A Genetic Programming Approach to Deriving the Spectral Sensitivity of an Optical System

Marc Ebner

Universität Würzburg, Lehrstuhl für Informatik II
Am Hubland, 97074 Würzburg, Germany
ebner@informatik.uni-wuerzburg.de
http://wwwi2.informatik.uni-wuerzburg.de/staff/ebner/welcome.html

Abstract. In color image processing, several sensors are used which respond to the light in the red, green and blue parts of the spectrum. When working with color images taken by an optical system it is very important to know the sensitivity of the entire optical system. The optical system consists of the sensor, lens and any filters which may be used. The response characteristics of the lens and filters can be measured inside the laboratory. However, for many digital cameras it is not clear how the sensors contained inside the camera respond to light. This information may not be available from the manufacturer of the camera. Even if we knew the response characteristics of the sensor, it may not be clear what algorithms are employed by the manufacturer before the data is finally stored as an image file. We show how genetic programming may be used to obtain the sensor response functions using a single image from a calibration target as input together with the reflectance data of this calibration target.

1 Motivation

The sensor array contained inside a digital camera measures the incident light. For many digital cameras, data about how the sensor responds to light is not publicly available because this data may not be released by the manufacturer. Knowing how the RGB values stored inside the image depend on the irradiance entering the lens of the camera is very important for colorimetry [1,2] and the research area of color constancy [3,4,5,6,7]. We show how genetic programming [8,9,10] can be used to obtain the sensor response functions using an image from a calibration target as input.

A standard sensor consists of a single type of light sensitive sensor and differently colored filters which are placed in front of the sensor to make it respond to light in the red, green and blue parts of the spectrum. These sub-pixel sensors are often arranged in a pattern which is called a Bayer pattern [11]. A full color image is obtained by interpolating the data from adjacent sensors. Other types of sensors where all three components of the incident light are measured at the same position also exist. Imaging chips which measure more than the three components red, green, and blue have also been developed.

The response of a sensor for a given wavelength is proportional to the irradiance falling onto the sensor times the sensitivity of the sensor for that given wavelength. The energy measured by the sensor is obtained by integrating over all wavelengths. If we know the irradiance falling onto the optical system and also know the energy measured by the sensor, then we can formulate an optimization problem in order to find the sensitivities of the optical system. Data is typically measured at intervals of 10nm. Therefore, we have 32 data points inside the visible range from 390nm to 700nm. Finding the sensitivity of an optical system is basically an optimization problem where the 32 sensitivities at positions $\{390\text{nm}, 400\text{nm}, ..., 700\text{nm}\}$ have to be found.

A number of different problems in image processing have been addressed by the evolutionary computation community. Zhang and Ji [12] as well as Rodehorst and Hellwich [13], have used a genetic algorithm for camera calibration. An evolutionary strategy was used by Cerveri et al. [14] to obtain the internal or external parameters of a camera. Johnson et al. [15] used a genetic algorithm for projector calibration. Carvalho et al. [16] has used a least squares approach to obtain the response function of a sensor. A genetic algorithm was used to maximize the prediction ability of an extended generalized cross-validation measure.

Ebner [17] was the first to apply an evolutionary strategy [18,19] to obtain the sensor response curves of an optical system. Due to the type of problem, constraints have to be enforced in order to solve it. Ebner has shown that best results were obtained by enforcing the constraints directly on the genotype. We will show how genetic programming can be used to find a solution to this type of problem. By properly choosing the set of terminal symbols and the set of elementary functions, constraints are enforced naturally.

This article is structured as follows. First, we describe the model of color image formation. We then explain how finding the response curves of an optical sensor can be defined as an optimization problem. Next, we show how genetic programming can be used to find a solution to this problem. We performed experiments on simulated data where the ground truth is known and also obtained the sensor response curves for two commercially available digital cameras. Conclusions are given at the end of the paper.

