The 8th International Symposium on Virtual Reality, Archaeology and Cultural Heritage
VAST (2007)
D. Arnold, F. Niccolucci, A. Chalmers (Editors)

GPU Spectral Viewer: analysing paintings from a colorimetric perspective

P. Colantoni, D. Pitzalis, R. Pillay & G. Aitken

Centre de Recherche et de Restauration des Musées de France
Palais du Louvre
Paris - France
colantoni@couleur.org, {denis.pitzalis,ruven.pillay,genevieve.aitken}@culture.fr

Abstract

Over the last fifteen years, multipsectral imaging has gained in importance and interest, especially in the field of Cultural Heritage, art investigation and conservation.

Extending the concept of scientific imagery such as colorimetric, infrared reflectography (IRR), ultraviolet (UV) and X-ray imaging applied to the study of paintings, multispectral imaging, coupled with high resolution and HDR (high dynamic range) has significantly improved the scope and accuracy of the non-invasive scientific analysis that is possible.

In order to exploit and study such multispectral data, a special GPU-based application using a custom color management process has been developed. In this paper we will present its innovative capabilities in image processing and visualization which enhance the study of works of art.

Categories and Subject Descriptors (according to ACM CCS): I.4.0 [Image Processing and Computer Vision]: Image processing software

1. Introduction

During the CRISATEL European Project [RSPL05] the Centre de Recherche et de Restoration des Musées de France had the opportunity to use a very innovative camera to acquire works of art. This new camera is able to produce ultra high definition (12.000*x*20.000 pixels) in 13 different wavelengths of light from ultraviolet (UV, $< 0.4\mu$m) to near infrared (NIR, 1.0μm) covering all the visible spectrum. The accuracy of these images with the knowledge of the spectral reflectance properties of the different materials associated with each pixel in the digital image allow us to do innovative studies and experiments such as, for example, virtually simulate the original colors or perform a varnish removal. These techniques in multispectral analysis are available via a very complex image processing pipeline that give us the possibility to investigate a painting in ways that were totally unknown until now.

The necessity to analyse the famous painting “Mona Lisa” by Leonardo da Vinci, and several hundred other multispectral acquired works of art, have pushed us to develop a special GPU based processing tool to better perform our advanced analysis. A GPU based solution can provide significantly superior performance to common CPU driven software for certain kind of algebraic computation. In this way we are able to provide to the user a tool for image processing that is both precise and usable, able to grant the high performance that an art historian expert needs in order to analyse multispectral data. An important development in this study has been the color management process.

2. Software modules

Our GPU Spectral Viewer is a software based on an *ad hoc* developed 2D rendering engine which uses OpenGL. This engine was chosen because it can run on several different architectures (Linux, *nix, MacOSX and various Windows) and present the only limitation is that it needs a Shader Models 3.0 compatible graphics card (e.g. latest *n*VIDIA^{TM}) in order to work properly. A custom C++ library based on a composition engine was also developed for this purpose. This library allows us to build the application using a new set of widgets rendered using textures and frame buffer objects. The result is a fast and portable GUI built with a specific design which use eye candy transparency and 3D effects. In order to better understand the function of the different software modules built in this application we have first to introduce the complex color computations used for this software. The GPU based calculation process has been described in a previous, related, publication [CPP*06].

2.1. Color computation

The *XYZ* system is based on the response curves of the three color receptors of the human eye. Since these differ slightly from one person to another, the *CIE* consortium has defined a "standard observer" whose spectral response corresponds, more or less, to the experimentally observed average response of the human population. This objectifies the colorimetric determination of colors.

The CRISATEL camera produces 13 channel images which correspond to the following frequencies: 400, 440, 480, 520, 560, 600, 640, 680, 720, 760, 800, 900 and 1000μm. Only the 10 first planes interact with the visible part of the spectrum (they are visible to the human eye). Considering this, the *XYZ*() tri-stimulus formulae is:

$$\begin{cases} X = \sum_{\lambda=400}^{\lambda=760} x(\lambda).R(\lambda).V(\lambda).L(\lambda) \\ Y = \sum_{\lambda=400}^{\lambda=760} y(\lambda).R(\lambda).V(\lambda).L(\lambda) \\ Z = \sum_{\lambda=400}^{\lambda=760} z(\lambda).R(\lambda).V(\lambda).L(\lambda) \end{cases} \quad (1)$$

where $R(\lambda)$ is the reflectance spectrum, $V(\lambda)$ is the varnish transmittance and $L(\lambda)$ is the light spectrum (used as illuminant). Using these formulae we can compute the resulting *XYZ* values for each pixel of the source image. One of the modules of our application can simulate a varnish removal using the known transmittance properties of the varnish based on this formulae. Of course, the obtained values are only an approximation of the values that would be measured in front of the original painting.

