Published in Image Processing On Line on 2018–12–24.
Submitted on 2017–01–27, accepted on 2018–11–29.
ISSN 2105–1232 © 2018 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2018.202

An Affine Invariant Patch Similarity

Vadim Fedorov, Coloma Ballester

DTIC, Universitat Pompeu Fabra, Spain
{vadim.fedorov, coloma.ballester}@upf.edu

Abstract

Image and video comparison is often approached by comparing patches of visual information. In this work we present a detailed description and implementation of an affine invariant patch similarity measure that performs an appropriate patch comparison by automatically and intrinsically adapting the size and shape of the patches. We also describe the complete implementation of the proposed iterative algorithm for computation of those shape-adaptive patches around each point in the image domain.

Source Code

The implementation, written in C++, is available at the IPOL web page of this article[1]. Compilation and usage instructions are included in the `README.md` file of the archive.

Keywords: patch similarity; affine invariance; structure tensor; shape-adaptive patches

1 Introduction

Image comparison is a topic that has received a lot of attention from the image processing and computer vision communities. It is a main ingredient in many applications such as object recognition, stereo vision, image registration, image denoising, exemplar-based image inpainting, to name a few. There are plenty of techniques for image comparison and the particular choice always depends on the specific task.

Very frequently it is required to compare any two given points in an image or, in a more general situation, in two different images. Since a single color value of a point is not very descriptive, it is common to use a small neighborhood around that point in the comparison. Traditionally such neighborhoods are called *patches*. A distinctive property of all patch-wise image comparison techniques, sometimes also called template matching, is that they assign a similarity value (or comparison distance) to any given pair of points. In other words, for every single point in one image there is a *dense* similarity (or distance) field associated with another image. Let $u : \Omega_u \subset \mathbb{R}^N \to \mathbb{R}^M$ and $v : \Omega_v \subset \mathbb{R}^N \to \mathbb{R}^M$ be two images with M color channels. Here Ω_u and Ω_v denote their image domains. This general definition of u and v aims to emphasize that the image comparison theory deals

[1] https://doi.org/10.5201/ipol.2018.202

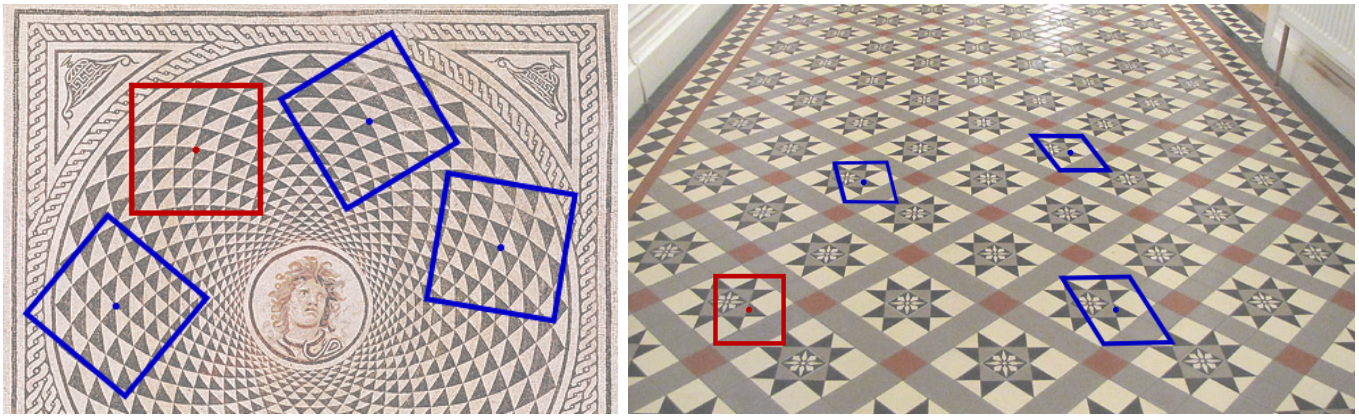

Figure 1: Possible extension of the space of available patches. On the left: rotations. On the right: full affine transformations.

both with regular 2D images and videos, as well as with 3D images, sequences of 3D images captured over time and other more exotic data structures. The patch-wise image comparison functions can then be generalized as

$$\begin{aligned} \mathcal{D} \ : \ \Omega_u \ &\times \ \Omega_v \longrightarrow \mathbb{R} \\ (x \ \ &, \ \ y) \longrightarrow \mathcal{D}(x,y), \end{aligned}$$

where u and v may coincide. Let p denote a patch domain, which usually is given by a connected subset of $\mathbb{R}^N$ centered at the origin (for instance a disk, or a square, centered at $0 \in \mathbb{R}^N$). Some examples of such functions, that are usually called *similarity measures*, are:

- Maximum Absolute Difference

$$\mathcal{D}_{\max}(x,y) = \max_{h \in p} |u(x+h) - v(y+h)| \,,$$

- Cross-Correlation

$$\mathcal{D}_{\mathrm{CC}}(x,y) = \sum_{h \in p} u(x+h)v(y+h),$$

- Sum of Absolute Differences

$$\mathcal{D}_{\mathrm{SAD}}(x,y) = \sum_{h \in p} |u(x+h) - v(y+h)| \,,$$

- Sum of Squared Differences

$$\mathcal{D}_{\mathrm{SSD}}(x,y) = \sum_{h \in p} (u(x+h) - v(y+h))^2 \,.$$

Of course, much more elaborated similarity measures can be found in the literature, such as Zero-mean Normalized Cross-Correlation (ZNCC) in [17], Phase Correlation in [11], IMage Euclidean Distance (IMED) in [33], etc.