2 Theory of Color Image Formation

Suppose that we use our optical system to take an image of a calibration target illuminated by a light source of known spectral power distribution. A calibration target consists of many differently colored patched of known reflectances. The optical system measures the light which is reflected from the calibration target. Let N_p be the number of colored patches on the calibration target. Let $E(p, \lambda)$ be the irradiance which is falling onto patch p at wavelength λ. Some of the irradiance is absorbed by the patch, the remainder is reflected and may enter the lens of the camera. Let $R(p, \lambda)$ be the reflectance of patch p at wavelength λ. We will assume that the optical system is using three sensors which measure the light in the red, green and blue parts of the spectrum. Let $S_i(\lambda)$ be the

sensitivity of the sensor $i \in \{r, g, b\}$. Then the energy $I_i(p)$ measured by sensor i for patch p is modeled as

$$I_i(p) = \int S_i(\lambda) R(p, \lambda) E(p, \lambda) d\lambda. \tag{1}$$

The integration is performed over all wavelengths to which the sensor responds. This model of color image formation is used by many algorithms in colorimetry and color constancy [20,21,22].

We now assume that the calibration target is a Lambertian reflector, i.e. an object which reflects the incident light in all directions. Let the radiance given off by the light source which illuminates the calibration target be $L(\lambda)$. Then the irradiance falling onto the calibration target is simply $E(p, \lambda) = L(\lambda) \cos \alpha$ where α is the angle between the normal vector $\mathbf{n}_S$ describing the orientation of the calibration target and the unit vector $\mathbf{n}_L$ pointing into the direction of the light source from the object patch. Hence, the energy $I_i(p)$ measured by the sensor i for object patch p is given by

$$I_i(p) = G(p) \int S_i(\lambda) R(p, \lambda) L(\lambda) d\lambda \tag{2}$$

where $G(p) = \mathbf{n}_S \mathbf{n}_L^T(p) = \cos \alpha$ is a geometry factor. The geometry factor scales all channels equally.

Digital cameras usually do not save the energy data measured by the sensors. Most produce an output image using the sRGB color space [23]. If the sRGB color space is used, then the measured data is stored in a non-linear way such that the non-linearity of the output device is compensated for. This is called a gamma correction. If we process such images of our calibration target, then this gamma correction needs to be undone such that the processed color data depends linearly on the measured data. Some digital cameras also allow the user to select that the raw measured data be stored in an image file. In this case, the raw data can be processed directly. From now on, we will assume that our optical system produces RGB color triplets c_i as output and that we have $c_i = I_i$.

3 Evolving the Sensitivities of an Optical System

We now show how evolutionary computation can be used to estimate the sensitivities of an optical system. Figure 1 shows the data flow which is used by our system. First an image of a calibration target is taken with the optical system. For our experiments we will be using a standard IT8 calibration target made by Wolf Faust. Such targets are frequently used for calibration of scanners or other optical systems. This calibration target consists of 22×12 colored patches at the top and 24 different gray patches at the bottom. It comes with a complete set of reflectances for each of the patches for wavelengths 390nm to 700nm in steps of 10nm.

Once an image of the calibration target has been taken, the pixel values of each patch are averaged in order to obtain a single color measurement $\mathbf{c}(p) =$

