Adaptive Step Size Window Matching for Detection

Nathan Mekuz, Konstantinos G. Derpanis and John K. Tsotsos
Department of Computer Science and Center for Vision Research (CVR)
York University, Toronto, ON, Canada M3J 1P3

Abstract

An often overlooked problem in matching lies in selecting an appropriate step size. The selection of the step size for real-time applications is critical both from the point of view of computational efficiency and detection performance. Current systems set the step size in an ad hoc manner. This paper describes an algorithm for selecting the step size based on a theoretical worst case analysis. We have implemented this adaptive step size method in an object detection algorithm. Experimental evaluation demonstrates the effectiveness of our proposed algorithm.

1 Introduction

Matching window techniques represent popular methods in computer vision for matching regions in one image to another. They constitute a central component of many important problems in computer vision such as stereo reconstruction (e.g., [2]), optical flow (e.g., [1, 3]), structure from motion (e.g., [8]) and object detection (e.g., [9]). Standard matching window techniques in the literature include: sum of squared differences (SSD), sum of absolute differences (SAD) and normalized cross correlation [5].

Several aspects of matching window methods that have previously been addressed include window size [6] and illumination invariance [4]. In this paper, our focus is on computational efficiency aspects of window matching approaches.

A match is determined by selecting the position of "maximum similarity" between the target image and a windowed image region considered at all possible positions of the search image. Given a target size of $m \times n$ in a search image of size $M \times N$, the complexity of an exhaustive search is $\mathcal{O}((M-m)(N-n)mn)$, assuming a similarity metric that can be computed in linear time. Several approaches have been proposed to reduce the computational burden, such as pyramid methods [1, 3, 10] and Fourier methods [5]. Rather than perform these image transformations and incur the additional costs, many search window implementations opt for applying the similarity computation on a subset of all possible positions by using a fixed step size greater than one. For instance, Viola and Jones [9] use a step size of 1.5. These choices are usually accompanied by the fuzzy argument that the distance metric changes gradually over space thus the risk of missing the target by skipping pixels is low. In this paper, we propose an adaptive step size algorithm for image search based on a worst-case analysis of the template image computed offline.

1.1 Outline of paper

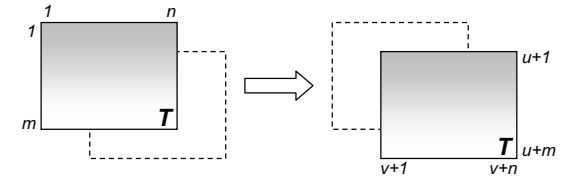
The remainder of this paper is subdivided into four main sections. Section 2 introduces the formulation of the problem and provides a theoretical analysis of step size selection. Section 3 outlines an efficient implementation of our adaptive step size scheme. Section 4 summarizes the empirical evaluation of an algorithmic instantiation of our analysis. Finally, Section 5 provides a summary.

2 Technical approach

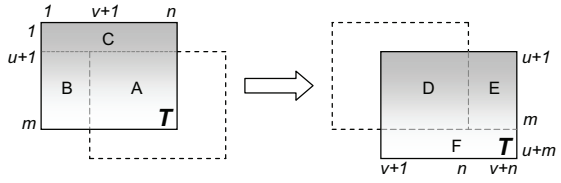
In this paper, we limit our discussion to intensity images with intensity values in the range $[I_L, I_H]$. A target object T of size $m \times n$ is searched for at all possible positions in a search image S of size $M \times N$. In this paper we present an analysis of the SSD matching metric. SSD (sum of squared differences) is a widely used similarity metric, due to its simplicity and efficient computation (linear time complexity in the size of the input images). SSD measures distance or dissimilarity between two input images; low SSD scores indicate a high degree of similarity between the two images. Using row-major indexing (matrix notation), the SSD between images P and Q is given by,

$$SSD(P,Q) := \sum_i \sum_j (P_{i,j} - Q_{i,j})^2. \qquad (1)$$

The window scanning technique slides a window over the search image S, and computes the SSD distance between the image segment defined by the window and the



a.



b.

