Histogram compression and image retrieval through Padua points interpolation

Roberto Montagna and Graham Finlayson; University of East Anglia; Norwich, UK

Abstract

Colour has proved to be a very powerful feature for image indexing. There are many examples of image retrieval systems based on colour or chromaticity histograms in the literature, following on from the histogram intersection method of Swain and Ballard. Here we propose a compact representation of the chromaticity histogram that achieves very good performance in image retrieval. Specifically, we use a new type of polynomial interpolation in two variables, which relies on the Padua points *as interpolation nodes. What we obtain is a vector of coefficients, that represents the interpolation polynomial and is "characteristic" for an image, and that can be compared to the corresponding vectors of other images. Experiments show that our new compact Padua point representation supports excellent indexing and recognition.*

Introduction

The increased storage capacity together with the large diffusion of digital photography makes it possible to produce and store huge amounts of pictures. This is true not only for very large on-line image archives, but also for end users who own a modern personal computer: a consumer digital camera can easily store hundreds of images in its internal memory, therefore users tend not to pay attention to the number of pictures taken. The result is the production of several gigabytes of personal photos in a few months, where it's hard to locate individual pictures unless an effort has been made from the very beginning to properly index them. When it comes to on-line services where the subscribers can upload their own pictures to share them with other people, the same issue grows exponentially.

A very common strategy adopted to ease the search within such archives is tagging the pictures with a text label or description, but this requires time-consuming work to be performed by a human operator. In the case of on-line services, this task is usually left to the person who publishes the pictures: a drawback of this is that users might use inappropriate labels which might cause false search results. For these reasons it is desirable to produce an automated system that, given an example image, returns a collection of similar images.

Several cues can be used to index images, and colour has proven to be a very powerful feature for this task, especially in the form of colour and chromaticity histograms. Although in complex scenes these features might not, on their own, provide high accuracy, very fast algorithms can use them in order to narrow down the search for other more accurate, yet slower, algorithms that can also utilize other cues.

In this paper we propose a compact representation of the chromaticity histogram of an image, based on a new type of polynomial interpolation called *Padua points interpolation*. The approximation we obtain is univocally described by a vector of coefficients, that is "characteristic" for an image and therefore can be used for the indexing task. In this paper we first we introduce the Padua points and we describe their main features. Then, we propose our algorithm for indexing and retrieval of images; finally, we show our results in comparison with other histogram-based methods.

The Padua points

The problem of polynomial interpolation in two variables has been approached in a number of different ways. In an interpolation scheme, one would like to find some features that facilitate good performance. First of all, the approximation error should be low; it is bounded by a measure called the *Lebesgue constant*, which grows with the interpolation degree. It is obviously desirable that this growth be slow and it has been proven to be at least of $O(\log n)$ in the case of interpolation in one variable, and of $O(\log^2 n)$ in two variables, where n is the interpolation degree [4]. Last but not least, the computational efficiency of the interpolation scheme should be as high as possible.

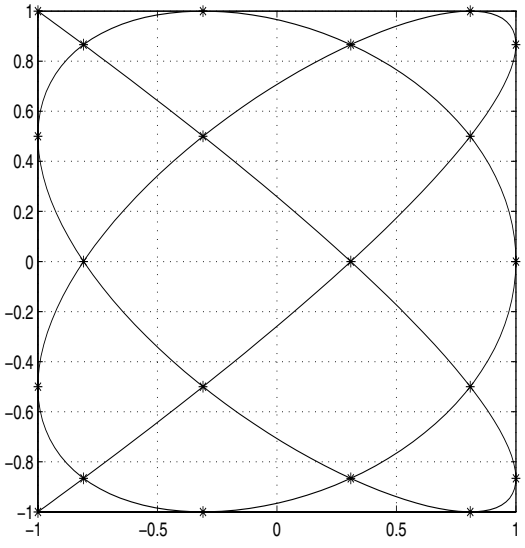

Figure 1. *Padua points of the first family of interpolation degree $n = 5$ and their generating curve as in equation* (2).

