FAST SCALE-SPACE FEATURE REPRESENTATIONS BY GENERALIZED INTEGRAL IMAGES

Konstantinos G. Derpanis, Erich T. H. Leung and Mikhail Sizintsev

Department of Computer Science and Engineering, York University
{kosta, leung, sizints}@cse.yorku.ca

ABSTRACT

In this paper, we place the integral image-based approach for multi-scale feature construction, popularized by Viola and Jones, into a common framework of understanding. The integral image within this framework represents space variant image filtering with the zero-order B-spline. Given this framework, we propose efficiently computable higher-order B-spline image features based on generalized integral images that have the potential to be more accurate, yet efficient as compared to previous integral image-based efforts.

Index Terms— scale-space, B-spline, integral image, interest point, feature descriptor

1. INTRODUCTION

Recently, we have witnessed a resurgence in the research of local feature detectors/descriptors and their applications. Given their demonstrated potential for successful application in various contexts, several researchers have turned their attention to efficient computational (approximation) schemes that do not substantially sacrifice performance (e.g., [1, 2, 3]). In this paper we show that these approaches represent a special case (the coarsest model) of a more general theoretical framework, that allows more accurate, yet efficiently computable multi-scale feature representations.

Concerns with fast computational techniques are also shared by the research in multi-scale representations that embed the original image into a one parameter family of derived images, where each derived image contains structures limited to a range of scales. The Gaussian-based linear scale-space paradigm, for example, constructs the derived representations with desirable multi-scale properties. Gaussian kernel convolution is, however, too resource demanding for many visual applications. Spline generated scale-spaces represent an alternative for fast realizations of multi-scale decompositions of images [4, 5]. Indeed, spline-based filtering represents a general class of multi-scale generators that include Gaussian-based linear filtering as its limiting case.

In this paper we explore an efficient approach based on generalized integral images for realizing space variant image descriptors by n^{th} order B-spline filtering. Indeed, the integral image representation used in recent applications represents a special case, namely, zero-order B-spline filtered representations realized by the use of the first-order integral image. Similarly, existing multi-scale techniques, such as cascaded uniform filtering and the Gaussian pyramid, are but a form of B-spline filtering [5]. In either case, our paper provides a more general framework that allows deeper insights into these approaches rather than a radical departure from them.

In the following, Section 2 discusses the B-spline scale-space and an efficient non-recursive realization based on the generalized integral image formulation. Section 3 concludes with a discussion of two visual applications that may profit from our approach.

2. ANALYSIS AND COMPUTATION

2.1. B-spline functions

A continuous B-spline of order n is defined recursively using the zero-order (centred) B-spline of width T as

$$\beta_T^0(x) = \begin{cases} 1/T, & \mid x \mid < T/2 \\ 0, & \text{otherwise} \end{cases} \quad \text{and,} \tag{1}$$

$$\beta_T^n(x) \;=\; \beta_T^{n-1}(x) * \beta_T^0(x), \tag{2}$$

where $*$ denotes the convolution operator, and β^n is the n^{th} order B-spline[1]. Thus, a B-spline can be generated by convolving a rectangular pulse with itself n times.

The B-spline space can be seen as a more general multiresolution function space with the Gaussian representing its limiting case. Any square integrable signal can be represented as a weighted sum of shifted and dilated B-splines in nested spaces of spline functions [5]. The B-spline of degree n is n times continuously differentiable except at the knot points which are $n-1$ times differentiable by construction [6]. B-spline kernels preserve the analytical properties of its Gaussian counterpart very well due to their fast convergence, where the variance of the n^{th} order B-spline, $\sigma_n^2 = \frac{T^2(n+1)}{12}$ [5]. The cubic B-spline provides a very close approximation of the Gaussian function. Furthermore, even a lower order

[1] $\beta^n(x)$ denotes the n^{th} order centred B-spline generated with a rectangular pulse of length $T = 1$. Similar notation is also applied to the discrete version (see below).

