Integral image-based representations

Konstantinos G. Derpanis
Department of Computer Science and Engineering
York University
kosta@cs.yorku.ca

Last Updated: July 14, 2007

In this note the *integral image* representation (Viola & Jones, 2001) (Section 1) and its three-dimensional generalization, the *integral video* (Ke, Sukthankar & Hebert, 2005) (Section 2), are reviewed. These representations admit very fast multi-scale feature recovery. In Section 3, the use of integral representation for scale-space analysis is briefly discussed. More generally, the integral image-based approach described here can be cast into a more general framework of understanding. The integral image within this framework represents space variant image filtering with the zero-order B-spline. If an image is pre-integrated n times, space variant filtering with higher-order B-splines can be achieved. For further details on the higher-order generalization, see (Derpanis, Leung & Sizintzev, 2007).

1 Integral image representation

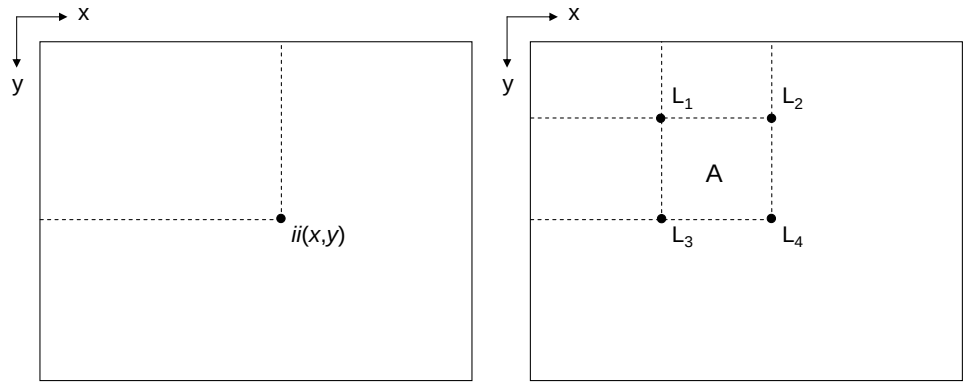
Rectangular two-dimensional image features can be computed rapidly using an intermediate representation called the integral image (c.f. *summed area tables* used in graphics (Crow, 1984)). The integral image, denoted $ii(x, y)$, at location (x, y) contains the sum of the pixel values above and to the left of (x, y) (see Fig. 1(a)), formally,

$$ii(x, y) = \sum_{\substack{x' \leq x \\ y' \leq y}} i(x', y'), \tag{1}$$

where $i(x, y)$ is the input image. The integral image can be computed in one-pass over the image using the following recurrence relation:

$$s(x, y) = s(x, y - 1) + i(x, y) \tag{2}$$

$$ii(x, y) = ii(x - 1, y) + s(x, y), \tag{3}$$



(a) (b)

Figure 1: Integral image representation. (a) integral image and (b) region A can be computed using the following four array references: $L_4 + L_1 - (L_2 + L_3)$.

