International Journal on Computer Vision manuscript No.
(will be inserted by the editor)

FAsT-Match: Fast Affine Template Matching

Simon Korman · Daniel Reichman · Gilad Tsur · Shai Avidan

Received: date / Accepted: date

Abstract Fast-Match is a fast algorithm for approximate template matching under 2D affine transformations that minimizes the Sum-of-Absolute-Differences (SAD) error measure. There is a huge number of transformations to consider but we prove that they can be sampled using a density that depends on the smoothness of the image. For each potential transformation, we approximate the SAD error using a sublinear algorithm that randomly examines only a small number of pixels. We further accelerate the algorithm using a branch-and-bound scheme. As images are known to be piecewise smooth, the result is a practical affine template matching algorithm with approximation guarantees, that takes a few seconds to run on a standard machine. We perform several experiments on three different datasets, and report very good results.

Keywords Pattern matching · template matching · image matching · sublinear algorithms

This work was supported by the Israel Science Foundation (grant No. 873/08, in part) and the Ministry of Science and Technology.

Simon Korman · Shai Avidan
School of Electrical Engineering,
Tel Aviv University, Tel Aviv, 69978 Israel
E-mail: simonkor@mail.tau.ac.il · avidan@eng.tau.ac.il

Daniel Reichman
Computer Science department, Cornell University, Ithaca, NY, 14853
Work done while author was at the Weizmann Institute, Israel.
E-mail: daniel.reichman@gmail.com

Gilad Tsur
Yahoo Research Labs, Haifa, 31905 Israel
Work done while author was at the Weizmann Institute, Israel.
E-mail: gilad.tsur@gmail.com

1 Introduction

Image matching is a core computer vision task and template matching is an important sub-class of it. In this paper, we propose an algorithm that matches templates under arbitrary 2D affine transformations. The algorithm is fast and is guaranteed to find a solution that is within an additive error of the global optimum. We name this algorithm: FAsT-Match.

Template matching algorithms usually consider the set of all possible 2D-translations of a template. They differ in the way they discard irrelevant translations (see Ouyang *et al.* [17] for a comprehensive survey of the topic). Template matching under more general conditions, which include also rotation, scale or 2D affine transformation leads to an explosion in the number of potential transformations that must be evaluated.

Fast-Match deals with this explosion by properly discretizing the space of 2D affine transformations. The key observation is that the number of potential transformations that should be evaluated can be bounded based on how smooth the template is. Small variations in the parameters of the transformation will result in small variations in the location of the mapping, and therefore the smoother the template is, the less the Sum-of-Absolute-Difference (SAD) error can change.

Given a desired accuracy level δ we construct a net of transformations such that each transformation (outside the net) has an SAD error which differs by no more than δ from that of some transformation in the net. For each transformation within the net we approximate the SAD error using random sampling. When we take a small δ the net size becomes large and we therefore apply a branch-and-bound approach. We start with a sparse net, discard all transformations in the net whose errors are not within a bound from the best error in the net and then increase the sampling rate around the remaining ones.

It is instructive to contrast Fast-Match with classical direct methods, such as Parametric Optical Flow (OF) [13]. OF methods improved considerably over the years and are the building blocks of many computer vision applications. However, at their core OF are solving a nonlinear optimization problem and as such they rely on an initial guess and might be trapped in a local minimum. Fast-Match, on the other hand, does not rely on an initial guess and is guaranteed to find an approximation to the global optimum.

To overcome the limitations of OF there is a growing focus on feature based methods, such as SIFT [12]. Such methods assume that feature points can be reliably detected and matched in both the image and the template so that there are enough potent matches to estimate a global transformation model, perhaps using RANSAC [4]. Despite the large body of work in this field, the process can fail, especially if there are not enough distinct features in the template or the image. See Figure 1 for illustrations.

OF is clearly less practical when the size of the template is considerably smaller than the size of the image because it does not have a good initial guess. In such cases we can use feature point matching to seed the initial guess of an OF algorithm. However, it is increasingly difficult to detect distinct feature points as the size of the template decreases. Fast-Match does not suffer from this problem.

Fast-Match has some disadvantages when compared to other techniques. When dealing with images where the important information is sparse, e.g. diagrams and text, Fast-Match treats background pixels as if they are as important as foreground pixels, potentially achieving good SAD error at the expense of good localization, in contrast to feature based techniques. In addition, the smoothness of a template determines the complexity-accuracy tradeoff of Fast-Match, while other methods are generally agnostic to this property of the template. In general, while Fast-Match handles a generalized version of (the standard 2D-translation) template matching, it does not deal with a wider range of problems that are addressed using optic-flow or feature-based techniques.

By design, Fast-Match minimizes the SAD error and our experiments validate this, however we also show that minimizing SAD error serves as a proxy to finding the correct *location* of the template and we show results to this effect. Often, even when the size of the template is small, Fast-Match can still find the correct match, whereas feature based methods struggle to detect and match feature points between the template and the image.

We present a number of experiments to validate the proposed algorithm. We run it on a large number of images from the Pascal VOC 2010 data-set [3] to evaluate its performance on templates of different sizes, and in the presence of different levels of degradation (JPEG artifacts, blur, and gaussian noise). We also test Fast-Match on the data sets of Mikolajczyk *et al.* [14,15]. Finally, we report results on real-life image pairs from the ZURICH Buildings data-set [22].

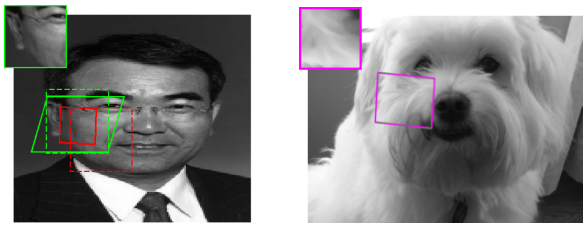


Fig. 1 Shortcomings of current methods: Left: Direct Methods (OF) **require (good) initialization**. They find the correct template location (green parallelogram) given a close enough initialization (dashed green parallelogram), but might fail (converge to solid red parallelogram) with a less accurate initialization (dashed red parallelogram). Right: **Indirect Methods** (feature based) **require (enough) distinct features**. They typically will not detect a single matching feature in such an example. Fast-Match solves both these cases.

2 Background

Our work grew out of the template matching literature which we review next. Since image-matching techniques can be used for template-matching we also include a short reference to this topic, whose full review is beyond the scope of this paper.

Template Matching Evaluating only a subset of the possible transformations was considered in the limited context of Template Matching under 2D translation. Alexe *et al.* [1] derive an upper bound on appearance distance, given the spatial overlap of two windows in an image, and use it to bound the distances of many window pairs between two images. Pele and Werman [18] ask "How much can you slide?" and devise a new rank measure that determines if one can slide the test window by more than one pixel.

Extending Template Matching to work with more general transformations was also considered in the past. Fuh *et al.* [6] proposed an affine image model for motion estimation, between images which have undergone a mild affine deformation. They exhaustively search a range of the affine space (practically - a very limited one, with only uniform scale). Fredriksson [5] used string matching techniques to handle also rotation. Kim and Araújo [8] proposed a grayscale template matching algorithm that considers also rotation and scale. Yao and Chen [27] propose a method for the retrieval of color textures, which considers also variations in scale and rotation. Finally, Tsai and Chiang [24] developed a template matching method that considers also rotation, which is based on wavelet decompositions and ring projections. The latter three methods do not provide guarantees regarding the approximation quality of the matching.

Rucklidge [20] proposed a branch-and-bound search for a gray-level pattern under affine transformations. His scheme is based on calculating worst-case ranges of pixel intensities in rectangles of the target image, which are in turn used to impose lower bounds on the improvement of match scores as a result of sub-divisions in transformation space. Another related work is that of Tian and Narasimhan [23], that estimates non-rigid distortion parameters of an image relative to a template under a wide variety of deformation types. Their method also employs an efficient search in parameter space, providing bounds on the distance of the discovered transformation, in parameter space, from the underlying deformation. Our method, is contrast, seeks a transformation that minimizes the distance in image (appearance) space. This allows us to provide provable guarantees even when the true deformation is not in the transformations space (e.g. in the existence of image noise or other geometric or photometric changes) or when the pattern appears repetitively in the image. We also explicitly show how distances in parameter space and distances in image space are related through the smoothness of the template.

