# A KERNEL APPROACH TO GAMUT BOUNDARY COMPUTATION

*Joachim Giesen*[†], *Eva Schuberth*[†‡], *Klaus Simon*[‡] *and Peter Zolliker*[‡]

† Institute for Theoretical Computer Science, ETH Zürich, CH-8092 Zürich
‡ Swiss Federal Laboratories for Materials Testing and Research, CH-8600 Dübendorf

## ABSTRACT

We present a kernel based method to associate an image gamut given as a point cloud in three-dimensional Euclidean space with a continuous shape. The shape we compute is implicitly given as the zero-set of a smooth function that we compute from the point cloud using an efficient optimization method. The feasibility of our approach is demonstrated on a couple of examples.

## 1. INTRODUCTION

Kernel methods lately became very popular in machine learning, see for example the book by Schölkopf and Smola [8]. Kernel methods are best known in the framework of *support vector machines*(SVM) used for binary pattern classification. Roughly speaking the binary pattern classification problem is the following: given a (finite) set of points in $d$-dimensional Euclidean space, where each point has an associated label 1 or $-1$, respectively. The task is to compute a function

$$f : \mathbb{R}^d \to \mathbb{R},$$

such that the set $\{x \in \mathbb{R} \,|\, f(x) \geq 0\}$ fairly well approximates the region that contains the input points with label 1 and the set $\{x \in \mathbb{R} \,|\, f(x) < 0\}$ approximates the region that contains the input points with label $-1$. The kernel approach to the binary pattern classification problem is to express the function $f$ as linear combination of *kernel* functions centered at the input points. The coefficients in the linear combination are computed as the solution of an optimization problem, which is motivated from statistical learning theory. As we have described it here the binary pattern classification problem is basically a *shape* fitting problem.

An image gamut is essentially a (finite) point cloud in $\mathbb{R}^3$. When processing image gamuts one often faces the problem to associate a continuous shape with them. The approach taken here to compute such a shape is exactly as in the SVM approach to the binary pattern classification problem, namely, we compute a function $f : \mathbb{R}^3 \to \mathbb{R}$ as a linear combination of kernel functions centered at the points contained in the image gamut. We derive the coefficients from an optimization problem similar to the one in pattern classification. The only difference to the pattern classification problem is that our input points are not labeled, i.e., all belong to the same class. We are by far not the first to extend the kernel methodology to single class data, see [7]. But we demonstrate that the kernel approach to equip an image gamut with a continuous shape is computationally feasible and gives shapes that in some respects are better suited for further processing, e.g., gamut mapping, than existing methods [3, 1, 5].

## 2. IMAGE GAMUTS

A *gamut* is an entirety of colors. For many purposes it is appropriate to describe a gamut as a subset of a three-dimensional space. We distinguish two kinds of gamuts - device and image gamuts. A device gamut is associated with a device like a printer or monitor and contains all colors that can be physically realized by the device, whereas an image gamut is the set of all colors that are contained in an image. Device gamut descriptions are usually part of ICC-profiles from which a continuous shape, e.g., a polyhedron, that contains all the realizable colors can be inferred.

Here our focus is on image gamuts of digital images. Since there are only finitely many colors contained in a digital image the corresponding image gamut can be represented by a (finite) point cloud in three-dimensional space. We can think of the image gamut as embedded in three-dimensional Euclidean space $\mathbb{R}^3$. But one has to be careful, the Euclidean structure, i.e., essentially distances and angles, is only meaningful for certain embeddings. One such embedding, where the Euclidean structure approximately makes sense, is the device independent color space CIELAB.

Printing a digital image involves mapping the image gamut into the device gamut of the printer. This is a special case of the problem of gamut mapping, which is central to the whole area of color management. Most approaches for image-dependent gamut mapping, i.e., where the input is a point cloud describing an image gamut, first compute a continuous shape that contains the point cloud and base all subsequent computations on this shape. Such a shape is also of independent interest for gamut visualization.

