mapping of the pixels takes into account in a globally way the lightness and it is based on the Helmholtz-Kohlrausch (HK) Effect [15]. This is a technique which, in very simple terms, predicts the differences between colors with the same luminance values. The latter mentioned effect causes a more colorful stimulus to appear brighter (i.e. as if he had a greater luminance value) than another stimulus less colorful and with the same luminance value. The other half of the algorithm in discussion implements a contrast enhancement which behaves locally. The method proposed has linear complexity and Smith et al. [14] also bring into focus the applicability of their algorithm on videos. Figure 2 shows the consequences of the HK effect. Figure 2.a presents the original picture which is made up entirely of isoluminant colors (i.e. colors that have the same luminance value). Figure 2.b of the shows basic CIE XYZ transformation, the HK effect is not incorporated, the chrominance differences are omitted. The last picture reflects the mapping proposed by Smith et al. [14] which uniquely and correctly matches ordered grey scale values.



Figure 2. The HelmHoltz-Kohlrausch Effect (image taken from Smith [14])

The global mapping techniques can be further classified into non-linear and linear.

Several global mapping algorithms were proposed by Rasche and others researchers in the field. For example in [15], a linear global function is used. The preservation of contrast is thus done by defining a quadratic objective function, after that they apply constraints in order to make sure of the consistency of the luminance. Some unique feature of their method is that they are able to extend it so that it can recolor images for the color-deficient viewers.

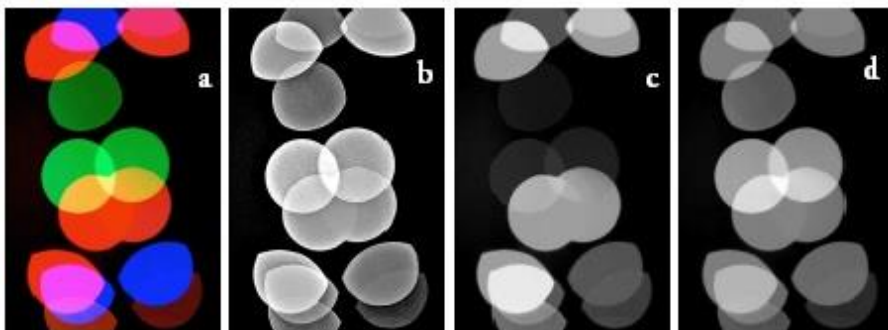


Figure 3. Comparison between Color-to-Gray conversions (image taken from [16])

Another approach was taken by [16]. The authors present a very interesting comparison between different types of mapping approaches (local, linearly global mapping, non-linear global mapping). As far as their experiments, it seems like their method is the one that better reflect the original lightness of color values. Figure 3a is the input. Figure 3b represents the result of the greyscale conversion of the local mapping algorithm of Smith et al. mentioned above [14]. Figure 3c is the result of global linear mapping [19] and figure 3d is their [16] non-linear global method.

The work developed is based on [1] and the results, as well as the theoretical portrayal of the algorithm, will be presented in a later chapter. Figure 4a represents the input image, 4b is an example of applying Photoshop grayscale method, 4c is the better result from a qualitative point of view obtained with Color2Gray algorithm.



Figure 4. Comparison between Color-to-Gray conversions in Gooch et al. [1]

Also there exist iterative methods that start with a best looking assumption (in this case the grayscale conversion using the weights the eye sees at a global level) and with each iteration the differences in contrast between the color image and grayscale image are minimized [24].

3. GREYSCALE CONVERSIONS – THEORETICAL PORTRAYAL

Challenges in greyscale conversions

A very obvious and already discussed issue is the problem with the isoluminant colors: how to make them visually perceptible and distinguishable after the greyscale processing, on what criteria an isoluminant component should be brighter or darker. The common approaches are to let the user manipulate this issue through some parameters. For example Figure 5 shows how the colorful blue circle from the input image could be very well mapped to a brighter luminance value (right) or to a darker one (left) than the isoluminant background color (grey).

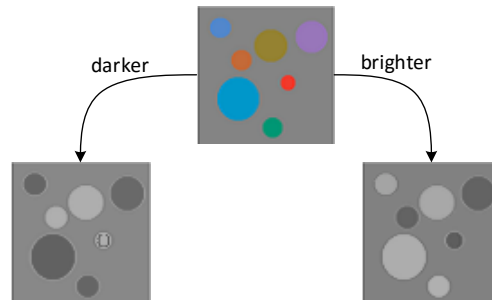


Figure 5. Challenge1: Mapping to darker or brighter values the isoluminant colors

Another second challenge when it comes to converting color images to greyscale ones, especially when a local mapping algorithm comes into discussion, is how to optimally and carefully select the size (radius) of the neighborhood. The importance of the locality size is described here: [26].