Image Matching Methods Image matching algorithms are often divided into direct and feature-based methods.

In direct methods, such as Lukas-Kanade [13], a parametric Optic-Flow mapping is sought between two images so as to minimize the Sum-of-Squared-Difference (SSD) between the images. See the excellent review by Baker *et al.* [2] on optic flow image alignment algorithms. Such iterative methods do not generally guaranty global convergence and may discover local minima, unless provided with good initializations which are not always known in advance. One exception is the 'Filter-Flow' work of Baker and Seitz [21], which gives an algorithm that can find globally optimal solutions for a broad range of transformations. However, it is done by solving a very large linear program, with impracticable runtime and memory requirements, even for the setting of limited scale and motion over low-res images.

Alternatively, one can use feature-based methods such as SIFT [12], or its variant ASIFT [16] which is designed to be fully affine invariant. In this scenario, interest points are detected independently in each image and elaborate image descriptors are used to represent each such point. Given enough corresponding feature points it is possible to compute the global affine transformation between the images. This approach relies on the assumption that the same interest points can be detected in each image independently and that the image descriptors are invariant to 2D affine transformations so that they can be matched across the images.

Other related work Our work is also inspired by techniques from the field of *sublinear* algorithms, which are extremely fast (typically approximation) algorithms. They are generally randomized and access only a subset of their input. The runtime of such algorithms is sublinear in the input size, and generally depends on some given accuracy parameters. The use of sublinear algorithms in the field of computer vision was advocated by Rashkodnikova [19] and later followed by Kleiner *et al.* [9] and Tsur and Ron [25].

3 The Main Algorithm

3.1 Preliminaries

We are given two grayscale images I_1 and I_2 of dimensions $n_1 \times n_1$ and $n_2 \times n_2$ respectively, with pixel values in the range $[0,1]$.[1] We will refer to I_1 as the *template* and to I_2 as the *image*. The *total variation* of an image I, denoted by $\mathcal{V}(I)$, is the sum over the entire image of the maximal intensity difference between each pixel p and any of its eight neighbors $q \in N(p)$ (we omit the dependence on I as it is always clear from the context). That is,

$$\mathcal{V} = \sum_{p \in I} \max_{q \in N(p)} |I(p) - I(q)| \tag{1}$$

We generally consider rigid geometric transformations T that map pixels p in I_1 to pixels in I_2. Specifically, we deal with the set $\mathcal{A}$ of 2D-affine transformations of the plane that have scaling factors in the range $[1/c, c]$ for a fixed positive constant c. Any transformation $T \in \mathcal{A}$ can be seen as multiplying the pixel location vector by a 2×2 non-singular matrix and adding a "translation" vector, finally rounding the resulting numbers. Such a transformation can be parameterized by six degrees of freedom.

For images I_1 and I_2 and transformation T, we define the error $\Delta_T(I_1, I_2)$ to be the (normalized) Sum-of-Absolute-Differences (SAD) between I_1 and I_2 with respect to T. More formally:

$$\Delta_T(I_1, I_2) = \frac{1}{{n_1}^2} \sum_{p \in I_1} |I_1(p) - I_2(T(p))| \tag{2}$$

Note that this error is in the interval $[0,1]$, as this is the range of pixels intensity values. If a pixel p is mapped out of the area of I_2 then the term $|I_1(p) - I_2(T(p))|$ is taken to be 1. We wish to find a transformation T that comes close to minimizing $\Delta_T(I_1, I_2)$. The minimum over all affine transformations T of $\Delta_T(I_1, I_2)$ is denoted by $\Delta(I_1, I_2)$.

A crucial component of our algorithm is the *net* of transformations. This net is composed of a small set of transformations, such that any affine transformation is "close" to some transformation in the net. To this end we define the ℓ_∞ distance between any two transformations T and T' that

[1] The algorithm is not restricted to square images but we discuss these for simplicity throughout the article

Algorithm 1 ***Approximating the Best Transformation***

Input: Grayscale images I_1, I_2; a precision parameter δ;

Output: A transformation T

1. Create a net $\mathcal{N}_{\delta/2}$ that is a $\frac{\delta \cdot n_1^2}{\mathcal{V}}$-cover of the set of affine transformations
2. For each $T \in \mathcal{N}_{\delta/2}$ approximate $\Delta_T(I_1, I_2)$ to within precision of δ (section 3.4). Denote the result by d_T.
3. Return the transformation T with the minimal value d_T

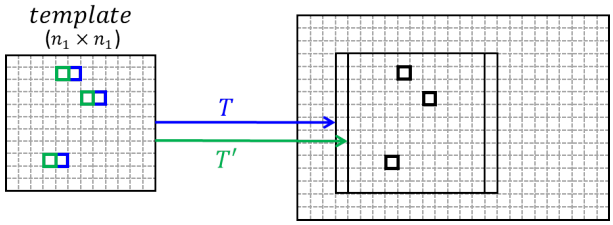


Fig. 2 Intuition for Theorem 1. Transformations that have an ℓ_∞ distance of 1 from each other map neighboring pixels from the template to the same pixel in the image. Thus, the change in error (when changing from one transformation to the other) is bounded by the total variation of the template.

quantifies how far the mapping of any point p in I_1 according to T may be from its mapping by T'. Formally,

$$\ell_\infty(T, T') = \max_{p \in I_1} \|T(p) - T'(p)\|_2 \tag{3}$$

where the $\|\cdot\|_2$ is the Euclidean distance in the target image plane. Note that this definition does not depend on the pixel values of the images, but only on the mappings T and T', and on the dimension n_1 of the source image I_1. The key observation is that we can bound the difference between $\Delta_T(I_1, I_2)$ and $\Delta_{T'}(I_1, I_2)$ in terms of $\ell_\infty(T, T')$ as well as the total variation $\mathcal{V}$ of I_1. This will enable us to consider only a limited set of transformations, rather than the complete set of affine transformations.

For a positive α, a net of (affine) transformations $\mathcal{T} = \{T_i\}_{i=1}^{l}$ is an α-cover of $\mathcal{A}$ if for every transformation $T \in \mathcal{A}$, there exists some T_j in $\mathcal{T}$, such that $\ell_\infty(T, T_j') = O(\alpha)$. In our algorithm, we use a net, which we denote by $\mathcal{N}_\delta$, with a very particular choice of the density parameter α of the cover: $\alpha = \frac{\delta \cdot n_1^2}{\mathcal{V}}$, where $\delta \in (0, 1]$ is the accuracy parameter of the algorithm and n_1 and $\mathcal{V}$ are the dimension and total-variation of the image I_1. Note that number of transformations in the net will grow as a function of both $1/\delta$ and $\mathcal{V}$.

3.2 Algorithm Details

We describe a fast randomized algorithm that returns, with high probability, a transformation T such that $\Delta_T(I_1, I_2)$ is close to $\Delta(I_1, I_2)$. The algorithm, outlined in Algorithm 1, basically enumerates the transformations in a net $\mathcal{N}_\delta$ and finds the one with the lowest error. In Step 2 of the algorithm we use a sublinear method for the approximation of $\Delta_T(I_1, I_2)$ (instead of computing it exactly), which is presented in subsection 3.4.

The rest of this section is dedicated to establishing guarantees on the algorithm's quality of approximation. We wish to bound the difference between the quality of the algorithm's result and that of the optimal transformation (i.e. one which attains the optimal error $\Delta(I_1, I_2)$) in terms of two parameters - the template total variation $\mathcal{V}$ and the precision parameter δ.

The following theorem, which is our main theoretical contribution, formulates a relation which will enable us to bound the degradation in approximation that occurs as a result of sampling the space of transformations rather then enumerating it exhaustively. More specifically, it bounds the difference between $\Delta_{T'}(I_1, I_2)$ and $\Delta_T(I_1, I_2)$ for a general affine transformation T' and its nearest transformation T on the sampling net.