In the following we present a method that computes for a given image gamut $X = \{x_1, ..x_n\}$ a continuous shape that contains $X$. Our method works in all three dimensional color spaces, but since it relies on distance and angle computations it is arguably best suited for color spaces that approximately respect the Euclidean structure like for example the CIELAB color space.

## 3. SHAPES FROM KERNELS

The choice of the continuous shape to be associated with an image gamut depends on the intended application. But there are two frequently encountered objectives, see also [3]. First, the shape should capture the "geometry" of the gamut as well as possible, and second, the shape should be efficiently computable. Another objective may be that the boundary of the shape is a smooth (closed) surface. The latter is helpful for the gamut mapping framework that we designed in [4]. Here we describe an approach to equip an image gamut with a shape that has a smooth boundary based on *positive kernels*, or more explicitly on *radial basis functions*.

**Kernel and feature map.** A *kernel* on $\mathbb{R}^3$ is a function

$$K : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}.$$

A *feature map* on $\mathbb{R}^3$ is a function

$$\phi : \mathbb{R}^3 \to \mathscr{H},$$

where $\mathscr{H}$ is a Hilbert space. Using the inner product on $\mathscr{H}$ we get a kernel $K$ from a feature map $\phi$ by setting

$$K(x,y) := \langle \phi(x), \phi(y) \rangle$$

for all $x,y \in \mathbb{R}^3$. The latter kernel is an example for a symmetric kernel, i.e., it always holds $K(x,y) = K(y,x)$. Furthermore the kernel is positive semi-definite, i.e., for any finite subset $X = \{x_1,\ldots,x_n\} \subset \mathbb{R}^3$ the matrix $(K(x_i,x_j))$ is positive semi-definite.

In most cases one does not start with a feature map, but with a symmetric, positive semi-definite kernel. However, the notions coincide, namely, any symmetric, positive semi-definite kernel can be generated by a suitable feature map, see [6]. That is, given a symmetric, positive semi-definite kernel $K$, we know that there exists a corresponding feature map.

A special class of symmetric, positive semi-definite kernels are derived from *radial basis functions*, i.e., functions of the form

$$k : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}.$$

The kernel associated with a radial basis function $k$ is then $K(x,y) := k(\|x-y\|)$. In the following we refer to such kernels as *radial kernels*.

**Kernel expansion.** Given an image gamut $X = \{x_1,\ldots,x_n\} \subset \mathbb{R}^3$ and a radial kernel $K$ on $\mathbb{R}^3$. A *kernel expansion* is a function

$$f : \mathbb{R}^3 \to \mathbb{R}, \quad x \mapsto \sum_{i=1}^{n} \alpha_i K(x,x_i),$$

where the $\alpha_i \in \mathbb{R}$ are the coefficients of the expansion. We can use a kernel expansion to associate a shape $S$ with the gamut $X$ as follows,

$$S = \{x \in \mathbb{R}^3 \,|\, f(x) \leq t\},$$

where $t \in \mathbb{R}$ is some threshold.

**Shape optimization.** Again, let $X$ be an image gamut, $K$ a radial kernel and $\phi$ an associated feature map into feature space $\mathscr{H}$. We want to associate $X$ with a shape $S$ derived from a kernel expansion. In order for $S$ to be meaningful the kernel expansion has to be meaningful. Here we take an optimization approach in feature space $\mathscr{H}$ as it is usually done for kernel methods in machine learning, see for example [8]. If we find a hyperplane in $\mathscr{H}$ that contains all points of $\phi(X)$ on one side, then also the pre-image of this hyperplane, which besides in degenerate situations is a closed surface $\Sigma$ in $\mathbb{R}^3$, has this property. If we can ensure that the points in $X$ lie on the bounded side of $\Sigma$, then the shape $S$, i.e., the compact volume bounded by $\Sigma$, is at least meaningful in the sense that it contains the points of $X$. A hyperplane in $\mathscr{H}$ is given by a normal vector $w$ and an offset $\rho \in \mathbb{R}$. The constraints for the hyperplane $\{y \,|\, \langle w,y \rangle - \rho = 0\}$ read as,