Figure 1. a. The $m \times n$ target image translated by a displacement of (u, v) from position $(1, 1)$ in the search image to position $(u+1, v+1)$. b. A subdivision of the original and translated images into areas, to facilitate analysis. Note that areas A and D overlap.

target image T at each location. An SSD score below some set threshold is considered a match. As the window is moved over the search image, the SSD score changes gradually over the search space. The exact change at each pixel depends on the search image, but the maximum possible change as a result of a spatial shift by any given offset can be computed offline for target image T, independently of the search image, S, as described below. Given a known SSD score at point p, this worst-case analysis leads to an adaptive step size selection at location p, effectively eliminating from the search an area around p where the SSD score is guaranteed to be above the set threshold. Note that for our purpose, the worst-case analysis is only concerned with maximum negative changes in SSD.

For ease of exposition and without loss of generality, we set the coordinate system so that pixel p (where the SSD score is known) is at $(1, 1)$, as depicted in Figure 1(a), and derive a lower bound for the SSD score at location $(u + 1, v + 1)$, for $0 \leq u < m$, $0 \leq v < n$. A similar approach may be used to derive the lower bound for SSD scores at shifts with $v \leq 0$. Using the decomposition depicted in Figure 1(b), the SSD distances at $(1, 1)$ and $(u + 1, v + 1)$ can be written as,

$$d_{1,1} = A + B + C \tag{2}$$

$$d_{u,v} = D + E + F \tag{3}$$

where

$$A = \sum_{i=u+1}^{m} \sum_{j=v+1}^{n} (S_{i,j} - T_{i,j})^2 \tag{4}$$

$$B = \sum_{i=u+1}^{m} \sum_{j=1}^{v} (S_{i,j} - T_{i,j})^2 \tag{5}$$

$$C = \sum_{i=1}^{u} \sum_{j=1}^{n} (S_{i,j} - T_{i,j})^2 \tag{6}$$

$$D = \sum_{i=u+1}^{m} \sum_{j=v+1}^{n} (S_{i,j} - T_{i-u,j-v})^2 \tag{7}$$

$$E = \sum_{i=1}^{m-u} \sum_{j=n+1-v}^{n} (S_{i+u,j+v} - T_{i,j})^2 \tag{8}$$

$$F = \sum_{i=m+1-u}^{m} \sum_{j=1}^{n} (S_{i+u,j+v} - T_{i,j})^2 \tag{9}$$

Let $d_{y,x}$ denote the SSD score at location (y, x). We define $\Delta_{u,v}$ as the drop in SSD score resulting from a displacement (u, v), (e.g., from $(1, 1)$ to $(u + 1, v + 1)$),

$$\Delta_{u,v} := d_{1,1} - d_{u+1,v+1}. \tag{10}$$

We are interested in finding the upper bound of $\Delta_{u,v}$, given target image T over the domain of all values of $I_L \leq S_{i,j} \leq I_H$. Since components B, C, E and F are independent, from equations (2), (3) and (10),

$$\begin{aligned} \sup(\Delta_{u,v}) &= \sup(d_{1,1}) - \inf(d_{u,v}) \\ &= \sup(A - D) + \sup(B) + \sup(C) \\ &\quad - \inf(E) - \inf(F) \end{aligned} \tag{11}$$

where $\sup(X)$ denotes the supremum (least upper bound) and $\inf(X)$ denotes the infimum (greatest lower bound) [7]. Since $I_L \leq T_{i,j} \leq I_H$ and $I_L \leq S_{i,j} \leq I_H, \forall i, j$,

$$\inf(E) = \inf(F) = 0. \tag{12}$$

Since the derivative of components B and C with respect to $S_{i,j}$ is linear, their supremum lies at an extremum of $S_{i,j}$,

$$\sup(B) = \sum_{i=u+1}^{m} \sum_{j=1}^{v} \max\{(I_H - T_{i,j})^2, (I_L - T_{i,j})^2\} \tag{13}$$

$$\sup(C) = \sum_{i=1}^{u} \sum_{j=1}^{n} \max\{(I_H - T_{i,j})^2, (I_L - T_{i,j})^2\}. \tag{14}$$