A less obvious and more sophisticated challenge is the mapping of the chrominance changes into luminance values. It is necessary to take into account that the chrominance values for most of the color spaces usually have a larger spectrum of values than the luminance. For example, the CIE $L\alpha\beta$ color space chrominance (i.e. α and β components) range is from -128 to 127. For the CIE LUV color space, the U range is [-134...220] and the V's range is [-140...220]. The luminance spectrum is only from 0 to 100. The problem that could appear if we make a sloppy decision and/or a silly mistake is to obtain a completely saturated image from a luminance point of view.

Latest paragraph brought into discussion the existence of multiple color spaces and how their main characteristics can vary in a definite way. The two color spaces already mentioned, CIE LUV and CIE $L\alpha\beta$, are referred to in the specialty books as uniform color models. One of the most significant characteristics of the CIE $L\alpha\beta$ color space in discussion is the fact that its axes tend to have little correlation. The uniformity of a color space is the property by which the difference between any two colors from a perceptually human visual system point of view varies in a proportional way with the Euclidean difference measured inside the specific color space [17]. The choice of color space is important and thus it can also represent a challenge. In [18] Reinhard et al. performed a comparison between using different color spaces in color transfer techniques (see Figure 6) and came to the conclusion that color spaces $L\alpha\beta$ and CIE LAB (with illuminants E) perform the best. The MD, IN, MN and ND from the legend refer to different environmental settings (i.e. manmade day, indoors, manmade night and natural day respectively).

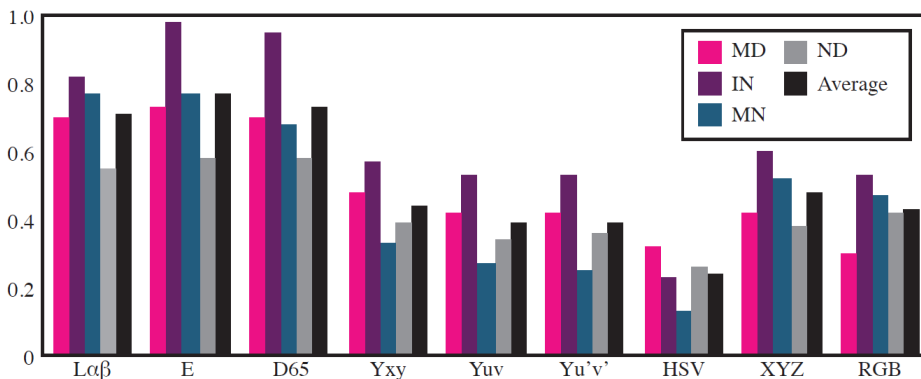


Figure 6. Percentage of successful color transfer for different context images (image taken from [18])

Because CIE Lab (E illuminant) was shown to perform well in most of the cases as per analysis [18], this uncorrelated color space is also the choice of the algorithm implemented in the available framework.

4. ALGORITHM DESCRIPTION AND IMPLEMENTATION

The algorithm on which the presented framework was built upon is based on the theoretical portrayal given by Gooch et al. in [1]. The first step as per the later chapter of this paper is to convert the data of the input color image from RGB space to CIE $L^*a^*b^*$ space. The function which does this performs a RGB to CIE XYZ space conversion and then from there the CIE XYZ values are transformed to CIE Lab illuminant using illuminant D65.

After acquiring the $L^*a^*b^*$ values, the next phase of the algorithm proposed is the computation of the deltas (i.e. differences) between pixels. This differences will be calculated based on the luminance (L^* component) and chrominance values (a^* and b^* components) from the uncorrelated $L^*a^*b^*$ color space. In the third phase, the array of differences of all pixels will be used to obtain the final greyscale image. This is done by using an iterative process in which the final image is computed out of the deltas from previous step. The goal of this iteration process is to obtain some data output which is as similar to the deltas computed as possible.

In this section the challenges of greyscale conversion were exposed. In the presented algorithm, the course of decisions regarding the mentioned challenges are solved by requiring the user to give input. Upon running the software program the user will be asked to input three parameters which will be further discussed.