We consider patches to be the basic units of information that allow us to analyze and exploit the self-similarity property, usually attributed to natural images, and which is a prior widely used in, e.g. most of the state-of-the-art methods for image denoising, restoration, super-resolution, inpainting and object recognition [5, 24, 27, 26, 1, 16, 25, 14, 8, 7]. Commonly, patches are set to have square

shape and fixed size, and are used as they appear in an image without any transformations. However, in the image formation process the final appearance of visual details of an observed scene is affected by the geometry of that scene and by their positions with respect to the camera. Therefore, for many tasks that involve comparison it might be beneficial to consider shape-adaptive patches despite the additional computational burden associated with them. This point is illustrated by Figure 1, where reference patches are shown in red and several other patches, containing similar visual details, are shown in blue. In the left image similar patches are related by rotations. In the right image patches containing the same pieces of texture are related by more complex affine transformations.

This work aims at promoting an *affine invariant patch similarity measure* that naturally also implies shape-adaptive patches. This similarity measure provides dense image comparison (in the sense that the local similarity can be computed for any arbitrary pair of points), and at the same time it is invariant to affine transformations. The main intuition is simple: the similarity measure compares elliptical patches (defined as spatially varying balls of the underlying metric previously defined in the image domain) by normalizing them to a common disk domain. Potential applications of such similarity measure are numerous, for example, a novel image inpainting method was presented recently in [8], a variational segmentation method in [23], and an image denoising method achieving top-tier performance in [10]. The similarity measure was originally introduced and studied in [9] (see also [7]), whereas this work is focused on the numerical implementation. Other affine invariant similarity measurements were proposed in the literature [20, 19, 31, 13, 30].

In the next section we briefly recall from [9] that the properties of the proposed affine invariant similarity measure are closely related to the so called affine covariant structure tensors. In Section 3 we discuss in detail the numerical implementation, and then in Section 4 show some similarity maps computed with the source code that accompanies this paper.

2 Affine Invariant Patch Similarity Measure

The affine invariant similarity measure that we consider in this work was originally introduced in [9], where it was derived from a linear model proposed, among many other models, for multiscale analysis of similarities between images on Riemannian manifolds, in [2]. Assuming, to simplify, that $\Omega_u = \Omega_v = \mathbb{R}^N$, the considered affine invariant similarity measure between two points $x, y \in \mathbb{R}^N$ is

$$\mathcal{D}^{\mathrm{a}}(t,x,y) = \int_{\mathbb{R}^2} g_t(h) \left(u(x + G_1(x)^{-\frac{1}{2}} h) - v(y + G_2(y)^{-\frac{1}{2}} h) \right)^2 dh, \tag{1}$$

where $G_1(x)$ and $G_2(y)$ are spatially varying Riemannian metrics defined on the image domains of u and v, respectively,

$$g_t(h) = \eta \, \exp\left(-\frac{d_{G_1}(x, x + G_1(x)^{-\frac{1}{2}} h)^2}{t} \right), \tag{2}$$

$d_{G_1}(\tilde{x}, \tilde{\tilde{x}})$ denotes the distance on a geodesic curve joining points $\tilde{x}$ and $\tilde{\tilde{x}}$ in the Riemannian manifold (Ω_u, G_1) and η is an appropriate normalization constant depending on t. The t parameter allows to control the support in the patch comparison, and is directly related to the multiscale character of $\mathcal{D}^{\mathrm{a}}$ [2, 9]. As will become clearer later in this section (see also [2, 9, 7]), the points $x + G_1(x)^{-\frac{1}{2}} h$ and $y + G_2(y)^{-\frac{1}{2}} h$ belong, respectively, to patches around x and y, respectively, when h belongs to the support of g_t (which, in practice, will be considered bounded as usual).

In (1) the similarity measure is given in a general form, since G_1 and G_2 can be any Riemannian metrics on the respective image domains of u and v. It is well known, that the structure tensor (second-moment matrix) can be seen as a metric in the image domain (e.g. [35, 36, 15, 4, 3]). A novel iterative scheme was presented in [9, 7] to define and compute the structure tensors. This

scheme guarantees, at least in the continuous setting, that the structure tensors are *affine covariant* which means that they transform appropriately by affine transformations. A study about how these tensor properties behave in the case of discrete real image was performed in [9]. In this section we first recall from [9] some definitions and then address the algorithmic construction of the *affine covariant structure tensors* and *affine covariant regions* (shape-adaptive patches). We also illustrate the geometrical meaning of using these affine covariant structure tensors as metrics G_1 and G_2 in (1).