$$\langle w, \phi(x_i) \rangle \geq \rho, \quad \text{for all } x_i \in X.$$

The pre-image of the hyperplane is given as

$$\{x \in \mathbb{R}^3 \,|\, \langle w, \phi(x) - \rho = 0\}.$$

Among all hyperplanes that fulfill the constraints we choose the one that has the largest distance to the origin in $\mathscr{H}$. The latter hyperplane is called a *maximum margin hyperplane*. Its choice is not justified by insights into the geometry of the problem, but we will see later that it gives reasonable results and can be modified to give even better results. One can show that a solution to the following optimization problem is a maximum margin hyperplane, see [8].

$$\begin{array}{ll} \min_{w,\rho} & \frac{1}{2}\|w\|^2 - \rho \\ \text{subject to} & \langle w, \phi(x_i) \rangle \geq \rho, \quad \text{for all } x_i \in X. \end{array}$$

This optimization is better solved in its Lagrangian dual form, which is a convex, quadratic program and only depends on kernel evaluations.

$$\begin{array}{ll} \min_{\alpha_i} & \sum_{i,j=1}^{n} \alpha_i \alpha_j K(x_i,x_j) \\ \text{subject to} & 0 \leq \alpha_i, \quad \text{for } i = 1,\ldots,n \\ & \sum_{i=1}^{n} \alpha_i = 1, \end{array}$$

where the $\alpha_i$ are the dual variables of the $x_i \in X$. The $\alpha_i$ are exactly the coefficients in the kernel expansion we are looking for, i.e., they completely characterize the maximum margin hyperplane in $\mathscr{H}$, namely

$$w = \sum_{i=1}^{n} \alpha_i \phi(x_i) \quad \text{and} \quad \rho = \|w\|^2 = \sum_{i,j=1}^{n} \alpha_i \alpha_j K(x_i,x_j).$$

The kernel expansion now reads as

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x,x_i)$$

and natural choice for the shape $S$ is

$$S = \{x \in \mathbb{R}^3 \,|\, f(x) \leq \rho\}.$$

The shape $S$ is essentially given by the points $x_i$ in $X$ for which the constraint $\langle w, \phi(x_i) \rangle \geq \rho$ holds with equality, i.e., for which

$$\langle w, \phi(x_i) \rangle = \rho.$$

We call the latter points *support vectors*. Note that the support vectors are contained in the surface $\Sigma$, i.e., in the boundary of the shape $S$. If we have only very few support vectors that might be an indication that the shape $S$ does not fit the gamut $X$ as tightly as possible. In that case we are looking for a way to increase the number of support vectors or points in $X$ close to the surface to be computed. Schölkopf et al. came up with a method to do so, see [7]. The idea is to introduce another set of constraints that fits a pair of parallel hyperplanes aka a *slab* to the data in feature space. The new constraints read as follows,

$$\delta \leq \langle w, \phi(x_i) \rangle - \rho \leq \delta^*. \quad \text{for all } x_i \in X.$$