Algorithm 1 Generalized integral image computation

1: *// pre-computation*
2: set initial integral image, $I_n(x, y)$, as input image
3: **for each** integral image order **do**
4: **for each** image point **do**
5: increase cumulative row sum, $c(x)$
6: $I_n(x, y) = I_n(x, y-1) + c(x)$
7: **end for**
8: **end for**
9: *// apply filter with a given kernel position, size and order*
10: compute intersection
11: normalize

may be sufficient for most applications. The discrete sampled B-spline $b_T^n(k)$ of order n can be generated by sampling its continuous counterpart at the scale $T \geq 1$ [5]:

$$b_T^n(k) = \frac{1}{T}\beta^n\left(\frac{k}{T}\right), \forall k \in \mathbb{Z}. \quad (3)$$

These results can be easily extended to higher-dimensional signal spaces using the tensor product splines, for example, the 2-D case is given by $\beta^n(x, y) = \beta^n(x)\beta^n(y)$.

2.2. Generalized integral images

The concept of the integral image was introduced in [7, 8] and later in [9] for the purpose of enabling constant time filtering with axis aligned rectangular filters (i.e., uniform B-spline). This section focuses on the generalization of the integral image that allows for non-recursive axis-aligned filtering with the B-spline kernel of order n. This generalization reported by several authors [10, 11] relies on repeated integration.

The key identity for formalizing the generalized integral image is:

$$f * g = \left(\int^x f(x')dx'\right) * \left(\frac{d^n g}{dx^n}\right) = f_n * g_{-n}, \quad (4)$$

where $*$ denotes the continuous convolution operator, subscript n denotes n-fold partial integration and subscript $-n$ denotes n-fold differentiation.

For the case of n-fold convolution of the box filter, $\beta_T^0(x)$, with the input signal, $I(x)$, from (4) it can be shown [10, 11] that this operation reduces to weighted sampling the (precomputed) n-fold partial summed image:

$$I_{\text{smooth}} = I(x) * b_T^n(x) = \frac{1}{T^n}\sum_{i=0}^{n}(-1)^i \binom{n}{i} I_n(x + \frac{Tn}{2} - iT), \quad (5)$$

where T^n is the normalization factor; see Algorithm 1 for an algorithmic presentation of (5).

The 1-D formulation extends easily to higher dimensional signals due to the separable definition of the B-spline. For 2-D, denoted $b^n(x_1, x_2) = b^n(x_1)b^n(x_2)$, $(n+1)^2$ weighted samples are required; Fig. 1 lists the weighted sampling coefficients for integral images of orders 1 to 3.

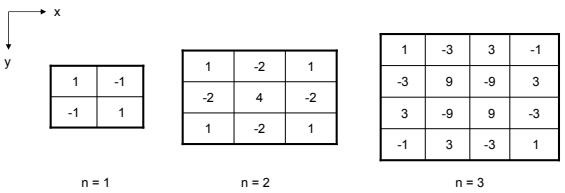


Fig. 1. Weighted sampling coefficients for low order nth order integral images. Note that the distance between samples is a function of the scale of the filter, T (see Eq. 5).

Finally, given the integral image computation we recover derivative measurements by performing numerical differentiation through Taylor series expansion, e.g., the first and second derivatives are given by,

$$I'_{\text{smooth}}(x) = (I^+_{\text{smooth}}(x) - I^-_{\text{smooth}}(x))/2 \quad (6)$$

$$I''_{\text{smooth}}(x) = I^+_{\text{smooth}}(x) - 2I_{\text{smooth}}(x) + I^-_{\text{smooth}}(x), \quad (7)$$

resp., where $I^+(x) = I(x+1)$ and $I^-(x) = I(x-1)$.

2.3. Computational costs

Low computational cost is the main motivation for using integral images. Here we show that the higher-order integral images conserve this useful feature (while providing a means for computing a better approximation to classical scale-space, see Section 2.1) by considering the cost of integral images of various orders in comparison to other well-established techniques. Similar to [2], we compare the integral images of various orders with the following Gaussian filter approaches: *non-separable/separable/recursive Gaussian* and *global FFT-based filtering*.