2.1 Preliminaries

Let u be a given image, $u : \mathbb{R}^N \to \mathbb{R}$ that we assume to be a function of bounded variation, $u \in BV(\mathbb{R}^N)$. Let $GL(N)$ be the set of invertible matrices in $\mathbb{R}^N$. Let $A \in GL(N)$ be an affine transformation. We denote by $u_A(x) := u(Ax)$ a version of an image u, transformed by an affinity A. Notice that it is equivalent to say that A transforms the coordinate system of the domain of u_A to the one of u.

Definition 1. *Let H_u be a $(1,1)$ tensor defined on $\mathbb{R}^N$ such that, for each $x \in \mathbb{R}^N$, it is represented by a $N \times N$ matrix $H_u(x)$ mapping a vector in $\mathbb{R}^N$ to another vector in $\mathbb{R}^N$. We say that H_u is an affine covariant tensor if it satisfies*

$$H_{u_A}(x) = A^t H_u(Ax) A, \tag{3}$$

where $u_A(x) := u(Ax)$ for $A \in GL(N)$.

Let us highlight a slight clash of terminology that we committed here. In general, a tensor is said to be of type (m,n) if it is contravariant of order m and covariant of order n. Thus, the term "covariant tensor" is used to denote any tensor of type $(0,n)$, that is, any tensor that has n covariant indices and no contravariant indices. On the other hand, in the context of Definition 1 we say "affine covariant" to emphasize that such tensor, computed from an image u, transforms in accordance with an affinity.

An interesting example of such a tensor is given by $F(u) = \nabla u \otimes \nabla u$, where ∇ denotes the gradient operator and $\otimes$ denotes the tensor product. The tensor field $F(u)$ is affine covariant, because

$$\begin{aligned} F(u_A)(x) &= \nabla u_A(x) \otimes \nabla u_A(x) = A^t \nabla u(Ax) \otimes A^t \nabla u(Ax) \\ &= A^t \nabla u(Ax) \otimes \nabla u(Ax) A = A^t F(u)(Ax) A. \end{aligned} \tag{4}$$

The law of transformations (3) is well adapted to define neighborhoods that transform properly with respect to affine transformations. Indeed, it can be shown [9] that

Lemma 2.1. *Let H_u be an affine covariant tensor. Let*

$$B_{H_u}(x,r) = \{y : \langle H_u(x)(y-x), (y-x)\rangle \le r^2\} \qquad x \in \mathbb{R}^N,\ r > 0. \tag{5}$$

Then, $B_{H_{u_A}}(x,r) = A^{-1} B_{H_u}(Ax,r)$.

We say that $B_{H_{u_A}}(x,r)$ is an *affine covariant neighborhood*.
In particular, if we define

$$B_u(x,r) = \{y : |\langle \nabla u(x), y-x\rangle| \le r\}, \tag{6}$$

then

$$\begin{aligned} B_{u_A}(x,r) &= \{y : |\langle \nabla u_A(x), y-x\rangle| \le r\} \\ &= A^{-1}\{y : y \in B_u(Ax,r)\} = A^{-1} B_u(Ax,r), \end{aligned}$$

that is, $B_{u_A}(x,r)$ is an affine covariant neighborhood as well.

2.2 Iterative Construction Scheme

At this point we have all the ingredients we need to describe the scheme for the construction of affine covariant structure tensors and affine covariant neighborhoods.

Let $B_u(x,r)$ be an affine covariant neighborhood in image u. For example, $B_u(x,r)$ can be computed using (6). Let

$$T(u)(x) = \int_{B_u(x,r)} \nabla u(y) \otimes \nabla u(y)\, dy. \tag{7}$$

Let us recall from [9] some properties of affine covariant tensors that were proved there. First, notice that $T(u)$ satisfies

$$T(u_A)(x) = \int_{B_{u_A}(x,r)} \nabla u_A(y) \otimes \nabla u_A(y)\, dy = \int_{A^{-1}B_u(Ax,r)} A^t \nabla u(Ay) \otimes \nabla u(Ay) A\, dy, \tag{8}$$

and by writing $\bar{y} = Ay$, $y \in A^{-1}B_u(Ax,r)$ we get

$$T(u_A)(x) = A^t \int_{B_u(Ax,r)} \nabla u(\bar{y}) \otimes \nabla u(\bar{y}) |\mathrm{det}A|^{-1}\, d\bar{y} A. \tag{9}$$

Let us remark that $T(u_A)(x) = |\mathrm{det}A|^{-1} A^t T(u)(Ax)A$; therefore, $T(u_A)$ is an affine covariant tensor with a weight expressed by $|\mathrm{det}A|^{-1}$. We refer to it as an affine covariant tensor density of exponent -1. More precisely, we will say that the image dependent tensor T is an *affine covariant tensor density of exponent* $p \in \mathbb{R}$ if it satisfies $T(u_A)(x) = |\mathrm{det}A|^p A^t T(u)(Ax)A$ for any affine transformation $A \in GL(N)$, where u_A is the corresponding affinely transformed image defined by $u_A(x) = u(Ax)$.