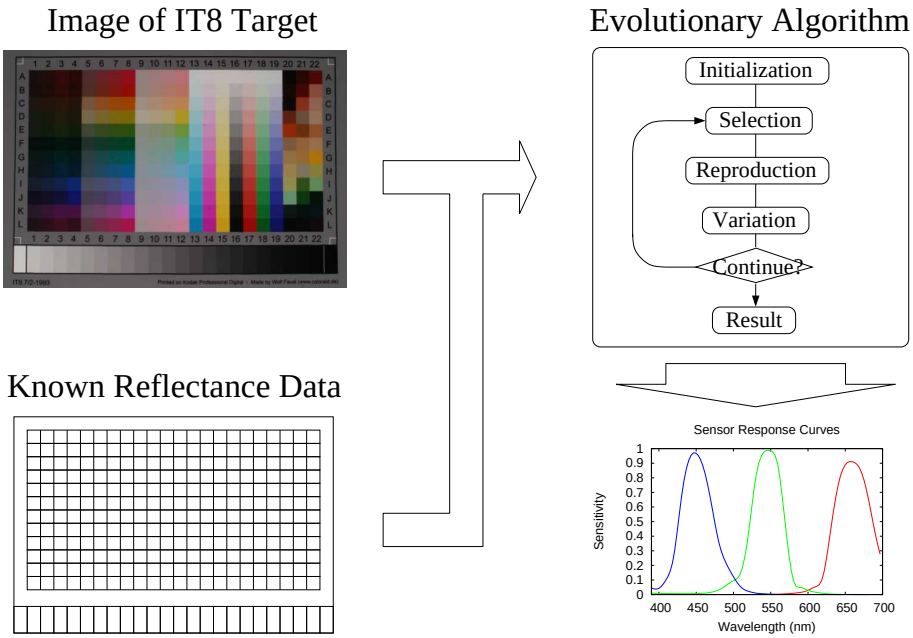

Fig. 1. Data flow of the method to obtain the sensitivities of an optical system. First, the optical system is used to take an image of the calibration target with known reflectances. The reflectance data is used by the evolutionary algorithm to compute the fitness of possible solutions to this problem. After several generations, the optimal sensitivities found by the evolutionary algorithm are output.

$[c_r(p), c_g(p), c_b(p)]$ for each patch p. Pixels close to the border of a patch are not included in the average as they are assumed to be linear combinations of the adjacent colors. Thus, we now have a virtually noise free measurement $\mathbf{c}(p) = \mathbf{I}(p)$ for each patch p. The calibration target comes with known reflectance data $R(p, \lambda)$ for each patch p for each wavelength λ. Before we can solve Equation 2 for $\mathbf{S}(\lambda) = [S_r(\lambda), S_g(\lambda), S_b(\lambda)]$, we also need an estimate of the radiance $L(\lambda)$ which is emitted by the light source. One way to obtain the radiance is to measure it using a spectrometer. Another way is to use a light source which has a known spectral power distribution.

Digital cameras usually perform some kind of white balancing. They correct the image colors for the spectral power distribution of the illuminant. Most consumer cameras either perform automatic white balancing or allow the user to set one of several possible illuminants, such as sun, cloudy sky, neon light, light bulb or flash. Given such a camera, it is best to set the white balance to sun and then take an image of the calibration target on a sunny day. Professional cameras allow the user to choose a particular color temperature. In most cases, it is not known what processing is actually performed inside the camera to obtain the RGB color triplets from the measured data.

Since we have assumed that we took appropriate measures to control the illuminant and that the camera corrects for the type off illuminant used, we now have to solve the following equation to obtain $\mathbf{S}(\lambda)$.

$$\mathbf{c} = G \int \mathbf{S}(\lambda) R(\lambda) d\lambda \tag{3}$$

Note that the geometry factor G scales all color channels equally. It can be removed by computing chromaticities $\hat{\mathbf{c}}$.

$$\hat{\mathbf{c}} = \frac{1}{c_r + c_g + c_b} \mathbf{c} \tag{4}$$

We will be coding the sensor response curves $\mathbf{S}(\lambda)$ as the individuals of our evolutionary algorithm. Given an individual which represents a particular set of sensor response curves, we can then compute how well this set describes the actual set of response curves. In order to determine the fitness of an individual, we compare the measured chromaticities $\hat{\mathbf{c}}_M(p)$ which were obtained from the image of the calibration target with the theoretical chromaticities $\hat{\mathbf{c}}_T(p)$ which are computed using the known reflectances for all patches p.

The known reflectances $R_p(\lambda)$ are used to compute the theoretical chromaticities $\hat{\mathbf{c}}_T(p)$ for patch p. Let $\mathbf{S}(\lambda)$ be the sensor response curve represented by a particular individual. Then the theoretical response is given as

$$\mathbf{c}_T(p) = \sum_{\lambda \in \{390,\ldots,700\}} \mathbf{S}(\lambda) R_p(\lambda). \tag{5}$$

Let $\hat{\mathbf{c}}_T(p)$ be the corresponding chromaticity, i.e.

$$\hat{\mathbf{c}}_T(p) = \frac{1}{\sum_i c_{iT}(p)} \mathbf{c}_T(p). \tag{6}$$

The deviation E_{fit} between the theoretical and the measured response is our error measure

$$E_{\text{fit}} = \sum_p \left(\hat{\mathbf{c}}_T(p) - \hat{\mathbf{c}}_M(p) \right)^2. \tag{7}$$

In other words, we compute the sum of the squared differences between $\hat{\mathbf{c}}_T$ and $\hat{\mathbf{c}}_M$ over all 288 image patches of the calibration target. The error measure E_{fit} describes how well the sensitivities of any given individual match those of the optical system. We want to minimize this error measure. A perfect individual would have $E_{\text{fit}} = 0$.