The $L^*a^*b^*$ color space is an attempt to linearise the human perceptibility of unit vector color differences. The main objective of this color space is to provide a scientific measurement of color. The colorimetric difference measure is called ΔE described, in literature, in several different forms; we propose the use three of them:

- ΔE_{1976} (the Euclidean distance [CIE95]);
- ΔE_{1994} (distance based on psycho-visual experiment [CIE95, LCR01]);
- ΔE_{2000} (complex distance also based on psycho-visual experiment [LCR01]. This one, in particular, needs to use more computational power because it uses complex mathematical functions: *cosinus*, *sinus*, *power*...).

In order to provide calibrated color (similar to the effective color stimuli) on a specific display device we must also make sure we characterise it [VT99]. Visualization device calibration is based on measurement of the input (e.g. *RGB* input values of the display device) and the output (e.g. *XYZ* values obtained on the display device using a spectrometer) values. This characterisation process is used to calibrate the visual quality of the color generated by our software using a custom virtual illuminant (fig. 2). A different module is in charge of the calibration process. It gives us the ability to generate a 3D Look Up Table (LUT) that will be used to compute the *RGB* values needed to display, on any device (screen, printer...), the closest color stimuli corresponding to the mathematical $L^*a^*b^*$ model.

As we have said before, all these computations are done using the accelerated GPU, providing a real time interaction environment where the user can choose different parameters and directly see the results.

2.2. Multispectral information visualization

The multispectral information visualization module is used to display the reflectance image as a set of 16bit grey level images. Different LUTs and a gamma correction factor can be modified during the visualization process (fig. 1).

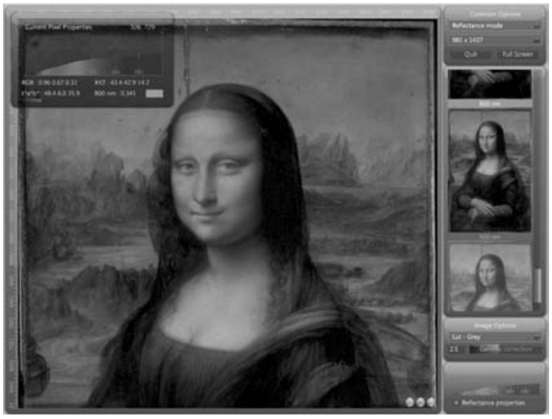

Figure 1: *Reflectance image visualization*

2.3. Color reconstruction

Based on *XYZ* and $L^*a^*b^*$ color spaces and on a custom color management process (which will be described in the next section), this module allows the user to interact in real time with a virtual environment lighting [CBR03, CPP*06]. All the generated and displayed images are HDR images (High Dynamic Range). The graphic interface associated with this module includes:

- an *RGB* image viewer
- a virtual light selector (based on standard illuminant or customised spectra)
- a virtual varnish simulator

- an $L^*a^*b^*$ color cloud viewer, including the gamut of the current display (we use it in order to see if all the colors are visible on the current display device)
- an *RGB* color cloud viewer (fig. 2).

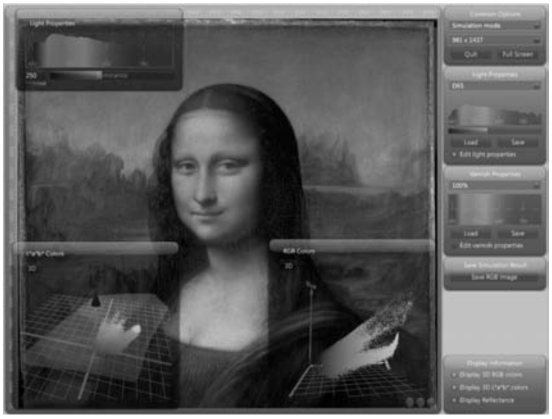

Figure 2: *Color reconstruction*

2.4. Color measure

The GPU processing algorithm presented in [CPP*06] is used in this module. The difference from the original one is in the resulting $L^*a^*b^*$ color image. In this version we use the white reference as virtual illuminant to do the computation. Using this new $L^*a^*b^*$ image we can provide a more usable and precise interface for color selection (fig. 3).