Theorem 1 Let I_1, I_2 be images with dimensions n_1 and n_2 and let δ be a constant in $(0, 1]$. For a transformation T', let T be the closest transformation to T' in the net $\mathcal{N}_\delta$ (which is a $\frac{\delta \cdot n_1^2}{\mathcal{V}}$-cover). It holds that: $|\Delta_{T'}(I_1, I_2) - \Delta_T(I_1, I_2)| \leq O(\delta)$.

The full proof of Theorem 1 can be found in the Appendix. To get an intuition of why it holds, consider a degenerate case of horizontal translations, which is illustrated in Figure 2. Let T be a translation of k pixels and let T' be a translation of $k+1$. Now consider the value of $|\Delta_{T'}(I_1, I_2) - \Delta_T(I_1, I_2)|$. Every pixel $p = (x, y)$ in I_1 is mapped by T to the same location that the pixel $p' = (x-1, y)$ is mapped to by T'. Thus the difference between $\Delta_{T'}(I_1, I_2)$ and $\Delta_T(I_1, I_2)$ is bounded by the total sum of differences between horizontally neighboring pixels in I_1. The sum of these differences is related linearly to the total variation of I_1. Likewise, in the case that one of the translations is by k pixels and the other is by $k + \delta n_1$ pixels - the change in the SAD is bounded by the total variation multiplied by δn_1. After normalizing by the size of I_1 we get the bound stated in the theorem.

In Section 3.3 we provide a construction of a net $\mathcal{N}_\delta$, which is a $\frac{\delta \cdot n_1^2}{\mathcal{V}}$-cover of the space $\mathcal{A}$ of affine transformations, and whose size is $\Theta\big((\frac{n_2}{n_1})^2 (\frac{\mathcal{V}}{n_1})^6 \frac{1}{\delta^6}\big)$. The correctness of Theorem 1, along with the existence of such a net and the fact that each δ-approximation of Δ_T (step 2 of Algo-

rithm 1) takes $\tilde{\Theta}(1/\delta^2)$ [2], lead directly to the following result on the accuracy and complexity of Algorithm 1:

Theorem 2 Algorithm 1 returns a transformation T such that $|\Delta_T(I_1, I_2) - \Delta(I_1, I_2)| \leq O(\delta)$ holds with high probability. Its total runtime (and number of queries) is $\Theta\big((\frac{n_2}{n_1})^2 \cdot (\frac{\mathcal{V}}{n_1})^6 \cdot \frac{1}{\delta^8}\big)$.

Interestingly, the fact that Algorithm 1's complexity depends on the total variation of the template I_1 is not an artifact of the analysis of the algorithm. In a theoretical analysis of the query complexity of such problems [10] we prove a lower bound that demonstrates how the number of pixels that need to be examined by any algorithm (and hence its runtime) grows with the total variation of the template.

3.3 Construction of the Net $\mathcal{N}_\delta$ (a $\frac{\delta \cdot n_1^2}{\mathcal{V}}$-Cover)

Once the density parameter of the net $\alpha = \frac{\delta \cdot n_1^2}{\mathcal{V}}$ has been selected, an appropriate α-cover of the space of affine transformations can be constructed. As a reminder, we consider the set of affine transformations from an image I_1 of dimension $n_1 \times n_1$ to an image I_2 of dimension $n_2 \times n_2$, which we denote by by $\mathcal{A}$. The cover will be a product of several 1-dimensional grids of transformations, each covering one of the constituting components of a standard decomposition of Affine transformations [7], which is given in the following claim.

Claim 1 Every orientation-preserving affine transformation matrix A can be decomposed into $A = TrR_2SR_1$, where Tr, R_i, S are translation, rotation and non-uniform scaling matrices [3].

We now describe a 6-dimensional grid, $\mathcal{N}_\delta$, which we will soon prove to be a $\frac{\delta \cdot n_1^2}{\mathcal{V}}$-cover of $\mathcal{A}$. The basic idea is to discretize the space of Affine transformations, by dividing each of the dimensions into $\Theta(\delta)$ equal segments. According to claim 1, every affine transformation can be composed of a rotation, scale, rotation and translation. These basic transformations have 1, 2, 1 and 2 degrees of freedom, respectively. These are: a rotation angle, x and y scales, another rotation angle and x and y translations.

The idea will be to divide each dimension into steps, such that for any two consecutive transformations T and T' on any of the dimensions it will hold that:

$$\ell_\infty(T, T') < \Theta\Big(\frac{\delta \cdot n_1^2}{\mathcal{V}}\Big) \tag{4}$$

Starting with translations (x and y), since the template should be placed within the bounds of the image I_2, we consider the range $[-n_2, n_2]$. Taking step sizes of $\Theta(\delta n_1^2/\mathcal{V})$, guarantees by definition that Equation (4) holds. Similarly, for rotations we consider the full range of $[0, 2\pi]$, and use steps of size $\Theta(\delta n_1/\mathcal{V})$. This suffices since rotating the template I_1 by an angle of $\delta n_1/\mathcal{V}$ results in pixel movement which is limited by an arc-length of $\Theta(\delta n_1^2/\mathcal{V})$. Finally, since the scales are limited to the interval $[\frac{1}{c}, c]$ and since I_1 is of dimension n_1, steps in the scale axes of size $\Theta(\delta n_1/\mathcal{V})$ will cause a maximal pixel movement of $\Theta(\delta n_1^2/\mathcal{V})$ pixels.

The final cover N_δ, of size $\Theta\big((\frac{n_2}{n_1})^2 \cdot (\frac{\mathcal{V}}{\delta n_1})^6\big)$, is simply a Cartesian product of the 1-dimensional grids whose details are summarized in the following table.

transformation	step size	range	num. steps
x translation	$\Theta(\delta n_1^2/\mathcal{V})$ pixels	$[-n_2, n_2]$	$\Theta(\frac{n_2}{n_1} \cdot \mathcal{V}/\delta n_1)$
y translation	$\Theta(\delta n_1^2/\mathcal{V})$ pixels	$[-n_2, n_2]$	$\Theta(\frac{n_2}{n_1} \cdot \mathcal{V}/\delta n_1)$
1st rotation	$\Theta(\delta n_1/\mathcal{V})$ radians	$[0, 2\pi]$	$\Theta(\mathcal{V}/\delta n_1)$
2nd rotation	$\Theta(\delta n_1/\mathcal{V})$ radians	$[0, 2\pi]$	$\Theta(\mathcal{V}/\delta n_1)$
x scale	$\Theta(\delta n_1/\mathcal{V})$ pixels	$[1/c, c]$	$\Theta(\mathcal{V}/\delta n_1)$
y scale	$\Theta(\delta n_1/\mathcal{V})$ pixels	$[1/c, c]$	$\Theta(\mathcal{V}/\delta n_1)$

The final result is formulated in the following claim, where the proof follows directly from the above construction: Given the net $\mathcal{N}_\delta$ and an arbitrary affine transformation A in $\mathcal{A}$, there exists a transformation A' in $\mathcal{N}_\delta$, such that A and A' differ by at most $\Theta(\frac{\delta \cdot n_1^2}{\mathcal{V}})$ (in the sense of the distance ℓ_∞) in each of the 6 constituting dimensions. Now, taking an arbitrary pixel p in I_1 and applying either A or A' on it, the results may not differ by more than $\Theta(\frac{\delta \cdot n_1^2}{\mathcal{V}})$ pixels, and this can be shown by a sequential triangle-inequality argument on each dimension.

Claim 2 The net $\mathcal{N}_\delta$ is a $\frac{\delta \cdot n_1^2}{\mathcal{V}}$-cover of $\mathcal{A}$ of size $\Theta\big((\frac{n_2}{n_1})^2 \cdot (\frac{\mathcal{V}}{\delta n_1})^6\big)$.