4 Obtaining the Sensitivities of an Optical System using Genetic Programming

Ebner [17] has previously used an evolutionary strategy to obtain the sensor response curves $S_i(\lambda)$ which closely match the sensor response curves of an optical system. An evolutionary strategy is usually used for parameter optimization. For this type of problem, an individual is simply a vector of floating point

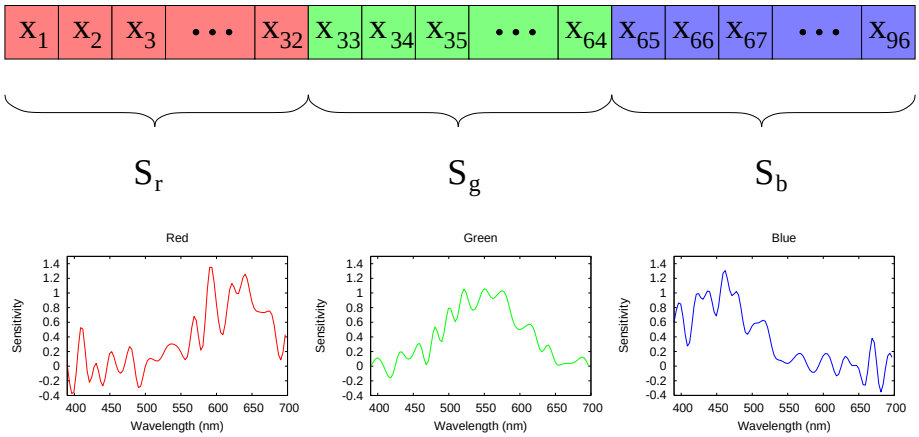

Fig. 2. Evolutionary strategy representation. The sensitivity of the three sub-sensors is stored consecutively inside the genotype. The drawback of this representation is that two adjacent sensitivities are independent from each other leading to a response function which may not necessarily be a smooth response function.

values which represents the sensitivities of the three sub-sensors at positions $\{390\text{nm}, 400\text{nm}, ..., 700\text{nm}\}$. Such an individual is shown in Figure 2. Due to the type of problem, the search space has to be constrained in order to guide evolution into the correct part of the search space. Here, we have several constraints. The first constraint is that the real sensor response curves are positive for all wavelengths λ, i.e. we have $S_i(\lambda) \geq 0$. Another constraint is that the sensor response curve is smooth without any discontinuities. Due to the computation of chromaticities, we also have the constraint that a uniform scaling of all parameters will not change the result. These constraints can be enforced either through the fitness function or through a repair mechanism on the genotype. Ebner [17] showed that enforcing all the constraints directly on the genotype produced best results.

Instead of encoding an individual as a floating point vector and then enforcing the constraints on the genotype, one may also use a more natural representation for this type of problem. The sensitivity of a sensor is usually Gaussian shaped. One can consider the sensitivity as a combination of Gaussians. This leads us to a genetic programming representation where the terminal symbols are Gaussians which have a particular position and standard deviation inside the visible spectrum and the set of elementary functions simply consists of the addition function. This representation is shown in Figure 3. The nodes are Gaussian functions which depend on the wavelength λ. The internal nodes are used to combine these Gaussian functions.