Parameters

The three parameters that the framework requires when running Color2Gray implementation are:

- Θ – This is the parameter that solves the decision regarding Challenge1 (see 3.1). It basically controls the direction in which the isoluminant colors are being made distinguishable, either by increasing in brightness or in darkness.
- α – This parameter relates to Challenge2. It must be found a way to scale the chrominance values of a^* which are in the range $[-200, 200]$ and those of b^* - $[-500, 500]$ to the range $[0, 100]$ which corresponds to the L^* channel. The α parameter sets a limit to how much of an impact the chrominance difference between two arbitrary pixels can have on the final computed luminance. It is obvious that this limit is equal to $[-\alpha, \alpha]$.
- μ - the framework gives the possibility to the user to control whether or not the algorithm should have a local behavior or a global one. Thus, by providing a third parameter with a value that is greater than 0, the algorithm will try to preserve local changes and will take decisions based on a neighborhood size equal to μ . If the value of the μ is 0 the algorithm will perform globally (Challenge3).

Notations

Throughout the description of the algorithm there will be used some notations that are worth describing and that will be used in the pseudo code of the algorithm. The luminance of one specific pixel i will be denoted as L_i . The difference between the luminance, a^* and b^* values of two arbitrary pixels i and j are δL_{ij} , δA_{ij} and δB_{ij} respectively. It suffices to consider that the luminance difference between two pair of pixels is exactly δL_{ij} but when it comes to chromatic difference between pixels i and j we introduce a new notation $\vec{\delta C}_{ij}$ which, in contrast to the scalar δL_{ij} , is a vector. Another notation worth mentioning is $\vec{V}_\theta$ which is a vector calculated based on the Θ parameter in order to decide the way the luminance should oscillate (see Challenge1 from 3.1). All the above notations lead to one goal, obtaining the output delta of some pair of pixels: δ_{ij} .

Pseudo code

```

ConvertRGBtoXYZ
ConvertXYZtoLab
if NeighborSize > 0 then
    for i := 0...AllPixels do
        for j in NeighbourPixels do
            Delta = ComputeDeltas(i,j)
            Deltas[i] += Delta
            Deltas[j] -= Delta
        done
    done
else
    for i := 0...AllPixels
        for j := i...AllPixels
            Delta = ComputeDeltas(i,j)
            Deltas[i] += Delta
            Deltas[j] -= Delta
        done //for j
    done //for i
if NeighborSize > 0 then
    for Iteration = MAX_ITERATIONS;
        Iteration > 0;
        Iteration := Iteration - 1
        for i : 0...AllPixels do
            Sum := 0
            Neighbors := 0
            for j in NeighbourPixels do
                Sum += Greyscale[j]
                Neighbors++
            done
            Greyscale[i] = (Sum + Deltas[i]) / Neighbors
        done

```

```

        done
    else
        for i := 1...AllPixels do
    Greyscale[i] = (Deltas[i] – Deltas [i-1]) /
    AllPixels + Greyscale [i-1]
        done
    ConvertLabToXYZ
    ConvertXYZtoRGB

```

Computing the deltas

The most intensively computational part of the algorithm, approximately 95% of the total processing time, is the one that deals with the calculation of the deltas, as per the pseudo-code above, the ‘computeDelta’ function. This function receives as input the position of two pixels, let’s denote them i and j , and it outputs δ_{ij} . The first decision that the logic of this function has to take is to compare the difference in luminance δL_{ij} and the difference in chrominance $\delta \vec{C}_{ij}$. In order to compute the difference in chrominance, the [1] defines a function named ‘crunch’ which has the main goal to limit the much bigger spectrum of values of the chrominance to a smaller one which is $[-\alpha, \alpha]$ in order to fit the luminance’s specific range of $[0,100]$. This ‘crunch’ function receives as input the Euclidian distance between the two chrominance points, the a^* coordinate and the b^* coordinate, and outputs its adjusted value. The value gets adjusted as per [1] by applying the equation on the Euclidian distance and by making use of the inputted α parameter.

$$crunch(x) = \alpha * \tanh\left(\frac{x}{\alpha}\right) \quad (1)$$

The equation (1) ensures that all possible values are least than α and bigger than $-\alpha$.