Although there are many examples of nodes that guarantee unisolvence and a minimal growth of the Lebesgue constant for interpolation in one variable, for interpolation in two variables only recently has a set of points been introduced that meets both these criteria. These points are called *Padua points* (see [5]), after the Italian university where they were first discovered. Up to now, they are the only *known* points that guarantee *unisolvence* (i.e. their interpolating polynomial is unique) and have a proven minimal growth of the *Lebesgue constant* of $O(\log^2 n)$ for interpolation in two variables (cf. [2, 3]).

The Padua points are defined in the square $[-1,1] \times [-1,1] \subset \mathbb{R}^2$ (but they can be easily mapped to any rectangular

domain) as the union of two grids of Chebyshev points. Given the interpolation degree n, we have $N = \frac{1}{2}(n+1)(n+2)$ points defined as

$$\begin{aligned} \mathrm{Pad}_n &= \{\boldsymbol{\xi} = (\xi_1, \xi_2)\} \\ &= \left\{ \gamma\left(\frac{k\pi}{n(n+1)}\right),\ k = 0, \ldots, n(n+1) \right\}, \end{aligned} \tag{1}$$

where $\gamma(t)$ is their *generating curve* [2]. There are four families of Padua points, obtained with subsequent 90 degree rotations of the following curve:

$$\gamma(t) = (-\cos((n+1)t), -\cos(nt)) \quad t \in [0, \pi]. \tag{2}$$

Two of the points lie on consecutive vertices of the square, $2n-1$ points lie on the edges of the square, and the interior points lie on the self-intersections of the generating curve (see figure 1).

There are two features of the Padua points which are especially suitable for compression purposes. First of all, once we know the degree n of the interpolation we have a formula to determine the points. This means that there is no need to store their coordinates. Second, the interpolation formula for the Padua points can be written in the bivariate Chebyshev polynomials orthonormal basis. This means that we can univocally identify an interpolating polynomial from its coefficients. Together with *unisolvence*, this feature can be seen as a valuable tool for the identification of a given function.

Provided that $\boldsymbol{\xi} = (\xi_1, \xi_2)$ are the Padua points for interpolation degree n, we can write the coefficients $c_{j,k-j}$ as follows:

$$\begin{aligned} b_{j,k-j} &= \sum_{\boldsymbol{\xi} \in \mathrm{Pad}_n} f(\boldsymbol{\xi}) w_{\boldsymbol{\xi}} \hat{T}_j(\xi_1) \hat{T}_{k-j}(\xi_2), \quad 0 \le j \le k \le n \\ c_{j,k-j} &= b_{j,k-j}, \quad c_{n,0} = \frac{1}{2} b_{n,0}, \end{aligned} \tag{3}$$

where $f(\boldsymbol{\xi})$ is the value at each Padua point of the function we are going to approximate; $\hat{T}_i(\cdot)$ are the normalised Chebyshev polynomials of the first kind of degree i, that is

$$\begin{aligned} \hat{T}_0(x) &= T_0(x) = 1 \\ \hat{T}_i(x) &= \sqrt{2} T_i(x) = \sqrt{2} \cos(i \arccos(x)); \end{aligned} \tag{4}$$

and $w_{\boldsymbol{\xi}}$ are the weights corresponding to each Padua point, defined as

$$w_{\boldsymbol{\xi}} = \frac{1}{n(n+1)} \cdot \begin{cases} \frac{1}{2} & \text{if } \boldsymbol{\xi} \text{ is a vertex point} \\ 1 & \text{if } \boldsymbol{\xi} \text{ is an edge point} \\ 2 & \text{if } \boldsymbol{\xi} \text{ is an interior point.} \end{cases} \tag{5}$$

In practice, once we have determined the interpolation degree and then the Padua points, the only thing we need to know to perform the interpolation is the value $f(\boldsymbol{\xi})$.