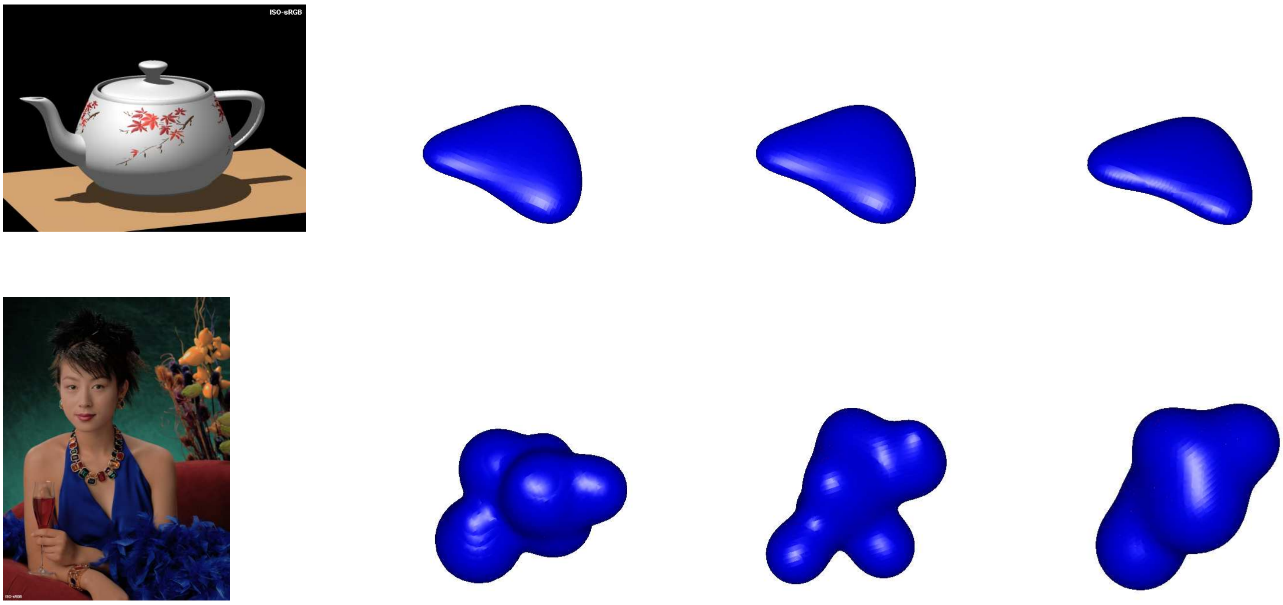

Figure 1: First row: the test image TEAPOT and the computed shape for a sampling rate of 0.2, 0.1 and 0.01. Second row: the test image WOMAN and the computed shape for a sampling rate of 0.2, 0.1 and 0.01

Maximizing the distance of the slab to the origin results in the following optimization problem,

$$\begin{array}{ll} \min_{w,\rho} & \frac{1}{2}\|w\|^2 - \rho \\ \text{subject to} & \delta \le \langle w, \phi(x_i)\rangle - \rho \le \delta^* \quad \text{for all } x_i \in X, \end{array}$$

where $\delta$ and $\delta^*$ are parameters that determine the width of the slab. The Lagrangian dual of this problem again is a convex quadratic program that reads as follows,

$$\begin{array}{ll} \min_{\alpha_i^{(*)}} & \sum_{i,j=1}^n (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(x_i,x_j) \\ & -\sum_{i=1}^n (\delta\alpha_i + \delta^*\alpha_i^*) \\ \text{subject to} & 0 \le \alpha_i^{(*)}, \quad \text{for } i = 1,\ldots,n \\ & \sum_{i=1}^n (\alpha_i - \alpha_i^*) = 1. \end{array}$$

Again, the $\alpha_i$ and $\alpha_i^*$ are exactly the coefficients in the kernel expansion we are looking for. They completely characterize the maximum margin slab in $\mathscr{H}$, namely

$$w = \sum_{i=1}^n (\alpha_i - \alpha_i^*)\phi(x_i) \quad \text{and} \quad \rho = \|w\|^2.$$

The kernel expansion now reads as

$$f(x) = \sum_{i=1}^n (\alpha_i - \alpha_i^*)K(x,x_i)$$

and natural choice for the shape $S$ is

$$S = \{x \in \mathbb{R}^3 \mid f(x) \le \rho + (\delta^* - \delta)/2)\}.$$

Note that we can recover the maximum margin hyperplane problem (and its solution) by setting $\delta = 0$ and $\delta^* = \infty$.