3.4 Approximating the Distance $d_T(I_1, I_2)$

We now turn to describe a sublinear algorithm which we use in Step 2 of Algorithm 1 to approximate $\Delta_T(I_1, I_2)$. This dramatically reduces the runtime of Algorithm 1 while having a negligible effect on the accuracy. The idea is to estimate the distance by inspecting only a small fraction of pixels from the images. The number of sampled pixels depends on an accuracy parameter ϵ and not on the image sizes. Algorithm 2 summarizes this procedure, whose guarantees are specified in the following claim.

Claim 3 *Given images I_1 and I_2 and an affine transformation T, Algorithm 2 returns a value d_T such that $|d_T - \Delta_T(I_1, I_2)| \leq \epsilon$ with probability $2/3$. It performs $\Theta(1/\epsilon^2)$ samples.*

[2] The symbol $\tilde{\Theta}$ hides (low order) logarithmic factors

[3] arguments are similar for orientation-reversing transformations (which include reflection)

Algorithm 2 ***Single Transformation Evaluation***

Input: Grayscale images I_1 and I_2; a precision parameter ϵ; and a transformation T;

Output: An estimate of the distance $\Delta_T(I_1, I_2)$

- Sample $m = \Theta(1/\epsilon^2)$ values of pixels $p_1 \dots p_m \in I_1$.
- Return $d_T = \sum_{i=1}^{m} |I_1(p_i) - I_2(T(p_i))|/m$.

The claim holds using an additive Chernoff bound. Note that to get the desired approximation with probability $1 - \eta$ we perform $\Theta(\log(1/\eta)/\epsilon^2)$ samples.

Photometric Invariance An adaptation of Algorithm 2 allows us to deal with linear photometric changes (adjusting brightness and contrast). We calculate the optimal change for the points sampled every time we run *Single Transformation Evaluation* by normalizing each sample by its mean and standard-deviation. This adjustment allows us to deal with real life images at the cost of little additional time.

4 The Branch-and-Bound Scheme

To achieve an additive approximation of $O(\delta)$ in Algorithm 1 we must test the complete net of transformations $\mathcal{N}_\delta$. Achieving a satisfactory error rate would require using a net $\mathcal{N}_\delta$ where δ is small. The rapid growth of the net size with the reduction in the value of δ (linear in $1/\delta^6$) renders our algorithm impractical, despite the fact that our testing of each transformation is extremely efficient. To overcome this difficulty, we devise a branch-and-bound scheme, using nets of increasing resolution while testing small fractions of the transformations in the rapidly growing nets. This improvement is possible with virtually no loss in precision, based on our theoretical results. As a result, the number of transformations we test in order to achieve a certain precision is reduced dramatically.

Algorithm 3 ***Fast-Match:* a Branch-and-Bound Algorithm**

Input: Grayscale images I_1, I_2, a precision parameter δ^*
Output: A transformation T.

1. Let S^0 be the complete set of transformations in the net $\mathcal{N}_{\delta_0}$ (for initial precision δ_0)
2. Let $i = 0$ and repeat while $\delta_i > \delta^*$
   (a) Run Algorithm 1 with precision δ_i, but considering only the subset S^i of $\mathcal{N}_{\delta_i}$
   (b) Let T_i^{Best} be the best transformation found in S^i
   (c) Let $Q^i = \{q \in S^i \ : \ \Delta_q(I_1, I_2) - \Delta_{T_i^{Best}}(I_1, I_2) < L(\delta_i)\}$
   (d) Improve precision: $\delta_{i+1} = fact \cdot \delta_i$ (by some constant factor $0 < fact < 1$)
   (e) Let $S^{i+1} = \{T \in Net_{\delta_{i+1}} \ : \ \exists q \in Q_i \ \ s.t. \ \ell_\infty(T, q) < \delta_{i+1} \cdot n_1^2/\mathcal{V}\}$
3. Return the transformation T_i^{Best}

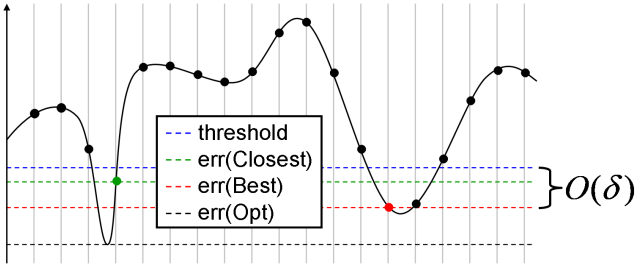


Fig. 3 Branch-and-Bound Analysis. One stage of the branch-and-bound scheme. For simplicity the space of transformations is in 1D (x-axis) against the SAD-error (y-axis). Vertical gray lines are the sampling intervals of the net. Dots are the samples. Horizontal dotted lines are SAD errors of: Black (Optimal transformation, which is generally off the net), Red (best transformation found on the net), Green (closest-to-Optimal transformation on the net) and Blue (threshold). Only areas below the (blue) threshold are considered in the next stage. The choice of the threshold is explained in the text.

We describe next the branch-and-bound scheme for which the pseudo-code appears below as Algorithm 3 (Fast-Match). In each stage, Algorithm 1 is run on a subset S of the net $\mathcal{N}_\delta$. Figure 3 gives an illustration of transformations examined by the algorithm and their errors (in particular *Opt* - the optimal, *Best* - the best examined, and *Closest* - the closest on the net to *opt*). We denote by $e(Opt)$ the error of *opt* and similarly for *best* and *closest*.

We wish to rule out a large portion of the transformation space before proceeding to the next finer resolution net, where the main concern is that the optimal transformation should not be ruled out. Had we known $e(Closest)$, we could have used it as a threshold, ruling out all transformations with error exceeding it. We therefore estimate $e(Closest)$ based on the relations between $e(Opt)$, $e(Best)$ and $e(Closest)$. On one hand, $e(Best) - e(Opt) = O(\delta)$ (following Theorem 2) and on the other hand, $e(Closest) - e(Opt) = O(\delta)$ (by the construction of the net and following Theorem 1). It follows that $e(Closest) - e(Best) = O(\delta)$ hence $e(Closest) < e(Best) + O(\delta)$. Using a large set of data, we estimated constants c_1 and c_2, such that $e(Closest) < e(Best) + c_1 \cdot \delta + c_2)$ holds for 97% of the test samples. This learned function $L(\delta) = c_1 \cdot \delta + c_2$ is used in Step 2c of Algorithm 3, for the choice of the points that are not ruled out, for each net resolution. In specific cases where the template occurs in much of the image (e.g. flat blue sky patch), we limit the size of Q_i so that the expanded S_{i+1} will fit into RAM.

Fig. 4 Example from a Fast-Match Run Left: The template (shown enlarged for clarity), with the 152 pixels that Fast-Match samples. Right: Target image, with origin of the template (ground truth location) in green and a candidate area it is mapped to in magenta.

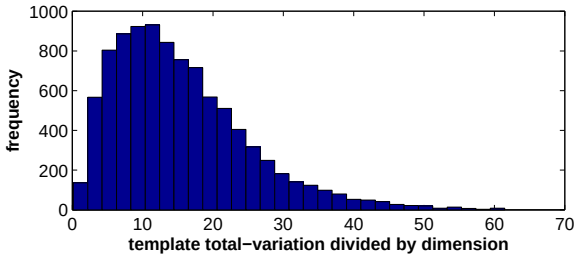


Fig. 5 Normalized total variation of 9500 random templates: We measured the total variation $\mathcal{V}$ of $9,500$ random square templates of varying edge dimension n_1. The total-variation values are arranged in a 30-bin histogram after dividing each by the specific template dimension n_1. The fact that natural image templates are typically smooth results in a distribution of values which allows us to empirically upper bound the total-variation $\mathcal{V}$ by a small constant times the template dimension n_1. See text for further details.