The total cost consists of four major components:

$$\text{cost} = wh(c_a n_a + c_m n_m + c_b n_b + c_t n_t), \quad (8)$$

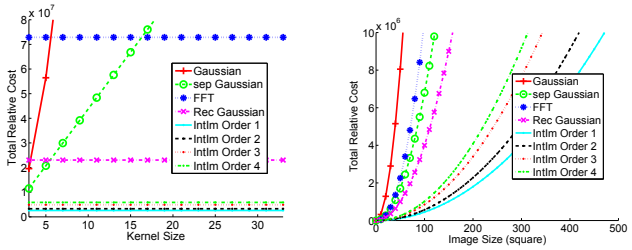
where w and h denote the width and height of the input image, resp.; c_a, c_m, c_b and c_t the cost of addition, multiplication, bit shift and type conversion, resp.; n_a, n_m, n_b and n_t denote the number of operations.

Assuming an integer-based input image, the integral image of order n requires one floating-point multiplication for normalization (hence, one type conversion), $2n$ integer additions for construction of the integral image, $(n+1)^2 - 1$ integer additions and $(n+1)^2$ integer multiplications for actual filter application. However, in the case of integers, it is possible to optimize the computation further by substituting the multiplications with bit shifts and additions. Table 1 compares the costs of all considered filtering techniques.

Next, we consider the relative cost of processor operations measured against the cost of integer addition and take the costs reported in [12] as a reference measure. [12] states

Filter technique	Complexity	
	Multiplication	Addition
Gaussian	N^2	N^2-1
Separable Gaussian	$2N$	$2(N-1)$
FFT	$2\log(w \cdot h)$	$2\log(w \cdot h)$
Recursive Gaussian	14	6
n^{th} *Order Integral Image*	$1+(n+1)^2$	$2n+(n+1)^2-1$

Table 1. Comparison of various 2-D linear filtering approaches (operations per pixel), where N, w and h correspond to kernel size (assuming square dimensions), image width and height, resp.. Adapted from [2].



(a) 256×256 image size (b) 9×9 kernel size

Fig. 2. Comparison of total relative computational cost for various 2-D Gaussian filtering techniques.

that integer addition and bit shift cost 1 unit each, integer multiplication costs 4 units, and type conversion, floating-point multiplication and addition cost 20 units each. Note, however, the relative costs will vary across architectures.

Figure 2 shows the relative costs of higher-order integral images up to order four and their alternatives. Notice that the costs for the integral image-based filtering are consistently below the costs of the other methods, where costs increase with order. Also, the running time of filtering via integral images does not depend on the kernel size. Importantly, higher-order integral images are still significantly more efficient than the recursive/separable Gaussian or FFT approaches.

Aside from their low relative cost, integral images are non-recursive, i.e., one can directly extract scale information for any region without prior construction of the "entire" scale space. Such a flexibility is extremely important for sparse feature computation (e.g., [13]), and cascade-type object detectors (e.g., [9]) where information is extracted only when required, thus avoiding unnecessary expensive computations. Significantly, out of all the non-integral-image filtering approaches in Table 1, only the non-separable Gaussian possesses this feature, however, it is significantly more expensive (especially at coarse scales).

In practice, care must be taken to avoid arithmetic overflow. For an input image of width and height 2^w and 2^h, resp. and image intensity resolution, b (in bits), the worst case memory resolution required for an n^{th} order integral image is $\log_2((2^{w+h})^n 2^b) = (w+h)n+b$ bits per pixel. For example, a third-order integral image of a 512×512 input image with intensity resolution $b = 8$ requires 62 bits per pixel, which is within the bounds of current 64-bit CPU architectures.