The set of elementary functions and terminal symbols is shown in Table 1. The terminal symbol $sG(\mu, \sigma)$ computes the following function.

$$sG(\mu, \sigma) = se^{-\frac{(\lambda-\mu)^2}{2\sigma^2}} \tag{8}$$

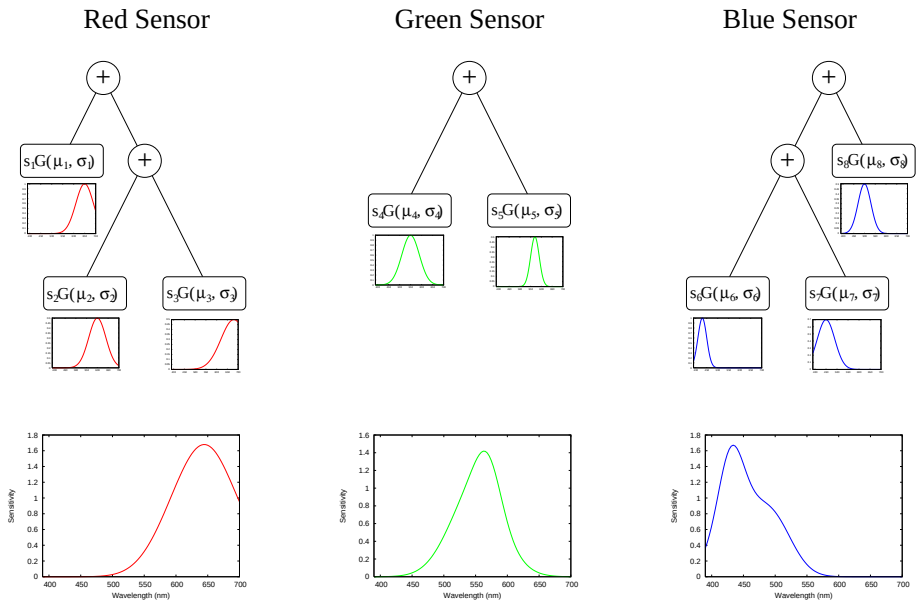

Fig. 3. Genetic programming representation. The response function of a sensor consisting of three sub-sensors responding to light in the red, green, and blue part of the spectrum is represented by three trees.

Table 1. Set of elementary functions and terminal symbols.

Name	Symbol	Arity	Internal Variables
Gaussian	G	0	(s, μ, σ)
Addition	+	2	none

The three variables s, μ and σ are stored inside each node. The internal parameter s specifies the strength of the Gaussian, μ specifies its position within the visual spectrum and σ specifies the standard deviation of the Gaussian. Addition is used as the only elementary function. This representation allows us to naturally enforce the constraints. The evolved sensor response curves are simply added Gaussians. Therefore, the evolved sensor response curves are smooth and also fulfill the constraint that the curves are positive for each wavelength λ.

Individuals of the first generation are generated randomly. We then select one of the genetic operators at random. The list of genetic operators are shown in Table 2. Several operators change the structure of the individual, i.e. the trees, while one evolutionary strategy type of mutation operator modifies the internal parameters of all nodes. Offspring are generated until the new population is filled. This process is then iterated for several generations.

Table 2. Genetic programming operators.

Name of Operator	Method to generate offspring
Mutation-ES	Evolutionary strategy type of mutation. All nodes of the individual are mutated by adding Gaussian distributed random numbers to the internal parameters. Each internal parameter x has an associated standard deviation δ which is mutated using $\delta := \delta e^{N(0,\tau)}$. The parameter x is then mutated using $x := x + N(0, \delta)$. $N(\mu, \sigma)$ denotes a random number having a normal distribution with mean μ and standard deviation σ.
Mutation-GP	An individual is selected from the parent population. A random node of a random tree of this individual is chosen. Internal nodes are chosen with a probability of 90%. External nodes are chosen with a probability of 10%. A new sub-tree is generated and replaces the chosen node.
Extend-Mutation	An individual is selected from the parent population. A random terminal node of a random tree of this individual is chosen. The chosen terminal node is replaced by the elementary function "Addition". A new terminal node is generated. The new terminal node and the node that was previously chosen become the child nodes of the newly generated elementary function.
Prune-Mutation	An individual is selected from the parent population. A random terminal node of a random tree of this individual is chosen. The parent node of the chosen terminal node is replaced by the other sub-tree of the parent node. If the tree only consists of a single terminal node then a new terminal node is generated replacing the old one.
Crossover	Two individuals are selected from the parent population. A random sub-tree is selected within the same random tree of both individuals. The two sub-trees are then exchanged between the two individuals. For each crossover, we only generate a single offspring. The second offspring is discarded.
Tree-Crossover	Two individuals are selected from the parent population. We generate one offspring selecting the trees for the offspring from either the first or the second parent.