The supremum for $A - D$ is derived in Eq. (15) and (16). Given any target image, $\sup(\Delta_{u,v})$ can be computed offline, effectively resulting in a two-dimensional map of the largest negative change in SSD score possible as a result of a spatial shift. Figure 2 presents several objects and

$$A - D = \sum_{i=u+1}^{m} \sum_{j=v+1}^{n} (S_{i,j} - T_{i,j})^2 - (S_{i,j} - T_{i-u,j-v})^2$$

$$= \sum_{i=u+1}^{m} \sum_{j=v+1}^{n} (T_{i,j} - T_{i-u,j-v})(T_{i,j} + T_{i-u,j-v} - 2S_{i,j}) \quad (15)$$

$$\sup(A - D) = \sum_{\substack{i=u+1\ldots m \\ j=v+1\ldots n \\ T_{i,j} \geq T_{i-u,j-v}}} (T_{i,j} - T_{i-u,j-v})(T_{i,j} + T_{i-u,j-v} - 2I_L)$$

$$+ \sum_{\substack{i=u+1\ldots m \\ j=v+1\ldots n \\ T_{i,j} \leq T_{i-u,j-v}}} (T_{i,j} - T_{i-u,j-v})(T_{i,j} + T_{i-u,j-v} - 2I_H) \quad (16)$$

their $\sup(\Delta_{u,v})$ maps depicted as contour maps. As (u, v) moves away from the zero position, representing larger shifts, $\sup(\Delta_{u,v})$ generally grows, indicating larger maximum possible drops in SSD score as a result of translation.

3 Efficient storage and search

The offline computation of $\sup(\Delta_{u,v})$ enables a faster matching search by eliminating significant areas of search image S from the search space. If the computation of SSD at location p yields a score that is higher than the target threshold by δ, then by superimposing the contour map of the model around point p, a region bounded by the δ contour may be eliminated.

A naive implementation may store a boolean mask over search image S to keep track of which search locations have been tested or eliminated from the search by previous computations. However, such a brute force algorithm would still be expensive, as it would require $\mathcal{O}(m \times n)$ additional operations at each location to compute the region that may be excluded from the search.

We propose a more efficient approach. Instead of storing the $\sup(\Delta_{u,v})$ contour map as a matrix, we store a list of triplets $\langle u, v, \sup(\Delta_{u,v}) \rangle$, sorted offline by values of $\sup(\Delta_{u,v})$. Each SSD computation is followed by a sequential scan of the list until an entry is found where $\sup(\Delta_{u,v}) > \delta$. At each entry (u, v) where $\sup(\Delta_{u,v}) \leq \delta$ one pixel is eliminated from the search space. The amortized cost of traversing this list for the entire search is proportional to the size of the search image S, since the operation required to eliminate one pixel is of constant time (vs. $\mathcal{O}(m \times n)$ in the naive computation).

Step size	Total SSD computations			False positives
	a	b	c	
1	36,494	230,919	160,640	no
1.5	16,340	102,758	71,690	possibly
2	9,159	57,974	40,160	possibly
Adaptive	**4,443**	**32,516**	**14,379**	**no**

Table 1. The number of SSD computations performed by different algorithms on the object detection experiments (columns a, b and c) depicted in Figure 3, and whether false positives are introduced as a result of the step size used.

4 Empirical evaluation

To test the effectiveness of our technique, we conducted the experiments depicted in Figure 3. The detection of the model objects on the left was carried out using SSD matching in the search image on the right. We set the detection threshold empirically to the SSD score at the position of maximum similarity so that a match is realized at exactly one pixel.

Table 1 compares the computational cost incurred by our adaptive step size selection approach against several fixed step sizes in terms of the number of SSD computations performed. Running times were approximately proportional to the number of SSD computations performed, indicating that the time spent scanning the sorted contour maps was negligible. In techniques that use a fixed step size greater than one, the target was missed since the search was not exhaustive. In contrast, our adaptive approach detected the object while computing SSD in only 12, 14 and 9% of the search space, in test cases a, b and c, respectively.