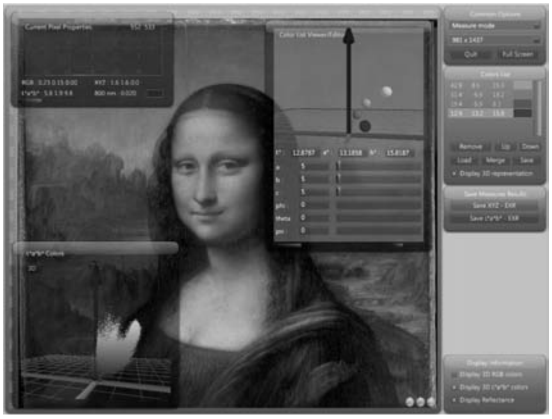

Figure 3: *Color measure*

2.5. Sensor simulation

The analysis of a painting using colorimetry can not be precise using a simple color management based on spectral properties on different virtual lights (metamerism [S05,B01] analysis for example). For this reason we provide the possibility to choose interactively the sensitivity curves of a virtual *RGB* simulated sensor. These sensitivities can be based on an existent sensor (like in fig. 4) or defined by the user.

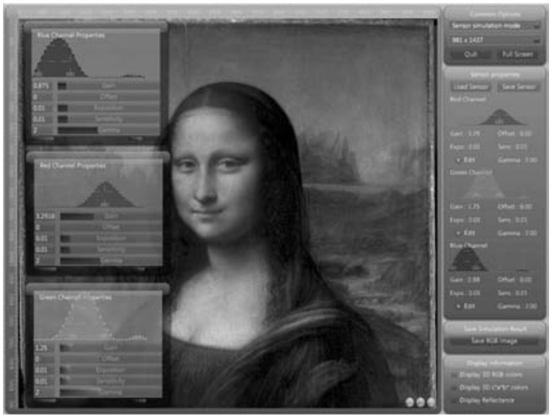

Figure 4: *Simulated sensitivity of a Canon 20D*

2.6. Color segmentation

Color segmentation is very important in studying works of art because it helps us to define the "palette" of an artist, to characterise his colors in order to help the expert in classification and to give more exploitable statistical data for matching algorithms. The segmentation process investigated for this application is based on color clustering and threshold approaches. It is time consuming especially if we use complex color distance (e.g. measured with ΔE_{2000}) and a large set of reference colors. The two methods are described here:

- *threshold*: associate a ΔE threshold (ΔE_{76},ΔE_{94} or ΔE_{2000}) to a reference color. This generates a 3D primitive (sphere, ellipsoid or cylinder) in the $L^*a^*b^*$ color space. The colors appearing in this 3D volume are associated to the same label (fig. 5);
- *clustering*: use an algorithm which isolates and labels individual colors by finding in a color reference set the closest match. With this step we provide a full segmentation of the image.

2.7. Color cloud analysis

The color cloud analysis can be done using a 3D representation of the colors in the $L^*a^*b^*$ space (fig. 6). This module is useful to define a statistical approach to the color distribution on the painting and it includes:

- the color cloud
- the selected color (associated to the corresponding 3D primitives)
- the 2D and 1D histogram (showing the color density)
- a pigment injection utility (with the 3D representation of pigments identified by their reflectance)