5 Experiments

A population size of 1000 individuals was used. It was evolved for 1000 generations. Thus, a total of 10^6 fitness evaluations were performed. All individuals from the first generation consisted of three Gaussians (one for each tree) with random positions along the range from $[390, 700]$ and standard deviations from the range $[1, 100]$. An evolutionary strategy type of mutation was used to optimize the strength, the position as well as the standard deviations of all Gaussians of an individual. We are using a standard evolutionary strategy mutation operation with automatic step size adaptation, i.e. each internal parameter has an

associated standard deviation. The mutation step size was initialized to $\sigma = 0.01$ and the variation of the step size was set to 5%, i.e. $\tau = 0.05$. The remaining genetic operators modify the structure of the individual.

The best individual was always reproduced once into the next generation. The remaining individuals of the population were filled using the following percentages: Mutation-ES (90%), Mutation-GP (2%), Extend-Mutation (2%), Prune-Mutation (2%), Crossover (2%), Tree-Crossover (2%). Tournament selection with a tournament size of 5 was used to select individuals. A human would probably approach the problem by first adapting the position and standard deviation of the single Gaussian for each tree and then refining this solution using additional Gaussians as needed. That's why we applied the evolutionary strategy type of mutation much more frequently than the other operators.

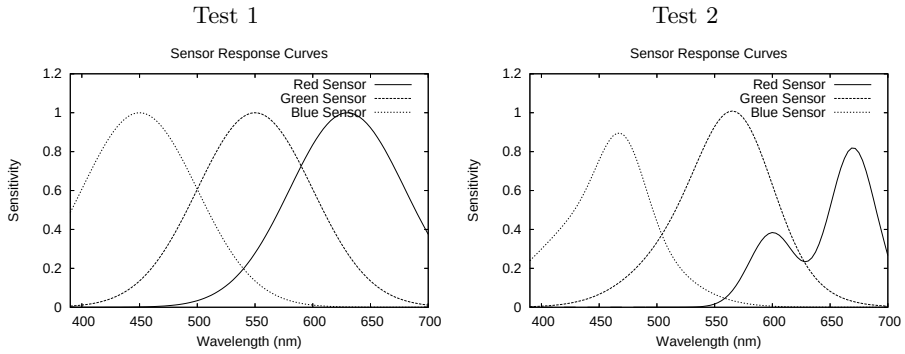

Fig. 4. Three different sensor response functions which are used to evaluate the evolutionary algorithm.

We first evaluated the proposed method on two sample problems where the ground truth data is known. We generated synthetic response functions by overlaying Gaussians. These two synthetic response functions are shown in Figure 4. A virtual calibration target with known reflectances was also created. The synthetic response functions were then used to compute the response of the simulated sensor using Equation 5. The evolutionary algorithm evaluates the fitness of an individual using Equation 7. Since we know the actual response function $\mathbf{S}(\lambda)$, we can evaluate how well the evolved response function $\tilde{\mathbf{S}}(\lambda)$ matches this data. For this evaluation, the evolved response function is normalized such that the maximum response is 1.0. The fit to the actual data is then evaluated by computing

$$E_{\text{actual}} = \frac{1}{96} \sum_{\lambda \in \{390,\ldots,700\}} \sum_{i \in \{r,g,b\}} (S_i(\lambda) - \tilde{S}_i(\lambda))^2. \tag{9}$$

The results obtained for both synthetic response functions are shown in Table 3. A total of 10 runs were performed for each sample problem. The table shows

Table 3. Experimental results obtained during 10 different runs. The standard deviation is shown in round brackets.