5 Experiments

In this section we present three experiments that evaluate the performance of our algorithm under varying conditions and setups, using several data-sets. In the first such experiment (Section 5.1) each template is extracted from an image and matched back to it. In the second (Section 5.2), the template is extracted from one image and matched to another, that is related to it geometrically by a homography. In the third experiment (Section 5.3) the template is taken from one image of a scene and is mapped to an entirely different image of the same scene.[4]

Implementation Details Fast-Match, when evaluating a transformation, estimates the SAD error between the template and the image region in the target location. This is consistent with the general approach of template matching schemes, minimizing a photometric measure (e.g. SAD, SSD) between the matching subimages. This approach allows the analysis of approximation quality, however it may not be informative in practice. A location-dependent measure of the correct mapping (which guarantees with respect to it could not be provided, due to ambiguity) is the *overlap error*, which quantifies the overlap between the 'correct' location and mapped location of the template in the image (see for example the green and magenta quadrilaterals in Figure 4). The overlap error is defined (following, e.g., Mikoalyciz et al [14,15]) to be: 1 minus the ratio between the intersection and union areas of the regions. In addition to validating our performance with respect to the SAD error, we use the overlap error to evaluate performance in the first two experiments (where 'ground truth' location is available).

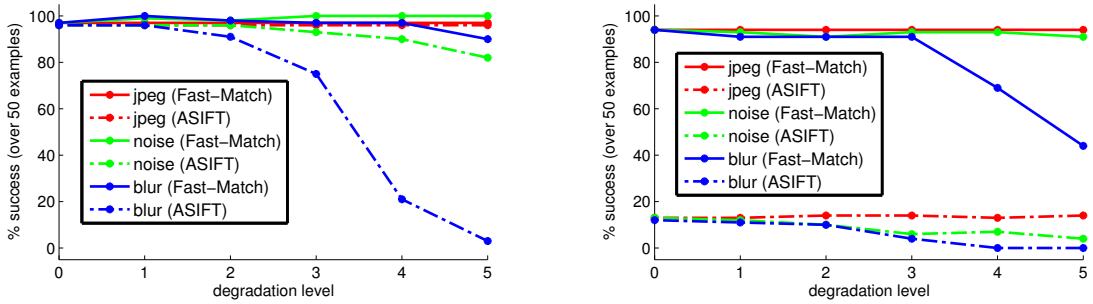
Achieving an accuracy of $O(\delta)$ was based on using a net $\mathcal{N}_\delta$ which was a $\frac{\delta \cdot n_1^2}{\mathcal{V}}$-cover of transformation space $\mathcal{A}$. The size of the required net is template-specific, depending not only on the precision δ but also on the template dimension n_1 and its total-variation $\mathcal{V}$. Recall that $\mathcal{V}$ can, theoretically, take any value in the range $[0, n_1^2]$. Nevertheless, we can take advantage of the piecewise smoothness of natural images and specifically use the fact that natural images are known to have small total variation (see, e.g., [26]).

We provide here a simple experiment, using the large collection of natural images of the Pascal VOC 2010 data-set [3]. We repeatedly extracted 'random' templates from these images using the following steps: (i) choose an image, uniformly at random; (ii) pick a random dimension n_1 for a square template to be extracted (between 10% and 90% of the smaller image dimension) and a random center location; (iii) Measure the total variation $\mathcal{V}$ of the extracted template and normalize it by its own dimension n_1. We then organized the 9500 values of $\mathcal{V}/n_1$ in a histogram of 30 bins, as can be seen in Figure 5. When examining the distribution of the per-template normalized values $\mathcal{V}/n_1$ - the high concentration around very low values (considering the mean $E[n_1] \approx 200$ and the standard deviation $\sigma(n_1) \approx 100$ of the dimensions n_1 over the 9500 random templates) suggests that taking $\mathcal{V}$ to be a relatively small constant times n_1 will suffice for a vast majority of templates. In our implementation, we take such an 'average-case' approach regarding the total-variation, allowing the net sizes to depend only on the template dimension n_1 and precision δ, resulting in more average, predictable, memory and runtime complexity.

5.1 Affine Template Matching

In this large scale experiment, we follow the methodology used in the extensive pattern matching performance evaluation of Ouyang *et al.* [17]. We use images from the Pascal VOC 2010 data-set [3], which has been widely used for the evaluation of a variety of computer vision tasks. Each pattern matching instance involves selecting an image at random from the data-set and selecting a random affine transformation, which maps a square template into the image (the mapped square being a random parallelogram). The parallelogram within the image is then warped (by the inverse

[4] source-code and extended results are available at [11]



(a) template dimension of 50% (b) template dimension of 20%

Fig. 6 Performance under different template sizes and image degradations. Analysis is presented for two different template dimensions: (a) 50% and (b) 20% of image dimension. In each, the **x-axis** stands for the increasing levels of image degradation, ranging from 0 (no degradation) to 5 (highest). The **y-axis** stands for the success rates of Fast-Match and ASIFT. Fast-Match is capable of handling smaller and smaller template sizes, while the feature based method ASIFT, deteriorates significantly as template dimension decreases. Like ASIFT, Fast-Match is fairly robust to the different image degradations and is even more robust to high levels of image blur than ASIFT ($\sigma = 4/7/11$ pixels). See text for details.

affine transformation) in order to create the square template. See Figure 4 for an example.

We test the method on different template sizes, where the square template dimensions are between 10% and 90% of the minimum image dimension. For each such size, we create 200 template matching instances, as described above. In the table in Figure 7 we report SAD and overlap errors of Fast-Match for the different template sizes. Fast-Match achieves low SAD errors, which are extremely close to those of the ground-truth mapping. The ground-truth errors are at an average of 4 graylevels (not zero), since interpolation was involved in the creation of the template. As can be seen, Fast-Match does well also in terms of overlap error. In the following experiments, we measure success only in terms of overlap error.

Template Dimension	90%	70%	50%	30%	10%
avg. Fast-Match SAD err.	5.5	4.8	4.4	4.3	4.8
avg. ground truth SAD err.	4.1	4.1	4.0	4.4	6.1
avg. Fast-Match overlap err.	3.2%	3.3%	4.2%	5.3%	13.8%

Fig. 7 Fast-Match Evaluation: SAD and Overlap errors. SAD errors are in graylevels (in [0,255]). Low SAD error rates are achieved across different template dimensions (10% to 90% of the image dimension). Fast-Match guarantees finding an area with similar *appearance*, and this similarity translates to a good overlap error, correctly localizing the template in the image. Fast-Match SAD error is comparable to that of the ground truth. See text for details.

Comparison to a feature based approach After we have evaluated Fast-Match on the Pascal dataset, we now show that its performance is comparable to template matching that can be achieved through a feature based approach. To this end, we compare its performance to that of ASIFT [16] - a state of the art method which is a fully affine invariant extension of SIFT [12], for extracting feature point correspondences between pairs of related images. As a disclaimer, we note that feature-based methods, such as SIFT, were designed (and are used) for handling a wide range of tasks, which our method does not attempt to solve. Nevertheless, they are most commonly used in practice for finding a rigid geometric transformation between a pair of images, e.g. in the process of stitching a panorama.

We examine Fast-Match's performance under 3 types of image degradations: additive white gaussian noise, image blur and JPEG distortion. We show its performance under varying template sizes at different levels of such degradations. Since ASIFT (without additional post-processing for transformation recovering) and Fast-Match cannot be directly compared due to their different output types[5], we define for ASIFT a success criterion which is the minimal requirement for further processing: Namely, it is required to return at least 3 correspondences, which are fairly close to being exact - the distance in the target image between the corresponded point and the true corresponding point must be less than 20% of the dimension of the template. The success criterion for Fast-Match is an overlap error of less than 20%. This is an extremely strict criterion, especially for templates mapped to small areas - See a variety of examples, below and above this criterion, in [11]. As is claimed in Mikolajczyk *et al.* [15], an overlap error of 20% is very small since regions with up to 50% overlap error can still be matched successfully using robust descriptors.

We consider 2 different template dimensions which are 50% and 20% of the minimal dimension of the image. For each such size, we repeat the template matching process described above. We consider 6 degradation levels of each type (applied to the target image), as follows: Image blurring with

[5] Unlike our method, such feature based methods do not directly produce a geometric mapping. These can be found, based on good quality sets of matching points, using robust methods such as RANSAC [4] by assuming a known geometric model that relates the images (e.g. affine).