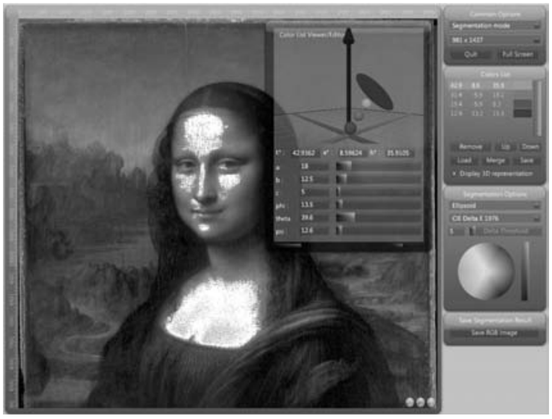

Figure 5: *Segmentation*

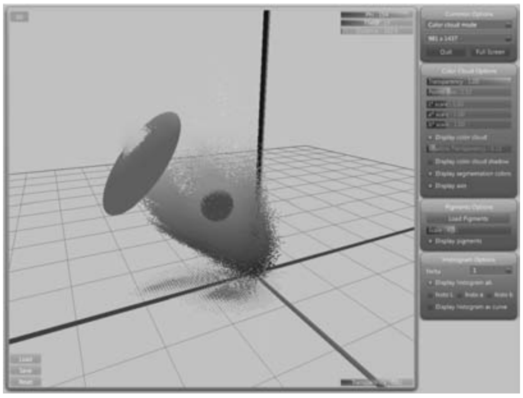

Figure 6: *Color cloud analysis*

2.8. Luminance elevation map

This module is very useful for experts that want to analyse separately each color component (luminance for example). A color component (L^*, a^*, b^*, X, Y, Z , R, G, B) is represented as the Z value in an elevation map where the reconstructed image is mapped (fig. 7).

2.9. Results

In this section we provide some benchmarks, such as the computation time of different processes for a 1963 × 2874 pixel ×13 planes (using half float internal representation - 16bit per channel) reflectance image. We used a 3D graphics card based on an nVIDIA^{TM} G80 graphics processor (GeForce 8800 GTX) with the following specifications: 575MHz GPU core, 1350MHz stream processors and 768Mb of 1800MHz RAM (SDRAM DDR3).

All these operations (table 1) correspond to a per-pixel process which requires the execution of a pipeline of several mathematical complex functions and vector calculations [FM04]. As the GPU is more suitable for this kind of operation, the results obtained are improved in comparison to using only the CPU.

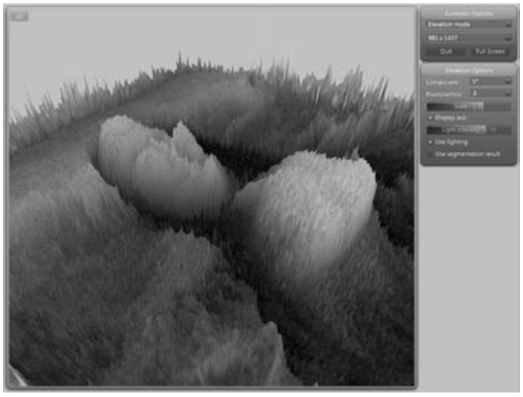

Figure 7: *Elevation map. We can easily identify areas of higher luminance*

processing type	Time	Images/s	Pixels/s
color simulation	0.010311149	96.9824	$547 * 10^6$
sensor simulation	0.00648231	154.266	$870 * 10^6$

Table 1: *Simulation Measure Process and Sensor Simulation Process*

colors	ΔE	Time	Images/s	Pixels/s
1	1976	0.000652175	1533.33	$8650 * 10^6$
1	2000	0.003529765	283.305	$1598 * 10^6$
5	1976	0.005996138	166.774	$940 * 10^6$
5	2000	0.043635729	22.917	$129 * 10^6$
10	1976	0.01042655	95.909	$541 * 10^6$
10	2000	0.086909666	11.5062	$65 * 10^6$

Table 2: *Segmentation process*

Table 2 indicates that the GPU performs very well in this particular case study especially if we only segment one color. Nevertheless a further investigation shows that the process time growth linearly with the number of color references used.

3. Color management process

Our multispectral calibration process gives us an accurate $L^*a^*b^*$ representation of the image. However, in order to display this as authentically as possible, display device calibration is necessary and has been integrated into our color management pipeline.

3.1. Device characterization

The characterization process describes certain aspects of a device such as the color correlation, the chromacity shift or even the gamut. Characterization is based on measurements of input values (e.g. *RGB* input values of a CRT monitor) and output values (e.g. *XYZ* or $L^*a^*b^*$ values measured on a CRT screen using a colorimeter or spectrometer) (figure.8).

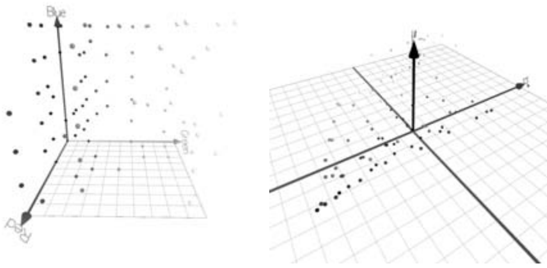

Figure 8: *Characterization process from RGB to $L^*a^*b^*$*

Traditionally a model of characterization (or forward model) is based on an interpolation or an approximation method. We found that radial basis function interpolation (RBFI) is the best model for this characterization problem.