Exp	E_{fit}	E_{actual}
Test 1	0.0024(0.0030)	0.0033(0.0092)
Test 2	0.0517(0.0226)	0.0057(0.0024)

the average minimum error measure E_{fit} and also the average deviation between the evolved solution and the actual sensor response function E_{actual}. The standard deviations are also shown. The best of the evolved individuals during all 10 runs is shown in Figure 5. The best individuals approximate the actual sensor response curves quite well. However, a problem of this approach is also apparent. Gaussians with a small standard deviation may be introduced which only have a small impact on the fitness of the individual and hence are only eventually removed. At present, it is not clear whether the approach of Ebner [17] or the approach presented here is better suited to this problem. This will be evaluated in future research.

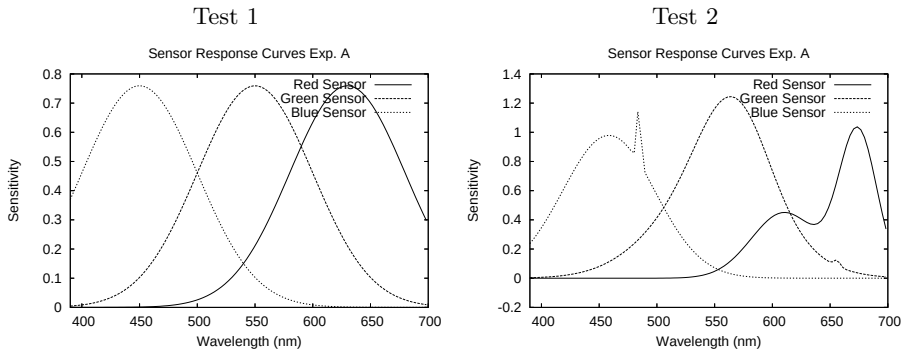

Fig. 5. Best evolved sensor response curves during all 10 runs for the two experiments.

Apart from testing the proposed method on artificial data, we also used it to obtain the sensitivities of two commercially available digital cameras: a Canon 10D and a FujiFilm FinePix F30. The results obtained are shown in Figure 6.

6 Conclusion

Knowing the spectral sensitivity of an optical system is very important for color vision research. The spectral sensitivities are a result of the type of sensor used and are also influenced by the type of lens and filters which are placed in front of the sensor. We have shown how genetic programming may be applied to this type of problem. The method uses a calibration target with known reflectances. The optical system is used to take an image of the calibration target. Evolution

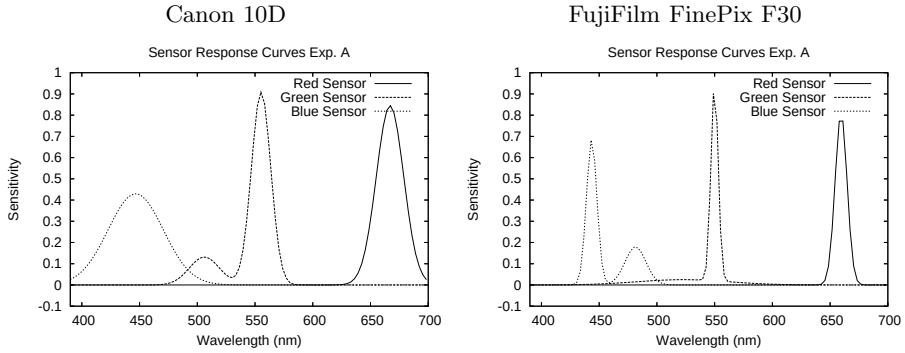

Fig. 6. Best evolved sensor response curves for two commercially available cameras: a Canon 10D with an EF 28-135mm 1:3.5-5.6 IS USM Canon lens and an UV filter and a FujiFilm FinePix F30.

then searches for sensor response curves which reproduce the colors shown in the image of the calibration target. Previously, evolutionary strategies were used to address this problem. Constraints have to be enforced in order to produce a physically plausible sensitivity. This is because the energy measured by a sensor is given by integrating over a range of wavelengths. With our approach the constraints are naturally fulfilled by the type of representation used. We simply represent a sensor response curve as the sum over several Gaussians represented as a tree. The shape of this tree is evolved using genetic programming. Internal parameters which define the position and standard deviations of the Gaussians are evolved using an evolution strategy. We have used two sample problems where the ground truth data is available to evaluate the approach. We then applied this method to obtain the sensor response curves of two commercially available digital cameras.