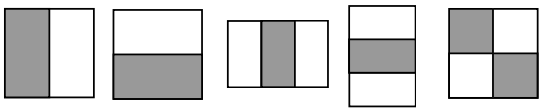


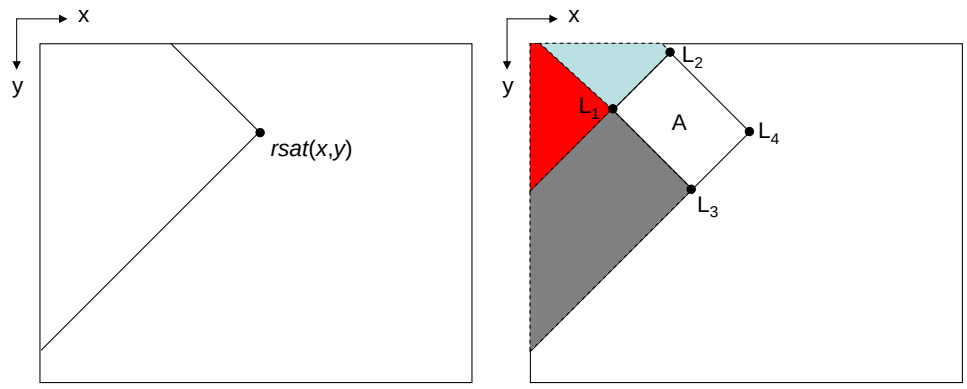
Figure 2: The set of rectangular filters used by Viola and Jones (Viola & Jones, 2001). Each filter is composed of several rectangular regions, where the white and grey regions within the features are associated with 1 and -1, respectively.

where $s(x, y)$ denotes the cumulative row sum and $s(x, -1) = ii(-1, y) \equiv 0$.

Given the integral image, the sum of pixel values within a rectangular region of the image aligned with the coordinate axes can be computed with four array references (i.e., constant time). For example, to compute the sum of region A in Fig. 1(b), the following four references are required: $L_4 + L_1 - (L_2 + L_3)$. Viola and Jones (Viola & Jones, 2001) propose a set of five rectangular filters to extract features for object detection (see Fig. 2). Given the integral image, each of these features can be computed in constant time.

For 45° oriented rectangular features, Lienhart and Maydt (Lienhart & Maydt, 2002) proposed an adaption of the integral image representation, they termed the *rotated summed area table*, denoted $rsat(x, y)$. The $rsat$ data structure yields the sum of pixels of the rectangle rotated by 45° with the rightmost corner located at (x, y) and extending to the image boundaries (see Fig. 3(a)):

$$rsat(x, y) = \sum_{\substack{x' \leq x \\ x' \leq x - |y - y'|}} i(x', y'). \tag{4}$$



(a) (b)

Figure 3: Rotated summed area table representation. (a) rotated summed area table and (b) region A can be computed using the following four array references: $L_4 + L_1 - (L_2 + L_3)$.

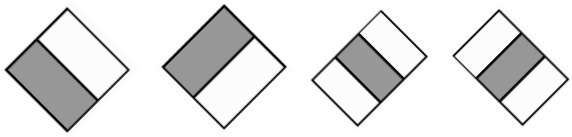


Figure 4: A subset of the oriented rectangular filters used by Lienhart and Maydt (Lienhart & Maydt, 2002). Each filter is composed of several 45° oriented rectangular regions, where the white and grey regions within the features are associated with 1 and -1, respectively.

$rsat(x, y)$ can be computed efficiently with two passes of the image. Rotated rectangles can be computed, like the integral image, with four table lookups. For example, to compute the sum of region A in Fig. 3(b), the following four references are required: $L_4 + L_1 - (L_2 + L_3)$. Figure 4 depicts several oriented rectangular feature prototypes used in (Lienhart & Maydt, 2002).

2 Integral volume representation

The integral video represents the three-dimensional generalization of the integral image. The integral video, denoted $iv(x, y, t)$, at spatial location (x, y) and time t

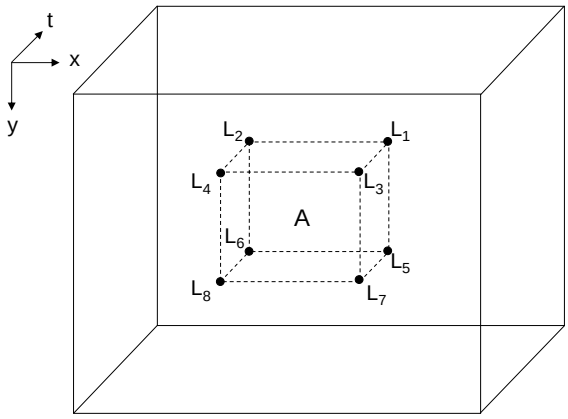


Figure 5: Integral video feature computation. Volume A can be computed using the following 8 array references: $L_5 - L_1 - L_6 - L_7 + L_2 + L_3 + L_8 - L_4$.