Template Dimension	90%	70%	50%	30%	10%
ASIFT	12.2 s.	9.9 s.	8.1 s.	7.1 s.	NA
Fast-Match	2.5 s.	2.4 s.	2.8 s.	6.4 s.	25.2 s.

Fig. 8 Runtimes on different template sizes: Average runtimes (in seconds) over 100 instances for each template dimension. Fast-Match is much faster in general. As opposed to ASIFT, Fast-Match's runtime increases with the decrease of template dimension. The reason is twofold: 1] The size of our net grows linearly in the image-area/template-area ratio. 2] Smaller templates are more common in the image and hence the Branch-And-Bound enhancement becomes less effective.

gaussian kernels with STD of $\{0,1,2,4,7,11\}$ pixels, additive gaussian noise with STDs of $\{0,5,10,18,28,41\}$ greylevels and finally - JPEG compression with quality parameter (in Matlab) set to $\{75,40,20,10,5,2\}$[6].

The comparison of the above success rates of ASIFT and Fast-Match is presented in Figure 6. This experiment validates our claim that unlike feature-based methods (e.g. ASIFT) our method can handle smaller and smaller templates (10% in each image dimension - which translate to 30x30 templates). In addition, Fast-Match is fairly robust with respect to noise and JPEG compression and even more robust to blur in comparison with ASIFT [7]. The table in Figure 8 shows the algorithm's average runtimes for several templates sizes, run on a single cpu of an Intel i7 2.7 MHz processor.

5.2 Varying Conditions and Scene Types

In the second experiment we examine the performance of Fast-Match under various imaging conditions and on different scene types. We test our algorithm on a dataset by Mikoalyciz et al [14,15], originally used to evaluate the performance of interest-point detectors and descriptors. The data set is composed of 8 sequences of images, 6 images each: Blur (2), a combination of rotation and zooming (2), viewpoint change (2), JPEG compression (1) and light conditions (1). In each sequence the degradation increases, e.g., in a blur sequence, from entirely unblurred extremely blurred. Unlike the first experiment, here the template is taken from one image and searched for in a different one, related by a homography (rather than an affinity), increasing the difficulty of the task.

Each experiment is conducted as follows: We first choose a random axis-aligned rectangle in the first image, where the edge sizes are random values between 10% and 50% of the respective image dimensions. We then use Fast-Match to map this template to each of the other 5 images in the series. We perform 50 such experiments for which the success rates are given in the table in Figure 9. The success criterion is identical to the first experiment (i.e. overlap error $< 20\%$) [8]. The sequences of images (with an example of a single experiment for each) are shown in Figure 10.

We achieve high success rates across the dataset, with the exception of the higher degradation levels of the 'Wall' and 'Boat' sequences. Note that, the smaller the template

[6] Note that in the 3 distortion types, the lowest degradation level is equivalent to no degradation at all

[7] ASIFT is based on SIFT, which has been shown in [14] to be prominent in its resilience to image blur, with respect to other descriptors.

Seq. \ Distortion Level	1	2	3	4	5
Zoom + Rotation (Bark)	100%	100%	87.5%	97.5%	87.5%
Blur (Bikes)	100%	100%	100%	100%	100%
Zoom + Rotation (Boat)	100%	100%	75%	87.5%	55%
Viewpoint change (Graffiti)	95%	95%	87.5%	90%	85%
Brightness change (Light)	97.5%	100%	100%	100%	97.5%
Blur (Trees)	100%	100%	100%	97.5%	100%
JPEG compression (UBC)	97.5%	100%	100%	100%	100%
Viewpoint change (Wall)	100%	100%	100%	5%	0%

Fig. 9 Percent of successful matches (overlap error$<$ 20%) per sequence and degradation level. Several examples appear in Fig. 10.

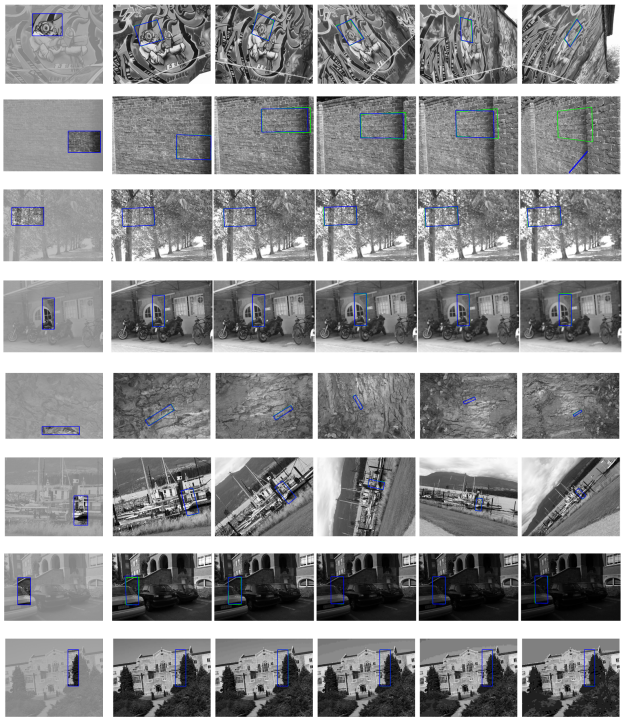


Fig. 10 A typical experiment for each of the Mikolajczyk [15] sequences. In the leftmost image - the area marked in blue is the input given to Fast-Match. In each of the remaining images a blue parallelogram indicates the mapping produced by Fast-Match, while a green quadrilateral marks the ground truth.

[8] Note that because we are approximating a projective transformation using an affine one (which means matching a general quadrilateral using a parallelogram), the optimal overlap error may be far greater than 0.

Fig. 11 Zurich Dataset [22] - Good Examples: In the blue rectangle on the left of each pair of images is the template presented to Fast-Match. In the blue parallelogram on the right is the region matched by the algorithm. Note that also for some of the non-affine mappings Fast-Match gives a good result.

area in the target image, the more demanding the overlap error criterion becomes[9]. This is relevant especially to the zoom sequences. The 'Wall' images are uniform in appearance and this makes it difficult to translate good SAD error to correct localization. The results of Experiment II can not be compared with those of [14] as they do not deal directly with template or image matching. In this experiment too, Fast-Match deals well with photometric changes as well as the blur and JPEG artifacts.

5.3 Matching in Real-World Scenes

In the third experiment, we present the algorithm's performance in matching regions across different view-points of real-world scenes. We use pairs of images from the Zurich buildings dataset [22]. As done in the second experiment, we choose a random axis-aligned rectangle in the first image, where the edge sizes are random values between 10% and 50% of the respective image dimensions. This dataset is more challenging for the performance of the algorithm, as well as for experimentation: The template typically includes several planes (which do not map to the other image under a rigid transformation), partial occlusions and changes of illumination and of viewpoint.

[9] This issue has been extensively discussed in [15].

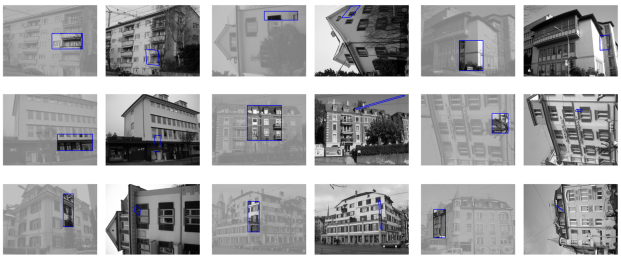


Fig. 12 Zurich Dataset [22] - the remaining: Failures (row 1), Occlusions (row 2), Template or Target template is out of plane/image (row 3)

As there is no rigid transformation between the images, we evaluated the performance of fast match on 200 images visually. On 129 of these we found that the mapping produced by the algorithm was good, in the sense that it corresponded almost exactly to what we judged as the best mapping. In most of the remaining cases producing a good mapping from the given template was impossible: On 40 of the images, the location corresponding to the template was not present in the other image, or that the template spanned several planes which can not be mapped uniquely. In 12 of the images the location that the template was a photograph of was occluded by some outside element, such as a tree. In only 19 of the images was locating the template possible, and the algorithm failed to do so. Examples of good mappings can be found in Figure 11. Examples of cases where a good match was not found appear in Figure 12. The results on the entire dataset appear in [11].