References

1. Wyszecki, G., Stiles, W.S.: Color Science. Concepts and Methods, Quantitative Data and Formulae. second edn. John Wiley & Sons, Inc., New York (2000)
2. International Commission on Illumination: Colorimetry, 2nd edition, corrected reprint. Technical Report 15.2, International Commission on Illumination (1996)
3. Ebner, M.: Color Constancy. John Wiley & Sons, England (2007)
4. Wandell, B.A.: The synthesis and analysis of color images. IEEE Transactions on Pattern Analysis and Machine Intelligence **PAMI-9**(1) (1987) 2–13
5. Finlayson, G.D., Drew, M.S., Funt, B.V.: Color constancy: generalized diagonal transforms suffice. Journal of the Optical Society of America A **11**(11) (1994) 3011–3019
6. Finlayson, G.D., Hordley, S.D.: Color constancy at a pixel. Journal of the Optical Society of America A **18**(2) (2001) 253–264
7. Finlayson, G.D., Hordley, S.D., Drew, M.S.: Removing shadows from images. In: Proceedings of the European Conference on Computer Vision, Berlin, Springer-Verlag (2002) 823–836

8. Koza, J.R.: Genetic Programming. On the Programming of Computers by Means of Natural Selection. The MIT Press, Cambridge, Massachusetts (1992)
9. Koza, J.R.: Genetic Programming II. Automatic Discovery of Reusable Programs. The MIT Press, Cambridge, Massachusetts (1994)
10. Banzhaf, W., Nordin, P., Keller, R.E., Francone, F.D.: Genetic Programming - An Introduction: On The Automatic Evolution of Computer Programs and Its Applications. Morgan Kaufmann Publishers, San Francisco, California (1998)
11. Bayer, B.E.: Color imaging array. United States Patent No. 3,971,065 (1976)
12. Zhang, Y., Ji, Q.: Camera calibration with gentic algorithms. In: Proceedings of the 2001 IEEE International Conference on Robotics & Automation, Seoul, Korea, May 21-26, IEEE (2001) 2177–2182
13. Rodehorst, V., Hellwich, O.: Genetic algorithm sample consensus (gasac) - a parallel strategy for robust parameter estimation. In: International Workshop '25 Years of RANSAC', New York, USA, IEEE (2006)
14. Cerveri, P., Pedotti, A., Borghese, N.A.: Combined evolution strategies for dynamic calibration of video-based measurement systems. IEEE Transactions on Evolutionary Computation **5**(3) (2001) 271–282
15. Johnson, C.M., Bhat, A., Thibault, W.C.: An evolutionary approach to camera-based projector calibration. In: Proceedings of the Genetic and Evolutionary Computation Conference 2006, Seattle, Washington, July 8-12, ACM (2006) 1871–1872
16. Carvalho, P., Santos, A., Dourado, A., Ribeiro, B.: On the estimation of spectral data: a genetic algorithm approach. In: Proceedings of the IEEE International Conference on Image Processing, Thessaloniki, Greece, October 7-10, IEEE (2001) 866–869
17. Ebner, M.: Estimating the spectral sensitivity of a digital sensor using calibration targets. In: Proceedings of the Genetic and Evolutionary Computation Conference, July 7-11, 2007, London, England, New York, ACM (2007) 642–649
18. Rechenberg, I.: Evolutionsstrategie '94. frommann-holzboog, Stuttgart (1994)
19. Schwefel, H.P.: Evolution and Optimum Seeking. John Wiley & Sons, New York (1995)
20. Buchsbaum, G.: A spatial processor model for object colour perception. Journal of the Franklin Institute **310**(1) (1980) 337–350
21. Finlayson, G.D.: Color in perspective. IEEE Transactions on Pattern Analysis and Machine Intelligence **18**(10) (1996) 1034–1038
22. Forsyth, D.A.: A novel approach to colour constancy. In: Second International Conference on Computer Vision (Tampa, FL, Dec. 5-8), IEEE Press (1988) 9–18
23. Stokes, M., Anderson, M., Chandrasekar, S., Motta, R.: A standard default color space for the internet - sRGB. Technical report, Version 1.10 (1996)