RBFI is an interpolation [CBC*01] scheme for arbitrarily spaced tabulated data. The idea is to build a function s whose graph passes through the tabulated data and minimizes a bending energy function.

We want to interpolate a 3D valued function $f(R,G,B) = (L,a,b)$ by $s(R,G,B)$ given the set of values $f = (f_1,...,f_N)$ at the distinct points $X = x_1,...,x_N \subset \Re^3$. We choose $s(R,G,B)$ to be Radial Basis Function of the form:

$$s(x) = p(x) + \sum_{i=1}^{N} \lambda_i \phi(|x - x_i|) \quad x \in \Re^3$$

where p is a polynomial of degree at most k, λ_i is a real-valued weight, $|.|$ denotes the Euclidean norm, ϕ is a basic function, $\phi : \Re^3 \longrightarrow \Re$, and $|x - x_i|$ is the distance between x and x_i. An RBF is a weighted sum of translations of a radially symmetric basic function augmented by a polynomial term. Different basic functions can be used. In this context we have used 4 real functions $\phi(r)$ where r is the distance, in radius, from the origin:

- the biharmonic ($\phi(x) = r$)
- the triharmonic ($\phi(x) = r^3$)
- the thin-plate spline 1 ($\phi(x) = r^2 log(r)$)
- the thin-plate spline 2 ($\phi(x) = r^2 log(r^2)$)

The used basic function depends on the display device which is characterized.

The RBF interpolant $s(x)$ is defined by the coefficients of the polynomial p and the weights λ_i. Given the interpolation values $f = (f_1,...,f_N)$, we seek the weights λ_i so that the RBF satisfies

$$s(x_i) = f_i \quad i = 1,...,N \tag{2}$$

Because this gives an under-determined system, i.e., there are more parameters than data, the orthogonality or side conditions

$$\sum_{j=1}^{N} \lambda_j p(x_j) = 0 \tag{3}$$

for all polynomials p of degree at most k are further imposed on the coefficients $\lambda = (\lambda_1,...,\lambda_N)$.

Let $p_1,...,p_l$ be a basis for polynomials of degree at most k and let $c = (c_1,...,c_l)$ be the coefficients that give p in term of this basis. Then equations 2 and 3 may be written in matrix form as

$$\begin{pmatrix} A & P \\ P^T & 0 \end{pmatrix} \begin{pmatrix} \lambda \\ c \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix} \tag{4}$$

where

$$A_{i,j} = \phi(|x_i - x_j|), \quad i,j = 1,...N,$$

$$P_{i,j} = p_j(x_i), \quad i = 1,...N, j = 1,...l.$$

Solving the linear system (4) determines c and λ, hence $s(x)$.

This RBF interpolation allows us to build the color gamut of a display device. To do that we use an *RGB* sampling and we compute for each *RGB* color the corresponding $L^*a^*b^*$ values (fig.9).

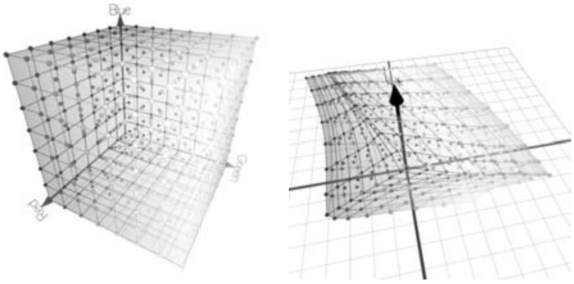

Figure 9: *RGB Sampling and the corresponding gamut in $L^*a^*b^*$*

3.2. Optimal color learning database selection

In order to increase the reliability of the model we introduced a new way to determine the learning data set of this interpolation (e.g. the color patches measured on the screen by the colorimeter). We found that our interpolation model was most efficient when the learning data set (the set of colors used to initialize the interpolation) was regularly distributed in our destination color space ($L^*a^*b^*$). This new method is based on a regular 3D sampling of $L^*a^*b^*$ color space associated by a forward - backward process which allows us to find the optimal RGB colors associated.