Finally, we would like to remark what the minimal growth of the Lebesgue constant implies. As above, it provides a theoretical boundary to the growth of the approximation error. This means that, given a function $f\colon [-1,1] \times [-1,1] \subset \mathbb{R}^2 \to \mathbb{R}$, the approximation error will be bounded by

$$E_n \approx O(n^{-p} \log^2 n), \tag{6}$$

where p is the continuity of the function (that is, $f \in C^p([-1,1]^2)$), as proven in the multivariate extension of Jackson's theorem [1].

Histogram approximation and comparison

As written above, the Padua points are defined in a square domain in $\mathbb{R}^2$ and it is matter of ongoing research as to whether it is possible to generalize them to $\mathbb{R}^3$ and $\mathbb{R}^n$. For this reason we can only represent two-dimensional histograms. In our experiments we approximated chromaticity histograms from the RGB colour space, with axes (r,g) defined as

$$r = \frac{R}{R+G+B}, \quad g = \frac{G}{R+G+B} \tag{7}$$

and hue-saturation histograms, with axes (H,S), from the well-known HSV colour space. This latter choice was due to the consideration that, similarly to the chromaticity, we are trying to take into account only the chromatic information, and to ignore the brightness information. What we get at the end is a smooth function, represented by the interpolating polynomial of the Padua points, that represents the density of the pixels projected onto the (r,g) or onto the (H,S) plane.

One of the first issues we had to address is the fact that the pixels of an image, projected onto the (r,g) chromaticity plane, always lie within a triangle of vertices $(0,0)$, $(0,1)$ and $(1,0)$. The Padua points, on the other hand, are defined in a square. We could choose to map them to a triangular domain (or equivalently map the triangular chromaticity domain to a square), but in this way the Padua points would lose their optimal features. This is also one of the reasons why we took into account also (H,S) histograms (since they are already defined on a square). However, our tests showed that without applying any square-to-triangle map, the Padua points correctly represent a density function which has values close to zero outside the triangular domain.

The main problem in approximating a histogram with a polynomial is described concisely in equation (6): the smoother the original function, the lower the approximation error. Our original function here is a histogram, that is intrinsically not smooth, since it is a discrete representation. How can we address this issue? Our approach is not to build a smooth density function that approximates a histogram, but rather to build the density function directly. Since all that we need to know to obtain the interpolating polynomial are the values of some function at the Padua points, we weight each pixel with a Gaussian distribution. More precisely, we perform the following steps:

1. Project each pixel of the image on to the domain (r,g) or (H,V).
2. Compute the coordinates of the Padua points in the same domain (which is $[0,1] \times [0,1] \subset \mathbb{R}^2$).
3. Build a bivariate Gaussian distribution centred on a Padua point $\boldsymbol{\xi} = (\xi_1, \xi_2)$ (i.e. with mean equivalent to the coordinates of $\boldsymbol{\xi}$).
4. Evaluate the Gaussian function at the coordinates of each projected pixel, sum all these values and assign the result to $f(\boldsymbol{\xi})$ (computing this sum of Gaussian-weighted values is relatively slow, although it is possible to speed up considerably this part using a *fast Gauss transform* algorithm [8]).
5. Repeat the procedure from step 3 for each Padua point.

In this way we get the values of $f(\boldsymbol{\xi})$ for each Padua point, which is sufficient for us to compute the coefficients $c_{j,k-j}$ of equation (3), and store them into a characteristic vector $\boldsymbol{C}$. This vector can be regarded as a sort of *signature* of an image, or rather of its chromaticity distribution. Indeed, as we said before, the Padua points are unisolvent, meaning that from a given set of values $f(\boldsymbol{\xi})$ we can get *only one* interpolating polynomial.

Here we outline the steps to perform if we want to match an unknown image in an archive of M model images.

1. Compute the characteristic vectors of the model images $\boldsymbol{C}_1,\ldots,\boldsymbol{C}_M$ (in a real scenario we may assume that they are already stored with the images).
2. Compute the characteristic vector of the unknown image $\boldsymbol{C}_u$.
3. Sort the model images according to their distance $d(u,i) = \|\boldsymbol{C}_u - \boldsymbol{C}_i\|_2$ from the unknown one.