## 4. IMPLEMENTATION

We used a modified version of the LIBSVM, see [2] to solve the optimization problems. To ensure feasibility of the problems we allowed also outliers that we penalized in the objective function, i.e., the constraints for the problem with outliers are

$$\delta - \xi_i \le \langle w, \phi(x_i)\rangle - \rho \le \delta^* + \xi_i^* \quad \text{for all } x_i \in X.$$

and the objective function is

$$\min_{w,\rho} \quad \frac{1}{2}\|w\|^2 - \rho + c\sum_{i=1}^n (\xi_i + \xi_i^*),$$

where $c \in (0,1)$ is a parameter that controls the trade-off between penalizing the outliers and maximizing the distance of the slab to the origin. The dual of the latter problem reads as

$$\begin{array}{ll} \min_{\alpha_i^{(*)}} & \sum_{i,j=1}^n (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(x_i,x_j) \\ & -\sum_{i=1}^n (\delta\alpha_i + \delta^*\alpha_i^*) \\ \text{subject to} & 0 \le \alpha_i^{(*)} \le c, \quad \text{for } i = 1,\ldots,n \\ & \sum_{i=1}^n (\alpha_i - \alpha_i^*) = 1. \end{array}$$

## 5. RESULTS

For our experiments we used *Gauss kernels* that are given as

$$K(x,y) = \exp(-\|x-y\|^2/2\sigma),$$

where $\sigma > 0$ is a parameter. We believe that other kernels like *inverse multiquadrics*, see for example [6], should give similar results.

For our tests we used a suite of test images, both natural and artificial. Here we will present results for four test images: RGBCOLORS, WOMAN, BOUQUET and TEAPOT. The latter three are test images from the International Organization for Standardization (ISO) and are available at `http://www.iso.org`. RGBCOLORS is a generated image that contains the colors of a quantized RGB-cube. Each axis of the cube is quantized to 64 values.

We mentioned earlier that there is a time quality tradeoff when it comes to the computed shapes, i.e., computing a

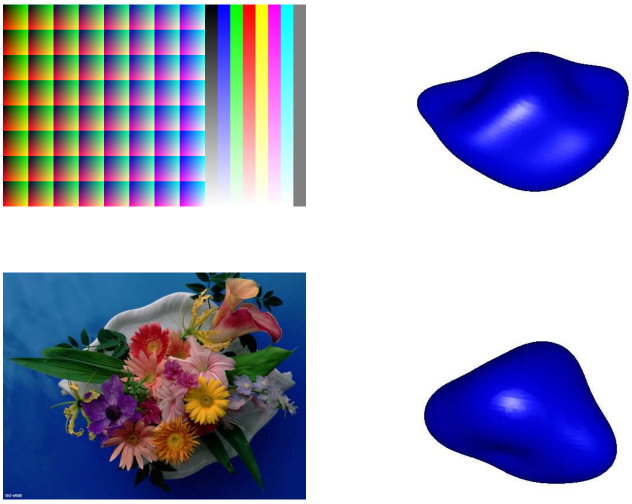

Figure 2: The test images RGBColors and Bouquet (on the left) and the computed shapes for a sampling rate of 0.01 (on the right).

"geometrically more accurate" shape takes longer than just computing a rough approximation. We dealt with this trade-off by randomly sub-sampling a gamut $X$ and computing a shape only for the sub-sample, which is faster but less accurate. Figure 1 shows the results for the test images Teapot and Woman for different sampling rates. Figure 2 shows the result for the test images RGBcolors and Bouquet for a sampling rate of 0.01. Note that in order to visualize the computed shapes we triangulated their boundaries using the marching cubes method.