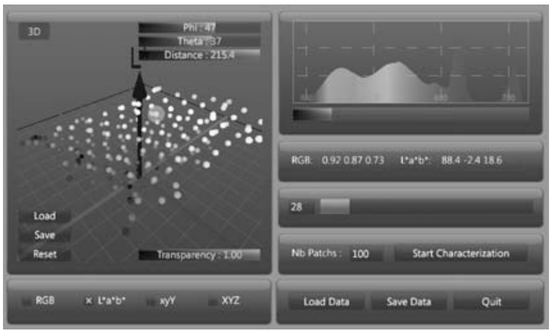

Figure 10: *Color characterization module*

This technique needs to select incrementally the RGB color patches that will be integrated into the learning database. For this reason an independent software tool has been developed, which is able to drive a colorimeter (fig.10). It is also able to measure a set of 64 test patches used in order to determine the accuracy of the model. Table 3 shows the results that we obtained with 4 different displays (2 LCD and 2 CRT) using 2 colorimeters (a basic GretagMacbeth EyeOne Display colorimeter and professional GretagMacbeth EyeOne Pro colorimeter). The accuracies achieved from a 200 color patch learning database are high especially when using the EyeOne Pro.

Display	colorimeter	ΔE_{mean}	ΔE_{max}
Mitsu SB230 (CRT)	EO Pro	0.510256	3.93594
EsayNote (LCD)	EO Pro	0.810274	2.04226
Mitsu SB2070 (CRT)	EO display	1.11172	6.73836
HP 2335 (LCD)	EO Display	1.33415	3.95475

Table 3: *Forward model accuracy*

The main advantages of this method are:

- it works with all kind of display devices (CRT, LCD, video projector, etc...) because it uses a 3D interpolation process
- it is very accurate even with a small learning database (but obviously bigger is better)
- the measurement process is fast (less than 3 minutes with 200 color patches chart and an EyeOne Pro colorimeter).

Note: all the accuracy errors shown in this paper have been obtained using custom model validation software (fig.11).

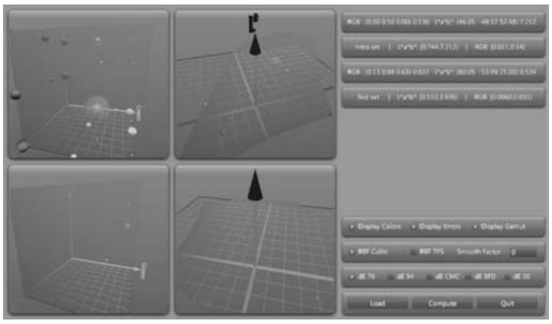

Figure 11: *Color error validation module*

3.3. Backward model

While the characterization process defines the relationship between the device “color space” and the *CIE* system of color measurement, our problem is different because we want to find, for a given $L^*a^*b^*$ (computed by the GPU during the simulation step), the *RGB* corresponding values (for a given display device previously characterized).

This backward model could be base on the same interpolation methods previously presented but a different and more accurate method [CSB05] have demonstrate better results. This new method is based on the fact that if our forward model is very accurate then it is associated with our optimal patch database. Basically it is an hybrid method: tetrahedral interpolation associated with an over-sampling of the *RGB* cube (fig.12) (which allows us to build a larger structure). We have chosen the tetrahedral interpolation method because of its geometrical aspect (this method can be associated with a gamut clipping algorithm).

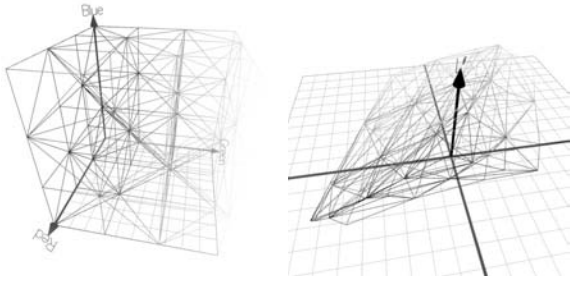

Figure 12: *Tetrahedral geometric structure in RGB and the corresponding structure in $L^*a^*b^*$*