Although the following results hold for the general case of $\mathbb{R}^N$, we will focus on the case of regular images where $B_u(x,r) \subset \mathbb{R}^2$. To cancel the factor $|\mathrm{det}A|^{-1}$, we observe that $\mathrm{Area}(B_{u_A}(x,r)) = |\mathrm{det}A|^{-1}\mathrm{Area}(B_u(Ax,r))$. Therefore, if we normalize $T(u)(x)$ and define

$$NT(u)(x) = \frac{\int_{B_u(x,r)} \nabla u(y) \otimes \nabla u(y)\, dy}{\mathrm{Area}(B_u(x,r))}, \tag{10}$$

we have $NT(u_A)(x) = A^t NT(u)(Ax)A$. In other words, $NT(u)$ is an affine covariant tensor (or an affine covariant tensor density of exponent 0), computed on an affine covariant neighborhood.

Lemma 2.2. *Let H^1 be an affine covariant tensor density of exponent $p \in \mathbb{Z}$ and let H^2 be an affine covariant tensor. Let H^i_A be the tensor after the affine transformation A. Let $B_{H^1}(x,r)$ be an affine covariant neighborhood, computed from H^1. Let*

$$T(H^1, H^2)(x) = \int_{B_{H^1}(x,r)} H^2(y)\, dy. \tag{11}$$

Then

$$T(H^1_A, H^2_A)(x) = |\mathrm{det}A|^p A^t T(H^1, H^2)(Ax)A. \tag{12}$$

That is, $T(H^1, H^2)$ is an affine covariant tensor density of exponent p.

Notice that, in our examples, $p = 0$ or $p = -1$. Lemma 2.2 permits to iterate the above construction (10) and redefine for $k \geq 1$

$$NT^{(k)}(u)(x) = \frac{\int_{B_{NT^{(k-1)}(u)}(x,r)} \nabla u(y) \otimes \nabla u(y)\, dy}{\mathrm{Area}(B_{NT^{(k-1)}(u)}(x,r))}, \tag{13}$$

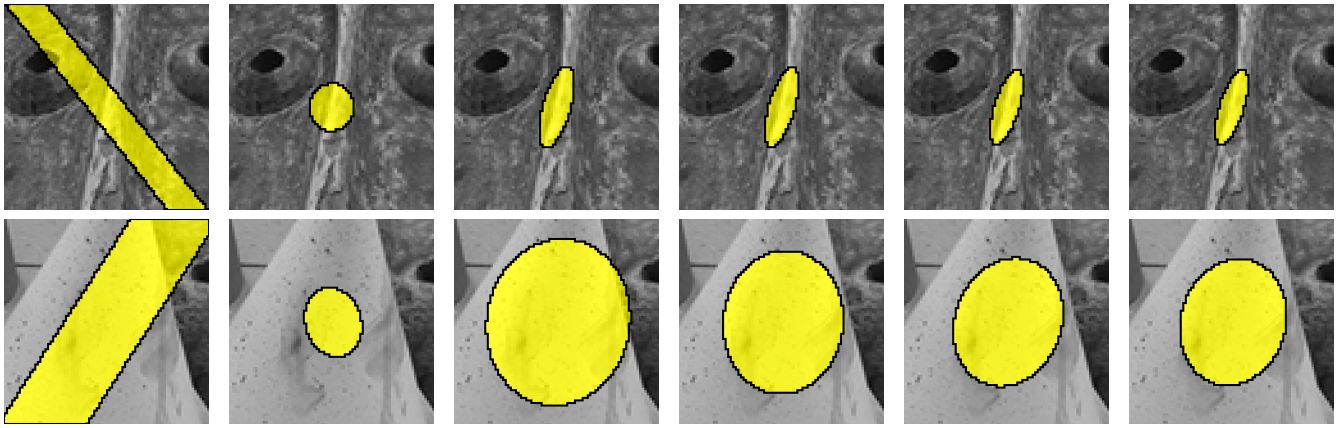

Figure 2: Evolution of two affine covariant neighborhoods over iterations of the construction scheme (k from 0 to 5).

where k is the index of iteration, and

$$B_{NT^{(k)}(u)}(x,r) = \{y : \langle NT^{(k)}(u)(x)(y-x),(y-x)\rangle \leq r^2\}, \tag{14}$$

for $k \geq 1$,

$$B_{NT^{0}(u)}(x,r) = \{y : |\langle \nabla u(x), y-x\rangle| \leq r\}, \tag{15}$$

for $k = 0$.

Equations (13), (14) and (15) constitute an iterative scheme for the calculation of affine covariant structure tensors and neighborhoods. Algorithm 1 details the pseudo-code of the provided code for its computation.

Notice that the initial neighborhood (15) takes into account non-local information due to its infinite band support. Moreover, its shape depends only on the gradient at a single point and thus may be subjected to noise. The iterative process decreases this dependency of the structure tensor on the initial neighborhood $B_{NT^0(u)}(x,r)$. Figure 2 illustrates the evolution of two affine covariant neighborhoods over iterations.

It was observed that after a few iterations the proposed scheme either converges to a single affine covariant structure tensor, or starts to cycle over a finite number of them (typically 2 or 3). Notice, however, that by Lemma 2.2 the structure tensor is guaranteed to be affine covariant at any iteration of the scheme. Therefore, for the correct comparison of two points in the case of oscillation, it is only required to run the construction scheme for the same amount of iterations k. We refer the reader to [9, 7] for the empirical study of the convergence of the iterative scheme and the above mentioned dependency on the initial iteration.