The image that is the "closest" to the unknown one in terms of the 2-norm distance will be the best match.

It is possible to perform a further small step of approximation; when we are using the r,g histogram, that lies in the triangle with vertices $(0,0)$, $(0,1)$ and $(1,0)$, it is possible to set to zero the values of the Padua points outside this triangle. The approximation will be slightly less precise, but we gain speed and the retrieval performance does not change noticeably.

Implementation issues

The entire part of our method that calculates the Padua points interpolation is based on the API provided in the package XuPad2D[1]. With these functions, it is straightforward to generate the Padua points, and then to obtain the coefficients of the interpolating polynomial. Once we get those, it is also very easy to evaluate the polynomial on any set of points. Thus, generating the Padua points is a very quick task. The bottleneck in our method is actually evaluating the Gaussian functions (one for each Padua point) at each pixel projected to the r,g space. Naively, if we have N Padua points and an image of M pixels, we need to perform $O(MN)$ operations to compute this sum. This task is known as the *Gauss transform*, and luckily some quicker algorithms exist, that require only $O(M+N)$ operations, although they only compute an approximation to the solution (see the *fast Gauss transform*, [8]). Only after this weighted sum has been calculated it is possible to obtain the coefficients of the interpolating polynomial, which are then used for the matching.

Figure 2. *The 100 objects photographed in the COIL-100 dataset.*

[1]Available at `http://www.math.unipd.it/~mcaliari/software/XuPad2D.tar.gz`.

The experiments

In order to test the efficiency of our retrieval system, we performed an experiment that consisted of creating an archive of model images (M_i), and then querying it with unknown images (U_i). The retrieval task consisted of sorting the model images according to the distance from the unknown image U_i. Prior to calculating their approximated histogram, the images were segmented by removing the black background with a simple thresholding. To evaluate the performance of the method we measured how many times the correct model ranked first, second and third, how many times it ranked greater than third, and how many times it ranked in the top ten matches. Finally, we calculated the average match percentile (AMP) where, given n models, the match percentile of a model M_i that ranks r, with $1 \le r \le n$ is given by

$$\mathrm{MP} = \frac{n-r}{n-1}. \tag{8}$$

Converting it to a percentage, a value of MP $= 100\%$ means perfect matching; a value of MP $= 99\%$ means that the correct model scored a higher rank than 99% of the other models, and so on. After that, the match percentile is averaged over the set of unknown images.

The first benchmark was the *Columbia Object Image Library* (COIL-100, [9]), which is a widely used dataset in pattern recognition. It features 7200 pictures of a hundred objects (see figure 2), that are each photographed from 72 different angles. Although their shape and size can be different from different view angles, their colour distribution does not change substantially; moreover, histograms are not sensitive to scaling, and we expect the Padua points representation to share this property. For this dataset, we chose as a model the front view of each object, and as query images all the other view angles.

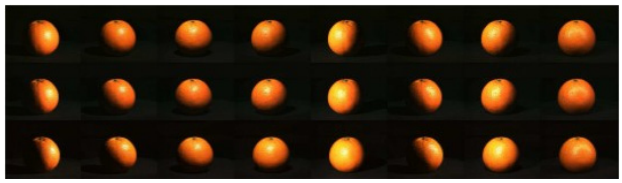

Figure 3. *An example of the viewing conditions of an object belonging to the ALOI dataset.*

Since our method projects the image onto the chromaticity plane, it should not theoretically be affected by differences in light intensity, therefore we also use a dataset where it is possible to test this capability. The *Amsterdam Library of Object Images* (ALOI) has exactly this feature. The library displays a thousand objects, each in several configurations: in particular, we were interested in the images captured under different illumination directions. In [7] the set where the pictures were taken is described. The object was surrounded by five lights, and the camera could be set-up in one of three different geometries: straight in front of the object, turned at 15 to the object and turned at 30 degrees to the object. In addition the object could also be turned with respect to the camera, so that the camera movement could be compensated. In some cases the object was photographed with only one of the five lights turned on, but two pictures were also taken where two lights (two on the right side and two on the left side) were simultaneously turned on, and finally one with all the lights turned on. For each of these light conditions, a picture was taken from all three positions of the camera. This resulted in 24 different illumination conditions for each object, as in figure 3. From the 24 versions of each object, we selected the one with the

most uniform illumination as the model: we had therefore a set of a thousand model images, and all the remaining 23 thousand images were treated as unknown.