6 Conclusions

We presented a new algorithm, Fast-Match, which extends template matching to handle arbitrary 2D affine transformations. It overcomes some of the shortcomings of current, more general, image matching approaches. We give guarantees regarding the SAD error of the match (appearance related) and these are shown to translate to satisfactory overlap errors (location related). The result is an algorithm which can locate sub-images of varying sizes in other images. We tested Fast-Match on several data sets, demonstrating that it performs well, being robust to different real-world conditions. This suggests that our algorithm can be suitable for practical applications. An interesting direction for future research is to apply similar methods to more diverse families of transformations (e.g. homographies) and in other settings, such as matching of 3D shapes.

Appendix - Proof of Theorem 1

We first restate Theorem 1 for completeness:

Let I_1, I_2 be images with dimensions n_1 and n_2 and let δ be a constant in $(0,1]$. For a transformation T', let T be the closest transformation to T' in the net $\mathcal{N}_\delta$ (which is a $\frac{\delta \cdot n_1^2}{\mathcal{V}}$-cover). It holds that: $|\Delta_{T'}(I_1,I_2) - \Delta_T(I_1,I_2)| \leq O(\delta)$.

To understand why the claim holds we refer the reader to Figure 13. Two close transformations T, T' map the template to two close parallelograms in the target image. Most of the error of the mapping T' is with respect to the area in the intersection of these parallelograms (the yellow region in Figure 13). This error cannot be greater than the total variation multiplied by the distance between the transformations T and T', as shown below. The rest of the error originates in the area mapped to by T' that is not in the intersection (the green region). The size of this area is also bounded by the distance between the transformations. Thus, the distance between the transformations, and the total variation, bound the difference in error between T and T'. This is formalized in the remainder of the section.

For convenience, throughout the discussion of the algorithm's guarantees we consider points in a continuous image plane instead of discrete pixels. Analyzing the problem in the continuous domain makes the theorem simpler to prove, avoiding several complications that arise due to the discrete sampling, most notable, that several pixels might be mapped to a single pixel. We refer the reader to a (slightly more involved) proof in the discrete domain, which we made available in a previous manuscript [10].

In order to switch to the continuous domain, we give some definitions and state some claims for points in the image plane. We begin by relating the intensity of points to that of pixels.

Definition 1 *The intensity of a point $p = (x,y)$ in the image plane (denoted $I_1(p)$) is defined as that of the pixel $q = ([x],[y])$, where $[\cdot]$ refers to the 'floor' operation. The point p is said to* land in q.

We now define the variation of a point and relate it to the variation of a pixel.

Definition 2 *The variation of a point p, which we denote $v(p)$, is $\max_{q\,:\,d(p,q)\leq 1} |I_1(p) - I_1(q)|$. Note that this is upper-bounded by the variation of the pixel that p lands in. For convenience of computation (this does not change the asymptotic results), for points p that have a distance of less than 1 from the boundary of the image, we define $v(p) = 1$.*

Finally, we define the total variation of an image in terms of the total variation of points in the image plane.

Definition 3 *The total variation of an image (or template) I_1 is $\int_{I_1} v(p)$. We denote this value $\mathcal{V}$. Note that this is upper bounded by the total variation computed over the pixels.*

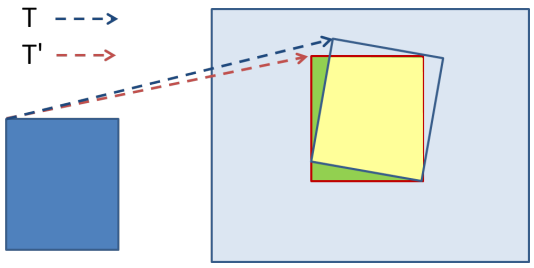


Fig. 13 A template mapped to an image by two close transformations. The close transformations map the template to close parallelograms. The error of T' cannot be very different from that of T. Most of the change in error is from different points being mapped to the intersection area (in yellow). This difference depends on the total variation of the template. The remaining error depends on the green area which is small because the transformations are close.

Our strategy towards proving Theorem 1 involves two ideas. First, instead of working with the pair of transformations T and T', we will more conveniently (and we show the equivalence) work with the identity transformation I and the concatenated transformation $T'^{-1}T$. Second, note that in Theorem 1, we bound the difference in error between transformations T and T', which are δn_1 apart. A simplifying approach, is to 'relate' the transformations T and T' through a series of transformations $\{T_i\}_{i=1}^m$ (where $T_0 = T$ and $T_m = T'$), which are each at most at a unit distance apart, with $m = O(\frac{\delta \cdot n_1^2}{\mathcal{V}})$. Thus, in Claim 6 we handle the case of transformations that are a unit distance apart.

In the following lemmas we introduce a constant u, such that if $\ell_\infty(T,T') \leq u$ it holds that $\ell_\infty(T^{-1},T'^{-1}) \leq 1$.

Claim 4 Given affine transformations T, T' with scaling factors in the range $[1/c, c]$ such that $\ell_\infty(T,T') \leq frac\delta \cdot n_1^2\mathcal{V}$, it holds that $\ell_\infty(T^{-1},T'^{-1}) = O(\frac{\delta \cdot n_1^2}{\mathcal{V}})$.

Proof To see that the claim above holds, consider a point q and we will show that $||T'^{-1}(q) - T^{-1}(q)|| \leq c\frac{\delta \cdot n_1^2}{\mathcal{V}}$ (see Figure 14). Let $p' = T'^{-1}(q)$ and let $p = T^{-1}(q)$. We wish to bound $||p' - p||$. Let $r = T(p')$. We get $||p' - p|| = ||T^{-1}r - T^{-1}q|| = ||T^{-1}(r-q)|| \leq c||r-q|| \leq c\frac{\delta \cdot n_1^2}{\mathcal{V}}$.

Claim 5 There exists a value $u \in (0,1)$ such that for any affine transformations T, T' where $\ell_\infty(T,T') \leq u$ and for any point $p \in I_1$, it holds that $||p, T'^{-1}(T(p))|| \leq 1$.

The correctness of Claim 5 follows directly from Claim 4 by noting that $p = T^{-1}(T(p))$.

Claim 6 Let I_1, I_2 be images with dimensions n_1 and n_2. There exists a constant $u \in (0,1)$ for which the following holds. For any two affine transformations T and T' such that $\ell_\infty(T,T') \leq u$:

$$|\Delta_{T'}(I_1,I_2) - \Delta_T(I_1,I_2)| \leq O\Big(\frac{\mathcal{V}}{{n_1}^2}\Big)$$

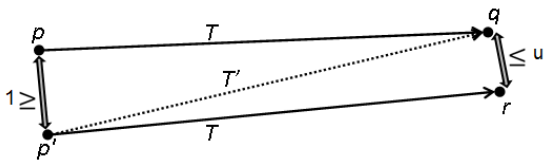


Fig. 14 Illustration for Claim 4. The distance between the points p and p' can be no more than a constant size greater than the distance between the points q and r, which is itself bounded by δn_1.

Note that the value $O\left(\frac{\mathcal{V}}{{n_1}^2}\right)$, bounds the difference in error for two transformations that have unit distance. This scales to the value $O(\delta)$ that appears in Claim 1 for transformations that have a distance of $\frac{\delta \cdot n_1^2}{\mathcal{V}}$.

Proof Using the triangle inequality we can write:

$$\left|\Delta_{T'}(I_1,I_2)-\Delta_T(I_1,I_2)\right| =$$
$$\left|\int_{I_1}|I_1(p)-I_2(T'(p))|-\int_{I_1}|I_1(p)-I_2(T(p))|\right| \leq$$
$$\int_{I_1}||I_1(p)-I_2(T'(p))|-|I_1(p)-I_2(T(p))||$$

where integrals go over points p in the template I_1.