To simplify the notation, we will denote by $T_u(x)$ the affine covariant structure tensor $NT^{(k)}(u)(x)$ for a fixed number of iterations k and a given value of r (see Algorithm 1). We say that T_u is the affine covariant structure tensor field associated with u. Similarly we will use the notation $B_{T_u}(x)$ to refer to the affine covariant neighborhood $B_{NT^{(k)}(u)}(x,r)$.

Let us note that r is a free parameter. It controls the size of the affine covariant neighborhood at a given point. On the other hand, the size of the neighborhood is also affected by the texture in the vicinity of that point. For the same value of r, elliptical patches are always significantly bigger in homogeneous regions than in textured regions or close to edges. Some examples of the affine covariant neighborhoods, computed using the same value of r, are shown in Figure 3. The choice of r might depend also on the application. For instance, in [22, 23] a thorough analysis was made to experimentally show the robustness of a patch-based variational segmentation method that considers the same adaptive patches depending on the values of t and r. It was also analyzed in [10] for image

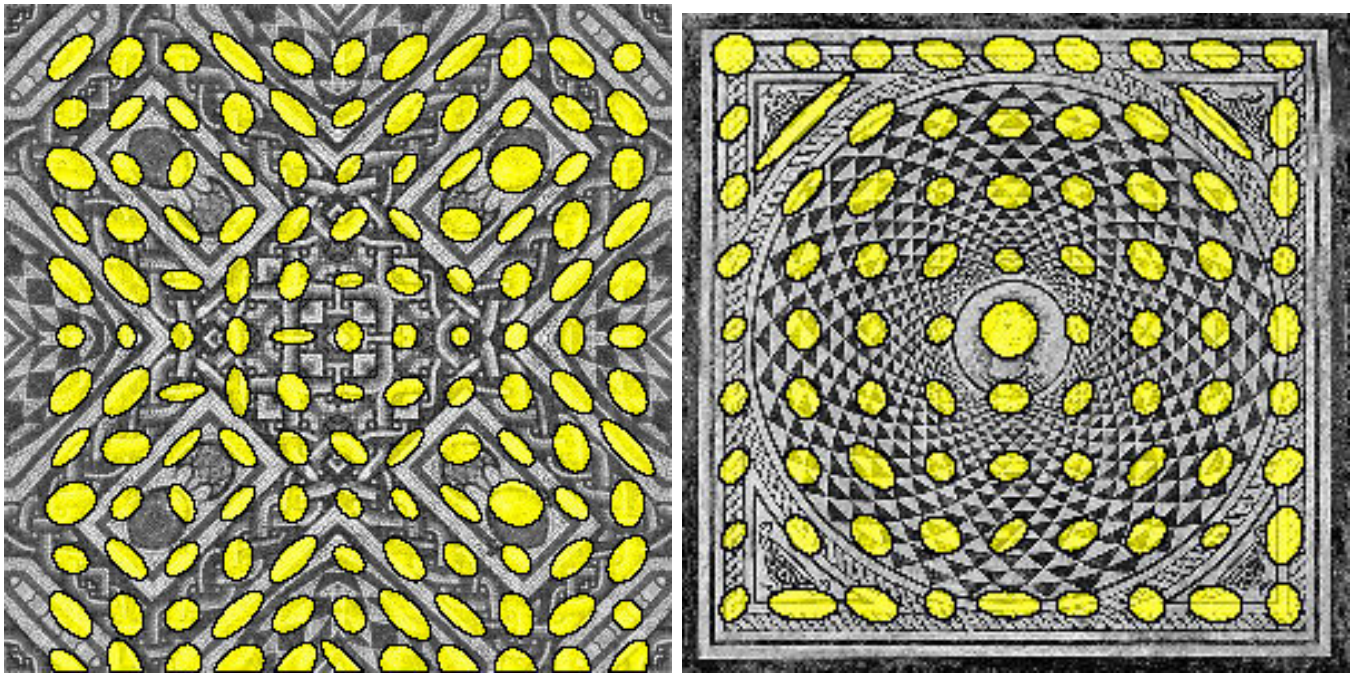

Figure 3: Affine covariant neighborhoods computed every 25 pixels using the same values of r.

denoising where, on the other hand, a patch size constraint limiting the maximum patch size was introduced. It was developed by slightly modifying the tensors adding an appropriate constant to its diagonal [10]. Obviously, this modification breaks the affine covariant property of the structure tensors and associated neighborhoods but limiting the maximum patch size to capture only small pieces of visual information, with the limit depending on the noise level, has proven to be beneficial in the context of denoising.

2.3 Local Affinity from Structure Tensors

Let $u : \Omega_u \to \mathbb{R}$ and $v : \Omega_v \to \mathbb{R}$ be two given images. For each point $x \in \Omega_u$, let $T_u(x)$ be the structure tensor of u, and let $T_v(y)$ be the structure tensor of v at $y \in \Omega_v$. By extending the domains of u and v to $\mathbb{R}^N$ and considering the structure tensor fields T_u and T_v as metrics on these domains, we obtain two Riemannian manifolds $(\mathbb{R}^N, G_1 := T_u)$, $(\mathbb{R}^N, G_2 := T_v)$.