Results

Comparison of the results with the histogram intersection and the Padua points retrieval system in the (r,g) space on the COIL-100 dataset.

	Histogram intersection		Padua points	
Coefficients stored	2500	144	496	66
Ranked 1st (%)	78.14	68.94	68.42	67.11
Ranked 2nd (%)	7.28	7.10	8.96	9.86
Ranked 3rd (%)	3.18	4.76	4.65	4.90
Ranked greater than 3rd (%)	11.39	19.20	17.97	18.13
1st and 2nd match (%)	85.42	76.04	77.38	76.97
Top ten (%)	98.84	90.96	91.31	91.85
Average match percentile (%)	98.18	97.45	97.31	97.34

Table 1 summarizes the test results on the COIL-100 dataset. We compared the Padua point representation of the chromaticity distribution with a full chromaticity histogram intersection [10]. The results are interesting, considering the difference between the quantity of data stored by the two methods. As we can see, using the Padua points we have a relatively significant reduction of the overall retrieval performance, but at the same time instead of storing 2500 coefficients we need only 66 of them, with a compression ratio of nearly 38:1 (with interpolation degree $n = 10$ we get a polynomial with 66 coefficients).

Comparison of the results with the histogram intersection and the Padua points retrieval system in the (r,g) space on the ALOI dataset.

	Histogram intersection		Padua points	
Coefficients stored	2500	144	496	66
Ranked 1st (%)	34.76	28.58	37.79	32.21
Ranked 2nd (%)	7.62	8.72	7.96	7.54
Ranked 3rd (%)	4.33	5.33	4.22	4.38
Ranked greater than 3rd (%)	53.28	57.37	50.02	55.97
1st and 2nd match (%)	42.38	37.30	45.75	39.66
Top ten (%)	59.22	59.23	62.91	56.56
Average match percentile (%)	95.18	95.22	95.06	94.07

In table 2, we are again comparing the Padua points and the histogram in the chromaticity space, but this time on the ALOI dataset. In this much larger and harder dataset, when keeping the compression ratio of 38:1 by using 66 coefficients, the performance of the Padua points is only slightly inferior to that of the histogram intersection. Moreover, when increasing the number of stored coefficients to 496, that is about one fifth of the coefficients needed by the histogram intersection, the Padua points achieve the best overall performance.

Comparison of the results with the histogram intersection and the Padua points retrieval system in the (H,S) space on the COIL-100 dataset.

	Histogram intersection		Padua points	
Coefficients stored	2500	144	496	66
Ranked 1st (%)	82.23	75.37	65.41	65.35
Ranked 2nd (%)	6.20	7.69	6.89	6.66
Ranked 3rd (%)	2.53	3.76	3.83	3.83
Ranked greater than 3rd (%)	9.04	12.92	23.87	24.15
1st and 2nd match (%)	88.42	83.32	72.30	72.01
Top ten (%)	96.41	95.75	88.24	88.11
Average match percentile (%)	98.49	98.27	94.62	94.63

Comparison of the results with the histogram intersection and the Padua points retrieval system in the (H,S) space on the ALOI dataset.