After having obtained also the chrominance difference the logic looks at testing whether or not the luminance difference is bigger than the chrominance. If this is true than there is no point in going to the next step and so the function simply returns δL_{ij} . But if the chrominance difference happens to be bigger than the luminance, it means that there is the possibility at looking at two isoluminant color values, and so the Challenge1 comes into focus. As per an earlier chapter, the Challenge1 is solved by making use of the θ parameter. Equation (2) basically is a way of separating the color space in negative and positive parts and based on that to make isoluminant colors brighter or darker. The output in this case is decided by $\vec{v}_\theta$ itself: if its sign is equal to that of the ‘crunched’ chrominance than the ‘crunched’ chrominance itself is returned, otherwise it is returned the negative value of the ‘crunched’ chrominance.

$$\vec{v}_\theta = \left(\cos \theta * \delta A_{ij} + \sin \theta * \delta B_{ij} \right) \quad (2)$$

Complexity and Performance

One first aspect that greatly influences the complexity and, implicitly, the performance of the algorithm is by the total number of pixels of the image to be processed. In the framework proposed, if the user does not specify a value for the input parameter μ (i.e. the number of neighbor pixel), then the algorithm will perform in a globally way, thus generating every pixel pair possible. Let's suppose that the input image is a $N \times N$ square image. Then the complexity for generating every pixel pair for the complete case (full neighborhood) is $O(N^2 * N^2)$ and for the partial case is $O(\mu^2 * N^2)$. Apart from this complexity, there is also the complexity for transforming the RGB color space to Lab color space and at the end back to RGB space, which brings an additional complexity of $O(N^2)$, but which is insignificant compared to that of $O(N^4)$ which is necessary to compute the deltas needed. As a result the computation for a 128x128 image takes 30 seconds on an i5 processor. Limiting μ to 16 the time drops to 7 seconds.

Results and Discussion

Figure 7 shows the impact that the varying of the parameter θ has on the outputted greyscale picture. As a reminder, this is the parameter which controls whether the difference for isoluminant colors should increase or decrease in brightness. It can be easily noticed how the green colors appear darker on the right side of the trigonometric circle and how they are lighter on the left side (especially in the second quadrant). The picture in discussion shows a very simple way of understanding how the θ influences the lightness variance. It is interesting to notice that the a^* and b^* components of the $L^*a^*b^*$ color space have been drawn with very suggestive colors. Large a^* values indicate red, small values indicate green. In a similar way, small b^* values point towards yellow and large ones are for blue.

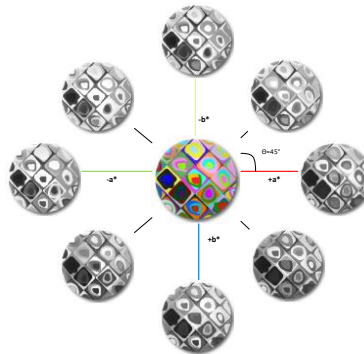


Figure 7. Influence of θ parameter on greyscale image

The next interesting phase of testing the proposed algorithm was that of varying the α parameter. This one is the one that is in charge with how much of an impact the difference between the chrominance between two pixels should have on the outputted greyscale image. The ‘crunch’ function limits the possible values of the deltas to $[-\alpha, \alpha]$. Figure 8

shows the results of running the tests with different alpha parameters. It is worth noticing that while the alpha increases, the chromatic variance between the images also gets more obvious. The images were obtained with full case neighborhood and $\theta=45^\circ$. The picture elements that best reflect this phenomenon is the sun and its reflection in the water. In the top most pictures ($\alpha=25$ and $\alpha=30$) they are more strident.

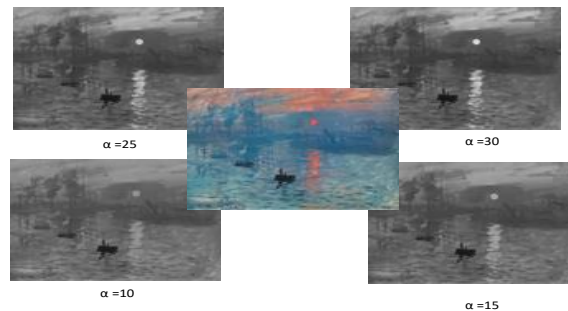


Figure 8. Influence of α parameter on greyscale image (image taken from [1])

The last parameter left to be exercised and whose influence is to be noticed is the radius of pixels (neighborhood size). It can be seen from test scenario Figure 9, that when taking into account a smaller number of surrounding pixels the color progression is more obvious because of the image gradients that do not correspond to the initial colorful picture (eq. the grass in bottom pictures from Figure 9). Also it is possible for two pixels that are not in each other's radius and which have same luminance but different chrominance to be mapped to the same final value, which is not necessary correct.