As was stated in Definition 1, a structure tensor that is affine covariant satisfies

$$T_{u_A}(x) = A^t T_u(Ax) A,$$

where $u_A(x) := u(Ax)$. When local vicinities of two given points x and y are related by a local affine transformation $A(x, y)$, we expect

$$T_u(x) = A(x, y)^t T_v(y) A(x, y).$$

This matrix equation has many solutions, as will be seen shortly, and the right solution needs to be precised. For the moment, let us continue our argument with the solution

$$A(x, y) = T_v(y)^{-\frac{1}{2}} T_u(x)^{\frac{1}{2}}. \tag{16}$$

To calculate the square root of $T_u(x)$ and $T_v(y)$ we first diagonalize the matrices

$$T_u(x) = U_u(x) D_u(x) U_u^t(x),$$

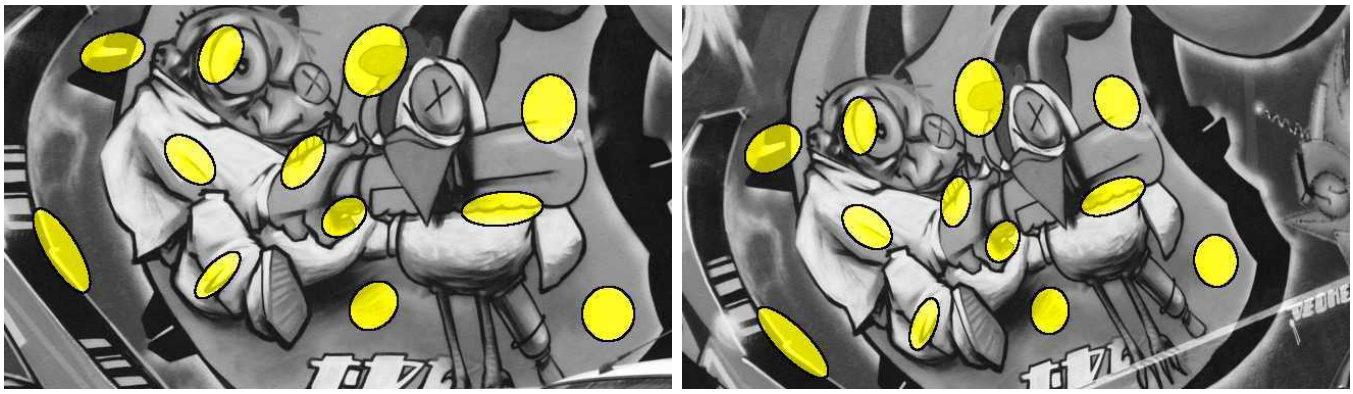

Figure 4: Affine covariant neighborhoods (shape-adaptive patches) computed at corresponding points in two images. Despite the difference in viewpoints, the patches capture the same visual information.

$$T_v(y) = U_v(y) D_v(y) U_v^t(y).$$

Here

$$D_u(x) = \mathrm{diag}(\lambda_{u,1}(x), \lambda_{u,2}(x)), \qquad \lambda_{u,1}(x) \geq \lambda_{u,2}(x),$$
$$D_v(y) = \mathrm{diag}(\lambda_{v,1}(y), \lambda_{v,2}(y)), \qquad \lambda_{v,1}(y) \geq \lambda_{v,2}(y).$$

The matrices $U_u(x)$ and $U_v(y)$ are rotation matrices formed by the eigenvectors of $T_u(x)$ and $T_v(y)$, respectively. Let $e_{u,i}(x)$ be the eigenvector of $T_u(x)$ associated with the eigenvalue $\lambda_{u,i}(x)$, $i \in \{1, 2\}$. Let $e_{v,i}(y)$ be the eigenvector of $T_v(y)$ associated with the eigenvalue $\lambda_{v,i}(y)$. That is, $e_{u,i}(x)$ is the i-th column of $U_u(x)$ and $e_{v,i}(y)$ is the i-th column of $U_v(y)$.

Recall that each structure tensor is associated with its corresponding region, which in $\mathbb{R}^2$ is an ellipse given by

$$B_{T_u}(x) = \{\bar{x} : \langle T_u(x)(\bar{x} - x), \bar{x} - x\rangle \leq r^2\},$$
$$B_{T_v}(y) = \{\bar{y} : \langle T_v(y)(\bar{y} - y), \bar{y} - y\rangle \leq r^2\}.$$

Figure 4 shows examples of affine covariant regions computed at corresponding points in two views of the same scene.