	Histogram intersection		Padua points	
Coefficients stored	2500	144	496	66
Ranked 1st (%)	35.79	24.85	27.42	29.10
Ranked 2nd (%)	8.34	7.17	7.24	7.61
Ranked 3rd (%)	4.67	4.49	4.51	4.27
Ranked greater than 3rd (%)	51.20	63.50	60.83	59.01
1st and 2nd match (%)	44.13	32.01	34.67	36.71
Top ten (%)	61.67	51.12	53.29	54.54
Average match percentile (%)	95.83	94.28	94.22	94.03

If we try to compress the (H,S) distribution both on the COIL-100 and on the ALOI dataset, as shown in tables 3 and 4, the performance of the Padua points decreases, especially in comparison to that of the histogram intersection, that instead improves. This is probably due to a different spreading of the distribution into the $[0,1] \times [0,1]$ domain.

Conclusion

Here we have proposed a new representation for two dimensional colour histograms that we have shown to be a good compromise between stored data and retrieval performance. Although it might be argued that the results of the retrieval tests are not excellent, it is known that in such large and complex datasets the colour cue alone cannot guarantee a very high retrieval ratio; however, we want to stress here how our method can efficiently compress the information carried by a larger histogram, without substantially losing much of the retrieval performance, especially when the Padua points are used to build a representation of the chromaticity distribution, as the experiments on the ALOI dataset show. It has to be pointed out that in some cases the histogram intersection has significantly better performance than the Padua points method. This appears to be especially true in the COIL-100 dataset, probably because the rotation of the objects introduces some variation in the colour distribution to which the

Padua points are more responsive than the histogram intersection.

There are still some open questions about the Padua points method, that further investigation can address. Specifically, the most important point is that the standard deviation of the Gaussian functions affects the final retrieval performance. So far we used a standard deviation that gives good results, but understanding how to find an "optimal" standard deviation might significantly improve the retrieval performance. A second interesting issue is to find a norm different from the Euclidean one, that better represents the distance between the coefficients of the polynomials.

Acknowledgements

The authors would like to thank David Connah (University of East Anglia, Norwich, UK) for his helpful feedback on this paper, and also Stefano De Marchi, Marco Caliari (University of Verona, Italy) and Marco Vianello (University of Padova, Italy) for their advice about the Padua points interpolation. This work is supported by the EPSRC grant number EP/E012248/1.

References

[1] Thomas Bagby, Len Bos and Norman Levenberg, Multivariate simultaneous approximation, Constr. Approx., 4, 18 (2002).

[2] Len Bos, Marco Caliari, Stefano De Marchi, Marco Vianello and Yuan Xu, Bivariate Lagrange interpolation at the Padua points: the generating curve approach, J. Approx. Theory, 1, 143 (2006).

[3] Len Bos, Stefano De Marchi, Marco Vianello and Yuan Xu, Bivariate Lagrange interpolation at the Padua points: the ideal theory approach, Numerische Mathematik, 1, 108 (2007).

[4] L. Brutman, Lebesgue functions for polynomial interpolation – a survey, Ann. Numer. Math., 4 (1997)

[5] Marco Caliari, Stefano De Marchi and Marco Vianello, Bivariate polynomial interpolation at new nodal sets, Appl. Math. Comput., 2, 165 (2005).

[6] Mark S. Drew, Jie Wei and Ze-Nian Li, Illumination–invariant image retrieval and video segmentation, Pattern Recognition, 32 (1999).

[7] J. M. Geusebroek, G. J. Burghouts and A. W. M. Smeulders, The Amsterdam Library of Object Images, Int. J. Comp. Vision, 1, 61 (2005).

[8] Leslie Greengard and John Strain, The Fast Gauss Transform, SIAM J. Sci. Stat. Comput., 1, 12 (1991).

[9] S. A. Nene, S. K. Kajar and H. Murase, Columbia Object Image Library (COIL-100), Technical Report CUCS-006-96 (1996).

[10] Michael J. Swain and Dana H. Ballard, Color indexing, Int. J. Comp. Vision, 1, 7 (1991).

Author Biography

Roberto Montagna received his BS in computer science in 2004 and his MS in intelligent and multimedia systems in 2007 from the University of Verona (Italy). Since 2007 he is PhD student at the University of East Anglia, Norwich (UK) under the supervision of Prof. Graham Finlayson.