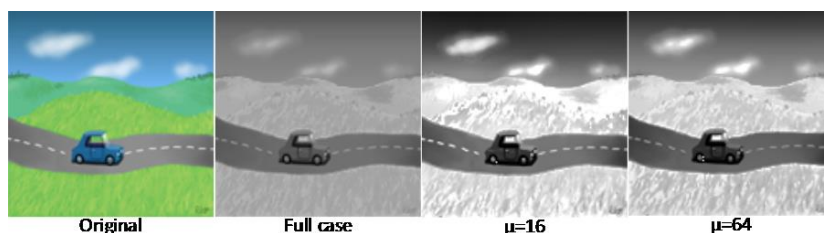


Figure 9. Influence of μ parameter on greyscale image (source image taken from [1])

5. CONCLUSIONS AND FUTURE WORK

The framework developed through this paper has several advantages. Firstly, it is very easy to use. The User Interface implemented in MATLAB through which the user interacts with the executable is very intuitive and friendly. This is what makes the effort of the user to be as low as possible.

Secondly, the framework offers some level of manipulation of the outputted images. The greyscale conversion gives the possibility to the user to vary its input parameters (theta, alpha angles and also the neighborhood size).

One limitation of the current developed framework is the fact that the output of the format image cannot be controlled. The output image will always be of PNG format, regardless of what format the input picture may be.

Another disadvantage may refer to performance issues. The algorithm for greyscale conversion has a complexity of $O(n^4)$ which means this is not one of the fastest algorithms that there can be for coloring in gray a colorful image.

Future work will be focused on finding a way to reduce the costly computation of the target differences; perhaps a method of storing the difference in luminance values for every pair of pixels in a table so that to avoid the overhead of computing them every time they are needed. Another solution to the performance issue is to try to parallelize the code as the logic of the algorithm seems very permissive as the values of the currently computed output pixels depend only on the initial values in the input image.

Maybe it could also be interesting investigating the behavior of the greyscale algorithm when applied in RGB space and what kind of adjustments are necessary in order to eliminate the problem of strong correlation of axes. Another interesting continuation of the greyscale conversion would be to recolor the obtained grayed out image back to a colored one by using the luminance values obtained through the algorithm and keeping the chrominance features from the initial image.

REFERENCES

- [1] Gooch, J. Tumblin, B. Gooch and S. Olsen. Color2Gray: Saliency Preserving Color Removal. In *ACM Transactions on Graphics*, Volume 24, Issue 3, 2005, pp. 634-369.
- [2] F. Pitie, A. Kokaram and R. Dahyot. Towards Automated Colour Grading. In *IEEE European Conference on Visual Media Production*, November 2005.
- [3] F. Pitie, A. Kokaram and R. Dahyot. Automated Colour grading using colour distribution transfer. In *Journal of Computer Vision and Image Understanding*, February 2007.
- [4] E. Reinhard, M. Ashikhmin, B. Gooch and P. Shirley. Color Transfer between Images. In *IEEE Computer Graphics Applications*, Volume 21, Issue 5, 2001, pp. 34-41.
- [5] D. Ruderman, T. Cronin, C. Chiao. Statistics of cone responses to natural images: implications for visual coding. In *Journal of the Optical Society of America*, Volume 15, Issue 8, 1998, pp. 2036-2045.
- [6] C.M. Wang, Y.H. Huang. A novel color transfer algorithm for image sequences. In *Journal of Information Science and Engineering*, Volume 20, Issue 6, 2004, pp. 1039-1056.
- [7] M. Grundland, N.A. Dodgson. Color histogram specification by histogram warping. In *Proceedings of the SPIE*, Volume 5667, 2004, pp. 610-624.
- [8] E. Pichon, M. Niethammer and G. Sapiro. Color histogram equalization through mesh deformation. In *IEEE International Conference on Image Processing*, Volume 2, 2003, pp. 117-120.