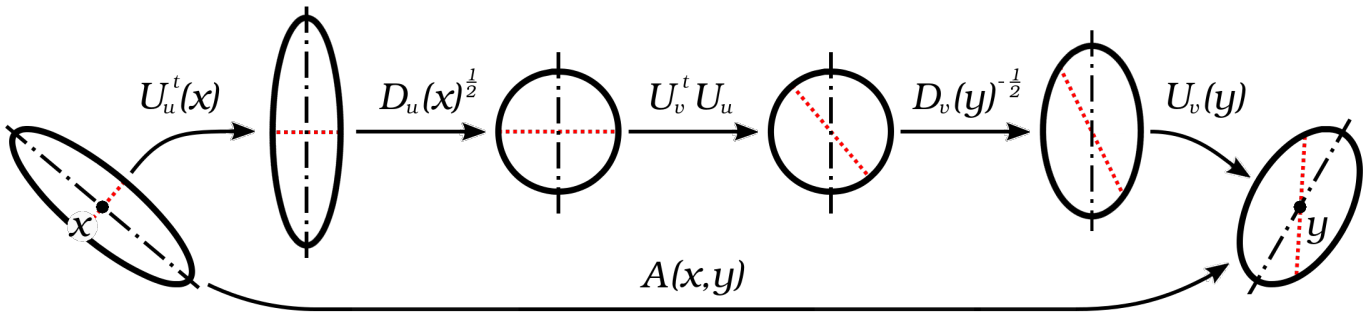

Figure 5: Decomposition of the transformation, obtained from a pair of structure tensors.

If we define $A_u(x) := D_u(x)^{\frac{1}{2}} U_u(x)^t$, then by the change of variables $X = A_u(x)x'$, we have $A_u(x)\frac{e_{u,i}(x)}{\sqrt{\lambda_{u,i}(x)}} = f_i$, where f_i is a Euclidean orthonormal basis. Which means that $U_u(x)^t$ rotates the ellipse, aligning the minor axis to f_1 and the major to f_2, and $D_u(x)^{\frac{1}{2}}$ changes the length of both axis. Similarly for the ellipse associated with $T_v(y)$, we define $A_v(y) := D_v(y)^{\frac{1}{2}} U_v(y)^t$ and, by the

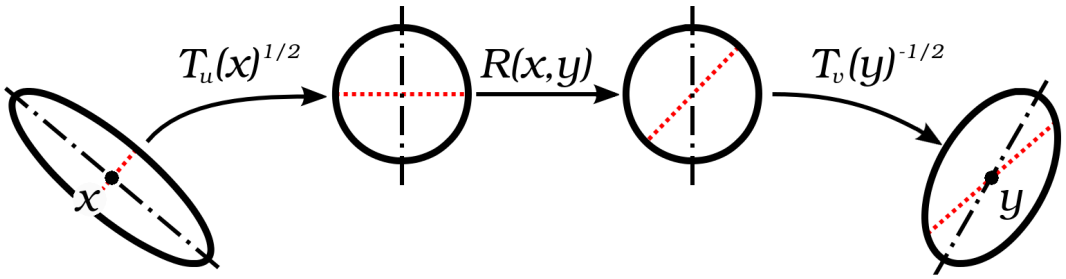

Figure 6: Schematic illustration of alignment of elliptical patches that takes into account the additional orthogonal transformation $R(x, y)$.

change of variables $Y = A_v(y)y'$, we have $A_v(y)\frac{e_{v,i}(y)}{\sqrt{\lambda_{v,i}(y)}} = f_i$. After these operations both ellipses are transformed into disks of radius r.

Using the above notation we can rewrite (16) as

$$A(x, y) = T_v(y)^{-\frac{1}{2}} T_u(x)^{\frac{1}{2}} = U_v(y) D_v(y)^{-\frac{1}{2}} U_v(y)^t U_u(x) D_u(x)^{\frac{1}{2}} U_u(x)^t. \tag{17}$$

This combined transformation warps an elliptical region at point x into an elliptical region at point y (Figure 5). Notice, however, that this transformation does not necessarily match the true affine transformation between the vicinities of points x and y. As will be shown in the next section, there is an orthogonal transformation missing in (17) between $U_v(y)^t$ and $U_u(x)$.

2.4 Additional Orthogonal Transformation

Similarly to [12], it can be shown that in general Equation (17) allows to determine a local affine transformation from two affine covariant structure tensors, but only up to some orthogonal transformation [9]. The *exact* local affinity can be obtained, when an additional constraint is applied. For example, it is possible in the context of stereo imaging, when the vertical displacement of points is known to be zero after rectification. In our case, the true local affine transformation satisfies [9]

$$A(x, y) := T_v(y)^{-1} \widetilde{R}(x, y) T_u(x), \tag{18}$$

where $\widetilde{R}(x, y)$ is some additional orthogonal transformation. This equation can be rewritten as

$$A(x, y) = T_v(y)^{-1} \widetilde{R}(x, y) T_u(x) = U_v(y) D_v(y)^{-\frac{1}{2}} R(x, y) D_u(x)^{\frac{1}{2}} U_u(x)^t, \tag{19}$$

where $R(x, y)$ is an orthogonal transformation that "absorbs" the two rotations $U_v(y)^t$ and $U_u(x)$ in (17) and involves rotation and/or mirroring. Figure 6 illustrates the complete chain of transformations that align one patch with another.

The additional orthogonal transformation $R(x, y)$ can be computed from the image content inside elliptical regions at points x and y. When two corresponding elliptical regions are normalized to disks, a proper additional orthogonal transformation should align the content of these circular regions. Since exhaustive search is not an option for any practical application, we instead split the sought-for transformation into two parts

$$R(x, y) = R_v(y)^{-1} R_u(x),$$

each of which depends on image content around only one point, x or y. We can now look for some invariant features inside both normalized circular regions and compute from them the transformations $R_u(x)$ and $R_v(y)$. In Section 3.3 we explain the estimation of these transformations in detail.