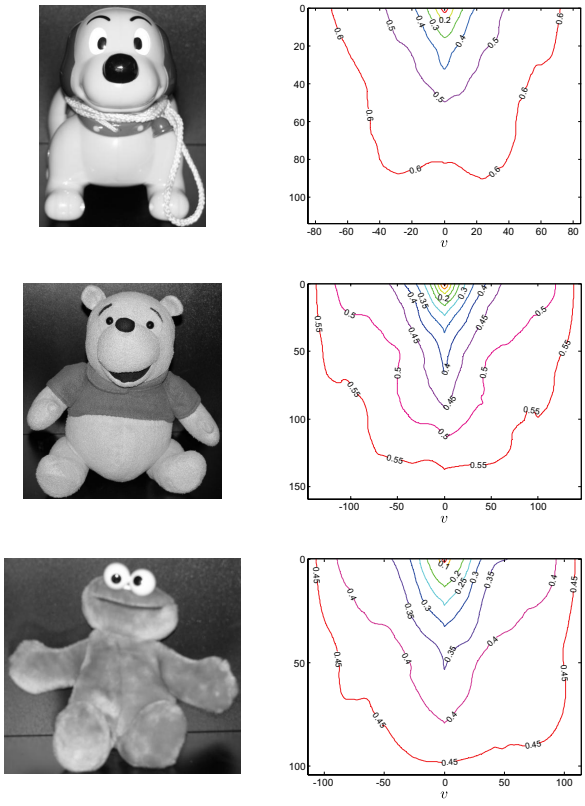
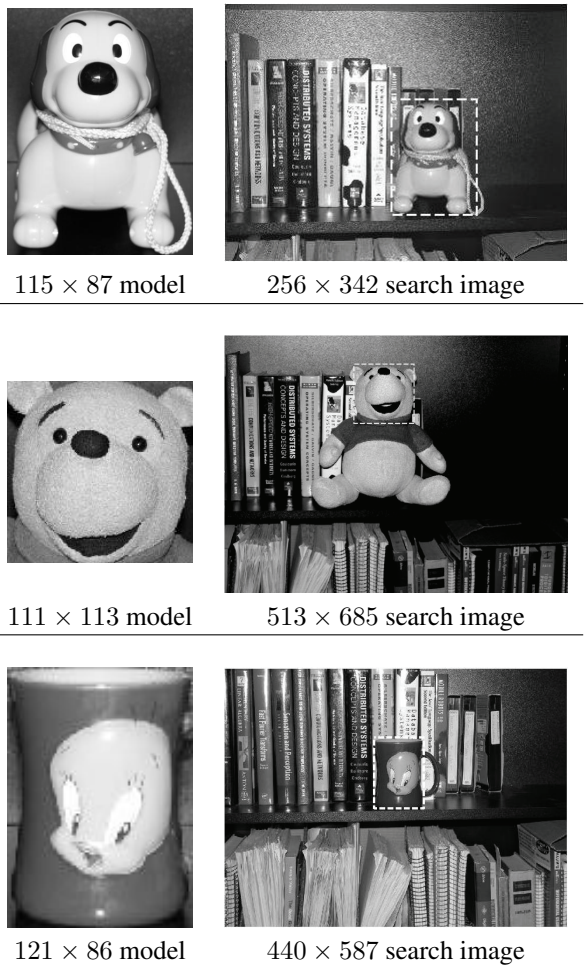


Figure 2. Target images (left) and their $\sup(\Delta_{u,v})$ contour maps computed offline (right).



a 115×87 model 256×342 search image

b 111×113 model 513×685 search image

c 121×86 model 440×587 search image

Figure 3. Three object detection experiments (rows a, b and c). Note that the images are not depicted at their proper relative scale.

5 Summary

In this paper we have demonstrated a principled way for selecting a step size adaptively for SSD-based matching. The adaptive step size is based on a worst-case analysis that guarantees that no missed detections are introduced as a result of not computing SSD at every pixel. Empirical results demonstrate the potential effectiveness of this method. In future work we plan to extend this approach to other distance metrics (e.g., correlation, SAD) and other image representations (e.g., color).

References

[1] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *IJCV*, 2(3):283–310, January 1989.

[2] M. Brown, D. Burschka, and G. Hager. Advances in computational stereo. *PAMI*, 25(8):993–1008, August 2003.

[3] P. Burt, C. Yen, and X. Xu. Local correlation measures for motion analysis: A comparative study. In *PRIP*, pages 269–274, 1982.

[4] G. D. Hager and P. N. Belhumeur. Real-time tracking of image regions with changes in geometry and illumination. In *CVPR*, pages 403–410, 1996.

[5] B. Jahne. *Digital Image Processing: Concepts, Algorithms, and Scientific Applications*. Springer, 1991.

[6] T. Kanade and M. Okutomi. A stereo matching algorithm with an adaptive window: Theory and experiment. *PAMI*, 16(9):920–932, 1994.

[7] A. N. Kolmogorov and S. V. Fomin. *Introductory Real Analysis*. Dover Publications, Inc., New York, 1975. Translated and edited by Richard A. Silverman.

[8] M. Spetsakis and Y. Aloimonos. Optimal computing of structure from motion using point correspondences in two frames. In *ICCV*, pages 449–453, 1988.

[9] P. Viola and M. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, May 2004.

[10] R. Wong and E. Hall. Sequential hierarchical scene matching. *TC*, 27:359–366, 1978.