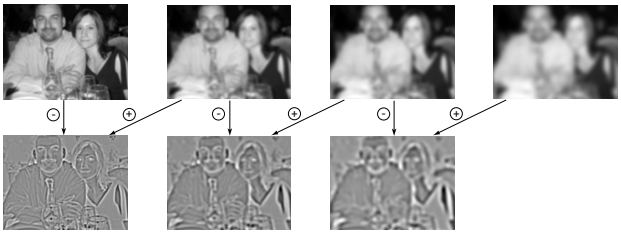


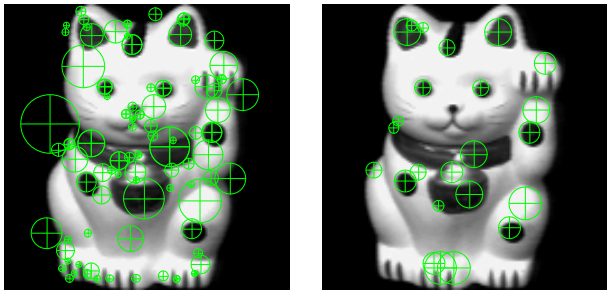
Fig. 3. Multi-scale example. Top row: multi-scale quadratic B-spline representation at scales $T = 0, 2, 4$ and 6 (left to right). Bottom row: DoB representation.

To reduce the number of bits per entry, image subdivision techniques have been proposed (for details, see [7]). Also, the image value range can be shifted such that the range spans both positive and negative values. This has the effect of removing the monotonicity of the integral image and thus reduces the maximum value reached.

3. DISCUSSION

The framework presented in this paper not only introduces new insights into many existing visual applications, but also opens new possibilities. In this section, we discuss this advantage through two applications, namely, interest point detection and steerable filters. The source code is available at: `http://www.cse.yorku.ca/~kosta/Generalized_Integral_Image/gii_main.html`.

The *difference of Gaussian* (DoG) is a popular means for identifying multi-scale key-points (e.g., [13]). It is an efficient approximation of the scale-normalized *Laplacian of Gaussian* (LoG) representation that is used to identify blob-like structures in the image. The DoG is recovered by taking differences between adjacent levels of a Gaussian scale-space representation. Given the DoG, key-points are identified by a local maxima search in space and scale. To accelerate the DoG construction, Grabner *et al*. [2] approximate the Gaussian filtering by box filtering using the (first order) integral image, they term the resulting images *difference of mean* (DoM) images. An advantage of the DoM over the DoG, is that it does not rely on subsampling, rather computations are done at the spatial resolution of the input image; the resulting localized key-points are spatially accurate within a pixel. Thus, a costly spatial interpolation post-processing step (e.g., [13]) is avoided. A drawback of the box filtering approach is that it may introduce distracting spurious structures in the form of Mach bands. In addtion, due to the pronounced non-isotropic nature of the box filter one can expect a reduction in rotation invariance. This can be clearly seen in Grabner *et al*.'s experiments where their DoM detector yields its worst result at $45°$. The DoM can be interpreted as filtering with the zero-order B-spline. Rather than limit filtering to zero-order, the DoG may be approximated by B-splines of higher-order that may increase accuracy while maintaining efficiency when computed with the generalized integrals, we term this generalization the *difference of B-spline* (DoB) representation (see



(a) DoG (b) DoB (first-order B-spline)

Fig. 4. Comparison of detected key-points (marked in green) found by Lowe's DoG detector and our DoB detector. Input image courtesy of Michael Grabner.

Fig. 3). Figure 4 provides a comparison between key-points detected using Lowe's DoG detector [13] and our first-order DoB detector. Notice that the DoB detects the prominent blob-like structures very well. In a future correspondence we will present a quantitative comparison between the DoG and our DoB detectors.

Steerable filters [14] are a class of filters where a filter of arbitrary orientation is synthesized by a linear combination of K basis filters, denoted $f_i(\mathbf{x})$, formally, $f(\mathbf{x};\theta) = \sum_{i=1}^{K} k_i(\theta) f_i(\mathbf{x})$. Gaussian derivatives are a widely used class of steerable filters, where the size of the basis is equal to one greater than the derivative order. For example, the first derivative of the Gaussian, G, at an arbitrary orientation θ is given by, $G_1(\theta) = \cos(\theta)\frac{\partial G}{\partial x} + \sin(\theta)\frac{\partial G}{\partial y}$. Villamizar *et al.* [3] propose to approximate the steered Gaussian derivatives by replacing the Gaussian derivative by Haar-like filters and use the (first-order) integral image for fast computation. The vertical Haar filter, $h(x,y)$, can be seen as a special case of the derivative of a B-spline, specifically, the derivative of the first-order B-spline along the x-axis with zero-order blurring along the y-axis., formally,