We build the initial tetrahedral structure using an over sampling of the *RGB* cube. This over sampling process uses the forward model in order to compute the corresponding structure in the $L^*a^*b^*$ color space. Once this structure has been built we can compute for an unknown C_{Lab} color the associated C_{RGB} color in two steps. First, the tetrahedron which encloses the point C_{Lab} to be interpolated should be found (the scattered point set is tetrahedrized). Then, an interpolation scheme is used within each tetrahedron. More precisely, the color value C of the point is interpolated from the color values C_i of the tetrahedron vertices. The linear interpolation is the follows:

$$C = \sum_{i=0}^{3} w_i C_i$$

The weights can be calculated by $w_i = \frac{V_i}{V}$ with V the volume of the tetrahedron and V_i the volume of the sub-tetrahedron according to:

$$V_i = \frac{1}{6}(P_i - P)[(P_{i+1} - P)(P_{i+2} - P)]; i = 0, ..., 3$$

where P_i are the vertices of the tetrahedron and the indices are taken modulo 4.

Table 4 shows the results that we obtained with the same displays as in table 3. If we normalise our R, G and B values between 0 and 1 the ΔRGB_{mean} computed are good even with a basic colorimeter.

Display	Display type	ΔRGB_{mean}	ΔRGB_{max}
Mitsu SB230 (CRT)	EO Pro	0.0098113	0.029797
EsayNote (LCD)	EO Pro	0.0167477	0.0975634
Mitsu SB2070 (CRT)	EO display	0.0143968	0.0658026
HP 2335 (LCD)	EO Display	0.0131345	0.041826

Table 4: *Backward model accuracy*

3.4. Gamut mapping

The aim of gamut mapping is "to ensure a good correspondence of overall color appearance between the original and the reproduction by compensating for the mismatch in the size, shape and location between the original and reproduction gamuts" [M98].

The $L^*a^*b^*$ computed color can be out of gamut (e.g. the destination display can not generate the corresponding color). To achieve our objective, we propose a solution which simultaneously processes gamut mapping and color space conversion without any complex computation. This geometrical gamut clipping method is based on the pre-computed tetrahedral structure (generated in our backward model) and more especially on the surface of this geometrical structure (fig.12).

In order to determine the intersection point between the input color and the gamut we need to define a target point. This point will define, with the input color, a segment which intersects the 2D shape of the gamut. This intersection will be the result color (in the $L^*a^*b^*$ color space). The chosen target point used here is an achromatic $L^*a^*b^*$ color with a luminance of 50 (fig.13).

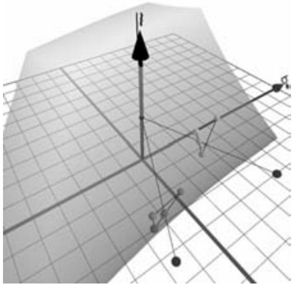

Figure 13: *Color clipping process*

3.5. GPU-based implementation principles

The GPU implementation of the previously describe color management process is based on a 3D LUT (Look Up Table) built using a 3D texture. The size of this 3D LUT is defined by the bounding box of the destination gamut (fig.9), our goal is to build cubic voxels (for a cubic bounding box we use a 32 × 32 × 32 3D texture).

The defined 3D LUT allows us to associate $L^*a^*b^*$ values with RGB values. Using the fact that the number of voxels is adequate to consider this process locally linear we can use a 3D tri-linear interpolation in order to find for a $L^*a^*b^*$ values the corresponding RGB values.

This process is perfectly suited to GPU implementation and allows us to compute several hundred thousand colors per second (1000 MPixel/s with an nVIDIA^{TM} GeForce 8800GTX).

4. The "Mona Lisa" case study

At the beginning of 1500 Leonardo da Vinci, an Italian painter, finished the "Mona Lisa" or "La Gioconda". This is an oil painting on poplar wood. Few other works of art have been subject to as much study, mythologizing and analysis. At the C2RMF we had the opportunity to investigate this work of art with the most innovative technologies.

Using the "Luminance elevation map" module (fig. 7), for example, we obtain a global representation of the luminance on the pictorial layer. This not only shows the area with higher luminance, but also give us an idea of the imperceptible variation of luminance between the sky and the mountains in the landscape; the high peaks in the border represents the "barbe", a quantity of pigment uncovered from the varnish that slightly reduces the painting luminance.

The "Reflectance image visualization" module (fig. 1) give us the possibility to analyse each multispectral layer. To search, for example, a drawing under the paint layer done using graphite we have to search in the infrared (NIR, between 850 and 1050μm) images. In our analysed painting there are few of these, notably under the green of the landscape, close to the "Mona Lisa" head. Moreover, if we compare the simulated *D*65 image with all the different spectral images, we can find the order in which the different painting layers have been applied.