In Table 1 we provide additional information on our test cases. The column *Size* contains the number of pixels of the image, the column *Sampling* contains the sampling rate (we used a random sampling strategy for our tests), the column *SV* contains the number of points with a positive or negative coefficient (these are the points on the boundary of the computed shape) and the column *Time* contains the running time measured in seconds (the timing was done on a PC with a 798MHz Pentium III processor).

| Image | Size | Sampling | SV | Time |
|---|---|---|---|---|
| RGBcolors | 393216 | 0.01 | 3932 | 164s |
| Woman | 196608 | 0.01 | 439 | 7s |
| | | 0.1 | 1144 | 24s |
| | | 0.2 | 2549 | 2046s |
| Bouquet | 196608 | 0.01 | 1934 | 33s |
| Teapot | 172800 | 0.01 | 1714 | 14s |
| | | 0.1 | 17377 | 198s |
| | | 0.2 | 34206 | 541s |

Table 1: Additional data for the test images

**Discussion.** The computed shapes are kind of roundish but nevertheless seem to be an appropriate representation for image gamuts. The smoothness of their boundaries make them especially appropriate for image dependent gamut mapping algorithms like the one described in [4]. However, we should point out that in order to make our method practical, we had to apply random sampling to reduce the number of color points. In our experiments it turned out that most of the computed shapes are stable under sub-sampling, i.e., the shape takes a similar form for a quite small sub-sample as it takes for the whole point cloud. In Figure 1 this is shown for the image Teapot: decreasing the sampling rate by a factor of 20 from 0.2 to 0.01 hardly influence the shape. However there are cases like for the example Woman, where a change in the sampling rate also changes the resulting shape. The running times increase with increasing sampling rate and range from some seconds to half an hour. For an application of the method in an image dependent gamut mapping algorithm this should and also could be improved by for example using a more sophisticated implementation and by decreasing the sampling rate even further. In [3] we presented another method for computing a shape capturing an image gamut, the *discrete flow complex*. The *discrete flow complex* is fast computable but leads to ragged shapes especially when an image only contains few colors. It seems that the shapes obtained by the kernel method are more appropriate to describe image gamuts however one has to deal with longer computation times. Finally, the kernel method that we presented here involves quite some free parameters, e.g., $\sigma$ the width of the Gaussian kernel or the width of the slab. So far we have not thoroughly examined how setting these parameters affect the computed shapes and the time to compute them. Hopefully there is a set of parameters that works well for almost all image gamuts. Since we did not change the parameter settings much in order to compute the results for different images and sampling rates it seems likely that such a set of parameters exists.

## REFERENCES

[1] T. J. Cholewo and S. Love. Gamut boundary determination using alpha-shapes. In *7th Color Imaging Conference*, volume 7, pages 200–204, Scottsdale, AR, 1999. IS&T/SID.

[2] Rong-En Fan, Pai-Hsuen Chen, and Chih-Jen Lin. Working set selection using second order information for training svm. *Journal of Machine Learning Research*, 6:1889–1918, 2005.

[3] J. Giesen, E. Schuberth, K. Simon, and P. Zolliker. Toward image-dependent gamut mapping: fast and accurate gamut boundary determination. In Reiner Eschbach and Gabriel Marcu, editors, *Color Imaging X: Processing, Hardcopy and Applications*, volume 5667, pages 201–210, San Jose, CA, 2005. SPIE/IS&T.

[4] J. Giesen, E. Schuberth, K. Simon, and P. Zolliker. A framework for image-dependent gamut mapping. In *Color Imaging XI: Processing, Hardcopy and Applications*, San Jose, CA, 2006. SPIE/IS&T.

[5] J. Morovic and M. R. Luo. Calculating medium and image gamut boundaries for gamut mapping. *Color Research and Applications*, 25(6):394–401, 2000.

[6] R. Schaback and H. Wendland. Kernel techniques: From machine learning to meshless methods. *Acta Numerica*, 2006.

[7] Bernhard Schölkopf, Joachim Giesen, and Simon Spalinger. Kernel methods for implicit surface modeling. In *Proceedings of the Annual Conference on Neural Information Processing Systems*, 2004.

[8] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels Support Vector Machines, Regularization, Optimization and Beyond.* MIT Press, Cambridge, 2002.