We now bound this sum. As we know that $\ell_\infty(T,T') \leq u$, we know that only points that have a distance of at most 1 (as $u \leq 1$) from the boundary of I_1 are mapped to 'new' areas of I_2 - areas to which no point from I_1 was mapped before. Each of these points has an error of 1 at worst (this is the greatest distance possible between intensities from 0 to 1). The total area of such points is $O(n_1)$, and thus they contribute $O(n_1)$ to the difference between $\Delta_{T'}(I_1,I_2)$ and $\Delta_T(I_1,I_2)$, before normalization. This is equal to their contribution to the total variation.

For the remaining points (that have distance greater than 1 from the boundary of I_1), under T each such point p is mapped to a point $T(p)$, and the pre-image of that point $T'^{-1}(T(p))$, is in the area of I_1. Instead of considering the value $E_{T,I_1,I_2}(p)$ for each such point p in I_1, consider instead the error over each point $q=T(p)$ in I_2 that has points mapped to it both by T and by T'. The distance between p and $T'^{-1}(T(p))$ is at most 1 (as seen in Claim 5), and the value of p and of $T'^{-1}(T(p))$ differ by at most $v(p)$, and thus $|I_2(q)-I_1(p)|-|I_2(q)-I_1(T'^{-1}(q))| \leq v(p)$ (By a triangle inequality). Thus, for points that have a distance greater than 1 from the boundary of I_1, the affect on the difference $|\Delta_{T'}(I_1,I_2)-\Delta_T(I_1,I_2)|$ for each point p is at most $v(p)$ and thus the total contribution is bounded by $\mathcal{V}$.

Summing both contributions and normalizing by ${n_1}^2$ we obtain $|\Delta_{T'}(I_1,I_2)-\Delta_T(I_1,I_2)| = O(\mathcal{V}/{n_1}^2)$ as required.

However, not all transformations have a distance of u from the net. We now turn to the goal of this section, proving Theorem 1.

Proof As T is the closest transformation to T' it holds that $\ell_\infty(T,T') \leq \frac{\delta \cdot n_1^2}{\mathcal{V}}$. Furthermore, from the construction that is summarized in Claim 2 we have that $T = TrR_2SR_1$ and $T' \in \mathcal{N}_\delta$ where $T' = Tr'R_2'S'R_1'$ such that $d(Tr,Tr') \leq \frac{\delta \cdot n_1^2}{\mathcal{V}}$, ... $d(R_1,R_1') \leq \frac{\delta \cdot n_1^2}{\mathcal{V}}$. Now consider a series of transformations $\{T_i\}_{i=1}^m$ where $T_0 = T$ and $T_m = T'$. For each transformation T_{i+1} it will hold that $\ell_\infty(T_i,T_{i+1}) \leq u$ (for the constant u from Claim 6). For such a series, repeated use of Claim 6 (and of the triangle inequality) will give us that

$$|\Delta(T)-\Delta(T')| = |\Delta(T_0)-\Delta(T_m)| \leq O\left(\frac{m\mathcal{V}}{{n_1}^2}\right)$$

To construct such a series of transformations we first add (or subtract) u from the translation matrix until it changes from Tr to Tr'. This takes $O(\frac{\delta \cdot n_1^2}{\mathcal{V}})$ steps. We then change the rotation matrix beginning with R_2 by u/n_1 for $O(\frac{\delta \cdot n_1^2}{\mathcal{V}})$ steps until we get to R_2'. We proceed like this and after $m = O(\frac{\delta \cdot n_1^2}{\mathcal{V}})$ steps transition from T to T', giving us the required bound of $O(\delta)$.

References

1. Bogdan Alexe, Viviana Petrescu, and Vittorio Ferrari. Exploiting spatial overlap to efficiently compute appearance distances between image windows. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2735–2743, 2011.
2. Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework. *International journal of computer vision (IJCV)*, 56(3):221–255, 2004.
3. Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision (IJCV)*, 88(2):303–338, 2010.
4. Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.
5. Kimmo Fredriksson. *Rotation Invariant Template Matching*. PhD thesis, University of Helsinki, 2001.
6. Chiou-Shann Fuh and Petros Maragos. Motion displacement estimation using an affine model for image matching. *Optical Engineering*, 30(7):881–887, 1991.
7. Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.
8. Hae Yong Kim and Sidnei Alves De Araújo. Grayscale template-matching invariant to rotation, scale, translation, brightness and contrast. In *Advances in Image and Video Technology (AIVT)*, pages 100–113. Springer, 2007.
9. Igor Kleiner, Daniel Keren, Ilan Newman, and Oren Ben-Zwi. Applying property testing to an image partitioning problem. *Pattern Analysis and Machine Intelligence (PAMI)*, 33(2):256–265, 2011.
10. Simon Korman, Daniel Reichman, and Gilad Tsur. Tight approximation of image matching. *arXiv preprint arXiv:1111.1713*, 2011.
11. Simon Korman, Daniel Reichman, Gilad Tsur, and Shai Avidan. Fast-Match webpage. `www.eng.tau.ac.il/~simonk/FastMatch`.
12. David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision (IJCV)*, 60(2):91–110, 2004.

13. Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. In *IJCAI*, volume 81, pages 674–679, 1981.
14. Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *Pattern Analysis and Machine Intelligence (PAMI)*, 27(10):1615–1630, 2005.
15. Krystian Mikolajczyk, Tinne Tuytelaars, Cordelia Schmid, Andrew Zisserman, Jiri Matas, Frederik Schaffalitzky, Timor Kadir, and Luc Van Gool. A comparison of affine region detectors. *International journal of computer vision (IJCV)*, 65(1-2):43–72, 2005.
16. Jean-Michel Morel and Guoshen Yu. Asift: A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, 2(2):438–469, 2009.
17. Wanli Ouyang, Federico Tombari, Stefano Mattoccia, Luigi Di Stefano, and Wai-Kuen Cham. Performance evaluation of full search equivalent pattern matching algorithms. *Pattern Analysis and Machine Intelligence (PAMI)*, 34(1):127–143, 2012.
18. Ofir Pele and Michael Werman. Accelerating pattern matching or how much can you slide? In *Asian Conference on Computer Vision (ACCV)*, pages 435–446. Springer, 2007.
19. Sofya Raskhodnikova. Approximate testing of visual properties. In *Workshop on Randomization and Approximation Techniques in Computer Science, (RANDOM)*, pages 370–381, 2003.
20. William Rucklidge. Efficient guaranteed search for gray-level patterns. In *Computer Vision and Pattern Recognition (CVPR)*, pages 717–723. IEEE, 1997.
21. Steven M Seitz and Simon Baker. Filter flow. In *International Conference on Computer Vision (ICCV)*, pages 143–150. IEEE, 2009.
22. Hao Shao, Tomáš Svoboda, and Luc Van Gool. Zubud-zurich buildings database for image based recognition. *Swiss Federal Institute of Technology, Switzerland, Tech. Rep*, 260, 2003.
23. Yuandong Tian and Srinivasa G Narasimhan. Globally optimal estimation of nonrigid image distortion. *International journal of computer vision (IJCV)*, 98(3):279–302, 2012.
24. Du-Ming Tsai and Cheng-Huei Chiang. Rotation-invariant pattern matching using wavelet decomposition. *Pattern Recognition Letters*, 23(1):191–201, 2002.
25. Gilad Tsur and Dana Ron. Testing properties of sparse images. In *Symposium on Foundations of Computer Science (FOCS)*, pages 468–477. IEEE, 2010.
26. A van der Schaaf and JH van Hateren. Modelling the power spectra of natural images: statistics and information. *Vision research*, 36(17):2759–2770, 1996.
27. Cheng-Hao Yao and Shu-Yuan Chen. Retrieval of translated, rotated and scaled color textures. *Pattern Recognition*, 36(4):913–929, 2003.