Another example of the results achieved using the multispectral technology is the analysis by "Color segmentation". Once we have obtained the color map we can start investigating the relationships between the different color groups looking at their position in the $L^*a^*b^*$ space: if they are aligned in the $L*$ axis, this would mean that the pigments have been mixed with a medium or with black or white pigments in order to create a gradient; if the alignment is on the $a*$ or $b*$ axis, it means that more than one pigment is present.

Using the previous module we also have been able to identify some pigments based on their metamerism: in the top of the wood there is a restored area; in that area the metamerism is different to than in the rest of the sky. More in-depth studies using infrared images and cracks analysis, have shown that in effect, the top layer of that area has been restored with a different pigment to that used by Leonardo.

More analysis has been done on multispectral images and they are presented in [C2RMF06].

Conclusion

The main aim of this work has been to investigate and develop a new tool to help experts in Cultural Heritage in the study and analysis of multispectral digital images. The work focused in developing something innovative for that domain using new technologies, like GPU based computation. The potential offered by color spectra decomposition techniques opens a new way of study of works of art.

In effect multispectral imaging system is still considered a cutting-edge technology not for the good results we are able to achieve interpreting it data, but because it is still a specialist technology.

The Gioconda not only gave inspiration to Leonardo, but also to our group to develop an application able to discover details otherwise invisible to the naked eye. We have demonstrated that this kind of applications represent a valuable aid in the field of art conservation and restoration and requires more in-depth investigation. Now that the procedures for image process is well known and that results have demonstrated their enhanced value compared to standard imaging results, it is sure that more effort will be put in order to make a more accessible system for non specialist use.

References

[B01] Berns R.S.: The science of digitizing paintings for color accurate image archives: a review. In *Journal of Imaging Science and Technology 45* (2001), pp. 305–325.

[C2RMF06] Centre de Recherche et de Restauration des Musées de France, coll.: Au coeur de La Joconde, Léonard de Vinci décodé. Gallimard, Musée du Louvre Edition, (2006), ISBN 2-07-011833-9.

[CBC*01] Carr J., Beatson R., Cherrie J., Mitchell T., Fright W., McCallum B., Evans T.: Reconstruction and representation of 3d objects with radial basis functions. In *SIGGRAPH* (2001), pp. 12–17.

[CBR03] Colantoni P., Boukala N., Rugna J. D.: Fast and accurate color image processing using 3d graphics cards. In *Vision Modeling anv Visualization, VMV* (2003), pp. 383–390.

[CIE31] CIE, Commission Internationale de l'Eclairage: International Congress on Illumination. *Proceedings, International Congress on Illumination* (1931), Cambridge.

[CIE95] CIE, Commission Internationale de l'Eclairage: Industrial colour-difference evaluation. *Proceedings, International Congress on Illumination* (1995), Tech. Rep.

[CPP*06] Colantoni P., Pillay R.,Lahanier C. and Pitzalis D.: Analysis of multispectral images of paintings. In *EUSIPCO* (2006), Florence, Italy.

[CSB05] Colantoni P., Stauder J., Blondé L.: Device and method for characterizing a colour device. In *European Patent, EP 05300165.7* (2005), Thomson Corporate Research.

[FM04] Fung J., Mann S.: Computer Vision Signal Processing on Graphics Processing Units. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing* (2004), Montreal, Quebec, Canada, May 17–21, 2004.

[LCR01] Luo M., Cui G., Rigg B.: The development of the CIE 2000 colour-difference formula: CIEDE2000. In *Ph.D. Thesis, University of Derby* (1998), Derby.

[M98] Morovic J.: TTo Develop a Universal Gamut Mapping Algorithm. In *Color Research and Application 26* (2001), 340–350.

[RSPL05] Ribés A., Schmitt F., Pillay R., Lahanier C.: Calibration and spectral reconstruction for crisatel: an art painting multispectral acquisition system. In *Journal of Imaging Science and Technology 49* (2005), 563–573.

[S05] Staniforth S.: Retouching and colour marching: the restorer and metamerism. In *Journal of Studies in Conservation 30* (1985), 101–111.

[VT99] Vrhel M. J., Trussell H. J.: Color device calibration: a mathematical formulation. In *IEEE Transactions on Image Processing 8*, 12 (1999), 1796–1806.