contains the sum of pixel values less than or equal to (x, y, t), formally,

$$iv(x, y, y) = \sum_{\substack{x' \leq x \\ y' \leq y \\ t' \leq t}} i(x', y', t'), \tag{5}$$

where $i(x, y, t)$ is the input image sequence. The integral video can be computed rapidly in one pass over the image sequence using the following recurrence relation,

$$s_1(x, y, t) = s_1(x, y - 1, t) + i(x, y, t) \tag{6}$$

$$s_2(x, y, t) = s_2(x - 1, y, t) + s_1(x, y, t) \tag{7}$$

$$iv(x, y, t) = iv(x, y, t - 1) + s_2(x, y, t), \tag{8}$$

where $s_1(x, -1, t) = s_2(-1, y, t) = iv(x, y, -1) \equiv 0$. In order to compute the sum within any rectangular volume aligned with the coordinate axes only eight array references to the integral video are necessary. For example, the volume of box A depicted in Fig. 5 can be computed by the following eight references to the integral video: $L_5 - L_1 - L_6 - L_7 + L_2 + L_3 + L_8 - L_4$. Ke et al. (Ke, Sukthankar & Hebert, 2005) propose a set of four cuboid filters to extract features for action detection (see Fig. 6). As with the case of integral images, each of these volumetric features can be computed in constant time.

3 Integral-based scale-space approximations

The Gaussian kernel and its partial derivatives can be shown to provide the unique set of operators for the construction of (linear) scale-space under certain conditions (ax-

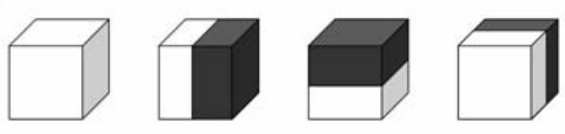


Figure 6: The set of cuboid filters used by Ke et al. (Ke, Sukthankar & Hebert, 2005). Each filter is composed of cuboid volumes, where the white and black volumes within the features are associated with 1 and -1, respectively.

ioms) (Koenderink, 1984). However, in practice the Gaussian needs to be discretized and cropped. Thus, the use of the Gaussian kernel in practice is of an approximative nature. Grabner et. al (Grabner, Grabner & Bischof, 2006) and Bay et. al (Bay, Tuytelaars & Van Gool, 2006) push the approximation even further with box filters for the purpose of rapidly computing approximations of Lowe's SIFT features (Lowe, 2004). These box filter-based approximations are constructed through the use of integral images. The respective papers report comparable performance to the discretized and cropped Gaussian.

References

Bay, H., Tuytelaars, T. & Van Gool, L. (2006). SURF: Speeded up robust features. In *European Conference on Computer Vision* (pp. I: 404–417).

Crow, F. (1984). Summed-area tables for texture mapping. *SIGGRAPH*, *84*, 207–212.

Derpanis, K. G., Leung, E. T. H. & Sizintzev, M. (2007). Fast scale-space feature representations by generalized integral images. Technical report, York Unviersity.

Grabner, M., Grabner, H. & Bischof, H. (2006). Fast approximated SIFT. In *Asian Conference on Computer Vision* (pp. I:918–927).

Ke, Y., Sukthankar, R. & Hebert, M. (2005). Efficient visual event detection using volumetric features. In *IEEE International Conference on Computer Vision* (pp. I: 166–173).

Koenderink, J. (1984). The structure of images. *Biological Cybernetics*, *50*, 363–370.

Lienhart, R. & Maydt, J. (2002). An extended set of Haar-like features for rapid object detection. In *IEEE Computer Vision and Pattern Recognition* (p. I:900:903).

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60(2)*, 91–110.

Viola, P. & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *IEEE Computer Vision and Pattern Recognition* (pp. I:511–518).