$$h(x,y) = \frac{d\beta^1(x)}{dx}\beta^0(y). \tag{9}$$

Given that higher-order B-splines provide better approximations of the Gaussian kernel, it suggests the use of the derivative of higher-order B-splines as basis filters (see Fig. 5 for an example). Obviously, the need for better accuracy in filter representation ultimately depends on its application. The use of the steerable Gaussian derivatives may provide richer yet efficiently computable features for feature selection-based learning approaches (e.g., [9]). Levi and Weiss [15] demonstrate that the use of richer features than the standard linear features of Viola and Jones [9] reduce the number of training examples required. Furthermore, they provide a means for rotation-invariant feature representations achieved by rotating responses based on a canonical orientation (e.g., [3]).

In summary, this paper develops in a principled manner a general framework that provides new insights into several

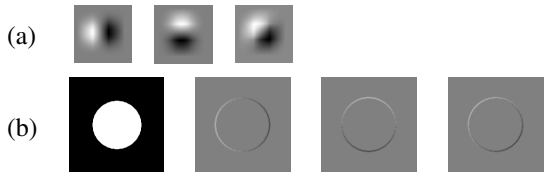


Fig. 5. Steerable filter example. (a) first derivative of the quadratic B-spline in the x and y directions (basis) and the steered result ($\pi/4$ rad.) (left to right). (b) Circular disk with its basis images and the steered result ($\pi/4$ rad.) (left to right).

existing approaches to multi-scale image description and feature representation. This framework includes as special cases, the first-order integral image and Gaussian multi-scale representations. Thus, not only are the desirable properties of both techniques preserved (e.g., efficiency) but further advantages are also acquired. Finally, we have presented two of a multitude of potential applications of this generalized theory.

4. REFERENCES

[1] H. Bay, T. Tuytelaars, and L.J. Van Gool, "SURF: Speeded up robust features," in *ECCV*, 2006, pp. I: 404–417.

[2] M. Grabner, H. Grabner, and H. Bischof, "Fast approximated SIFT," in *ACCV*, 2006, pp. I:918–927.

[3] M. Villamizar, A. Sanfeliu, and J. Andrade-Cetto, "Computation of rotation local invariant features using the integral image for real time object detection," in *ICPR*, 2006, pp. IV: 81–85.

[4] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing. I. Theory & II. Efficiency design and applications," *IEEE Trans. Sig. Proc.*, vol. 41, no. 2, pp. 821–848, Feb. 1993.

[5] Y.P. Wang and S.L. Lee, "Scale-space derived from B-splines," *PAMI*, vol. 20, no. 10, pp. 1040–1055, Oct. 1998.

[6] M. Unser, "Splines: A perfect fit for signal and image processing," *IEEE Sig. Proc. Mag.*, vol. 16, no. 6, pp. 22–38, 1999.

[7] F.C. Crow, "Summed-area tables for texture mapping," in *SIGGRAPH*, 1984, pp. 207–212.

[8] L.A. Ferrari and J. Sklansky, "A fast recursive algorithm for binary-valued two-dimensional filters," *CVGIP*, vol. 26, no. 3, pp. 292–302, June 1984.

[9] P. Viola and M.J. Jones, "Robust real-time face detection," *IJCV*, vol. 57, no. 2, pp. 137–154, May 2004.

[10] L.A. Ferrari, P.V. Sankar, J. Sklansky, and S. Leeman, "Efficient two-dimensional filters using B-spline functions," *CVGIP*, vol. 35, no. 2, pp. 152–169, Aug. 1986.

[11] P.S. Heckbert, "Filtering by repeated integration," in *SIGGRAPH*, 1986, pp. 315–321.

[12] S. Oualline, *Practical C++ Programming*, O'Reilly, 1995.

[13] D.G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[14] W.T. Freeman and E.H. Adelson, "The design and use of steerable filters," *PAMI*, vol. 13, no. 9, pp. 891–906, Sept. 1991.

[15] K. Levi and Y. Weiss, "Learning object detection from a small number of examples:the importance of good features," in *CVPR*, 2004, pp. II: 53–60.