- [9] L. Neumann, A. Neumann. Color style transfer techniques using hue, lightness and saturation histogram matching. In *Proceedings of Computational Aesthetics in Graphics, Visualization and Imaging*, 2005, pp. 111-122.
- [10] Y. Chang, S. Saito, K. Uchikawa, M. Nakajima. Example-based color stylization of images. *ACM Transactions on Applied Perception*, Volume 2, Issue 3, 2005, pp. 322-345.
- [11] T. Welsh, M. Ashikhmin, K. Mueller, M. Nakajima. Transferring color to greyscale images. *Proceedings of ACM SIGGRAPH*, San Antonio, 2002, pp. 227-280.
- [12] R. Bala, R. Eschbach. Spatial color-to-grayscale transform preserving chrominance edge information. In *Color Imaging Conference*, 2004, pp. 82-86.
- [13] L. Neumann, M. Čadík and A. Nemsics. An efficient perception-base adaptive color to gray transformation. In *Proc. Computational Aesthetics*, 2007, pp. 73-80.
- [14] K. Smith, P. E. Landes, J. Thollot and K. Myszkowski. Apparent Greyscale: A simple and fast conversion to perceptually accurate images and video. In *Computer Graphics Forum (Proc. Eurographics 2008)*, Volume 27, Issue 2, 2008, pp. 193-200.
- [15] Y. Nayatani. Simple estimation methods for the Helmholtz-Kohlrausch effect. In *Color Research and Application*, Volume 22, Issue 6, 1997, pp. 385-401.
- [16] K. Yongjin, J. Cheolhun, J. Demouth and S. Lee. Robust Color-to-Gray via Nonlinear Global Mapping. In *ACM Trans. Graph*, Volume 28, Issue 5, 2009, pp. 1-4.
- [17] H. Lekowitz. Color theory and Modeling for Computer Graphics, Visualization, and Multimedia applications. In *Kluwer Academic Publishers*, 1997, pp.55-58.
- [18] E. Reinhard and T. Pouli. Colour spaces for colour transfer. In *Proceedings of the Third International Conference on Computational Color Imaging*, Volume 6626, 2011, pp. 1-15.
- [19] M. Grundland, N. Dodgson. Decolorize: fast, contrast enhancing, color to grayscale conversion. In *Pattern Recognition*, Volume 40, Issue 11, 2007, pp. 2891-2896.
- [20] <http://blogs.wsj.com/economics/2011/03/28/top-10-dying-industries/>. "Top 10 Dying Industries - Real Time Economics – WSJ", Accessed on: 27 Jun 2014.
- [21] <http://www.mathworks.com/help/images/radon-transform.html>, "Radon Transform – Matlab & Simulink", Accessed on: 30 Jun 2014.
- [22] Van Gogh, Vincent. Olive Trees. 1889. The Minneapolis Institute of Arts, Minneapolis. <http://www.vangoghgallery.com/catalog/Painting/360/Olive-Trees-with-Yellow-Sky-and-Sun.html>. Accessed on 1 Jul 2014.
- [23] Van Gogh, Vincent. The Olive Trees. 1889. The Museum of Modern Art, New York. <http://www.vangoghgallery.com/catalog/Painting/359/Olive-Trees-with-the-Alpilles-in-the-Background.html>. Accessed on 1 Jul 2014.
- [24] Andrei Tigora, Costin-Anton Boiangiu, "Image Color Reduction Using Iterative Refinement", *International Journal of Mathematical Models and Methods in Applied Sciences*, Volume 8, 2014, pp. 203-207
- [25] Costin-Anton Boiangiu, Alexandru Victor Ștefănescu, "Target Validation and Image Color Calibration", *International Journal of Circuits, Systems and Signal Processing*, Volume 8, 2014, pp. 195-202
- [26] Costin-Anton Boiangiu, Alexandra Olteanu, Alexandru Victor Ștefănescu, Daniel Rosner, Alexandru Ionut Egner (2010). „Local Thresholding Image Binarization using Variable-Window Standard Deviation Response” (2010), *Annals of DAAAM*

- for 2010, Proceedings of the 21st International DAAAM Symposium, 20-23 October 2010, Zadar, Croatia, pp. 133-134
- [27] Costin-Anton Boiangiu, Andrei Iulian Dvornic. "Bitonal Image Creation for Automatic Content Conversion". Proceedings of the 9th WSEAS International Conference on Automation and Information, WSEAS Press, pp. 454 - 459, Bucharest, Romania, June 24-26, 2008
- [28] Diana-Maria Popa, "Recolorarea Imaginilor"/"Image Recoloring", License Thesis, Unpublished Work, Bucharest, Romania, 2014.
- [29] Costin-Anton Boiangiu, Ion Bucur, Andrei Tigora - „The Image Binarization Problem Revisited: Perspectives and Approaches”, The Proceedings of Journal ISOM Vol. 6 No. 2 / December 2012, pp. 419-427.