We modify the Riemannian metrics G_1 and G_2, given by the affine covariant structure tensors, to also take into account the additional orthogonal transformation. Then,

$$G_1^{-\frac{1}{2}}(x) = T_u(x)^{-\frac{1}{2}} R_u(x)^{-1}, \qquad G_2^{-\frac{1}{2}}(y) = T_v(y)^{-\frac{1}{2}} R_v(y)^{-1}.$$

Then the similarity measure (1) becomes

$$\mathcal{D}^{\mathrm{a}}(t,x,y) = \int_{\Delta_t} g_t(h) \left\| u(x + T_u(x)^{-\frac{1}{2}} R_u(x)^{-1} h) - v(y + T_v(y)^{-\frac{1}{2}} R_v(y)^{-1} h) \right\|_2^2 dh, \tag{20}$$

where $g_t(h)$ is either a Gaussian of variance t, or an approximated geodesic weighting function, and Δ_t is a disk centered at the origin with radius proportional to the scale t and big enough such that g_t has effective support in Δ_t. In the geometrical interpretation of this similarity measure two elliptical patches at points x and y are first normalized by $R_u(x)T_u(x)$ and $R_v(y)T_v(y)$ to disks of the same size, and then compared within this common space.

3 Numerical Implementation

In this section we describe the numerical implementation of the proposed affine invariant patch similarity measure (20). Throughout this work we always refer to u and v as *images* defined in $\Omega_u, \Omega_v \subset \mathbb{R}^N$, respectively, and they might represent regular videos, sequences of 3D images captured over time, etc. In this section we consider discrete images $u : \Omega_u \cap \mathbb{Z}^N \to \mathbb{R}^M$ and $v : \Omega_v \cap \mathbb{Z}^N \to \mathbb{R}^M$ defined on $\mathbb{Z}^N$ with values in $\mathbb{R}^M$ (or with values in $([0, \mathrm{range}] \cap \mathbb{Z})^M$). Actually, we will consider the most practical case of $N = 2$ (the case of planar images) which is the one implemented in the source code accompanying the manuscript. The case of $N = 3$ is described in detail in [7], Chapter 6. The 2D+time video case was also applied in [32] to obtain a spatio-temporal video segmentation method that uses spatio-temporal adaptive patches. The number of color channels M is usually assumed to be 1 or 3; however, since the particular choice of M does not affect the reasoning, we do not specify it explicitly. Without loss of generality we assume that $u = v$.

The similarity measure (20) assigns a comparison distance to a pair of shape-adaptive patches centered at two given points in the following way. The scheme described in Section 2.2 allows to calculate at any given point x both the affine covariant structure tensor and the affine covariant region (shape-adaptive patch) defined by it. In order to compare two shape-adaptive patches we have to align (register) them first. A proper registration can be obtained from the structure tensors; however, as commented in Section 2.4, we have to specify additional orthogonal transformations. For the sake of simplicity in our implementation we restrict these orthogonal transformations to be rotations only and ignore the possibility of mirroring. Therefore, from now on we will refer to them as "additional rotations". We estimate these rotations by extracting dominant orientations of gradients within the patches. For the purposes of comparison we normalize shape-adaptive patches to disks of the same radius and interpolate these normalized versions to the regular grid. In this way we obtain a convenient representation of shape- and size-varying patches by arrays of interpolated color values whose sizes are all equal and known in advance. Notice that there might be several dominant orientations within a patch; therefore, for a single point x we might have multiple normalized versions of the patch. We compare every version associated with the point x with every version associated with the point y and finally assign to these points the smallest distance among all the combinations.

In the following sections we start by presenting a more formal overview of the numerical implementation and then describe specific parts of it in more detail.

3.1 Outline of the Patch Similarity Computation

This section presents an overview of the numerical implementation of the proposed affine invariant similarity measure. The following high-level outline of the algorithm is structured in the form of a data flow, where every step is described by a set of inputs ("in"), a set of internal parameters ("prm") and a set of outputs ("out"). Figure 7 shows a data flow diagram that graphically represents the outline. The locations x and y are the inputs of the algorithm itself. They can come from the same image u or from two different images, u and v; however, for simplicity of notation we assume here that $u = v$. The distance d, together with the corresponding configuration of additional rotations, are the outputs of the algorithm.

Let us recall some useful notation from the previous sections. We denote by ∇u the gradient of the given image u. We denote by $T_u(x)$ the affine covariant structure tensor at point x and by $B_{T_u}(x)$ its corresponding affine covariant neighborhood. The parameter r controls the size of the affine covariant neighborhoods in the construction scheme described in Section 2.2. The rest of the notation used in the outline is explained upon appearance.

Let us note that in the provided code the discretization of the derivatives in the gradient computation is implemented using image derivative filters [6, 28], in particular as convolution with $(-1, 8, 0, -8, 1)/12$ kernel, and with Neumann boundary conditions.

3.1.1 Outline

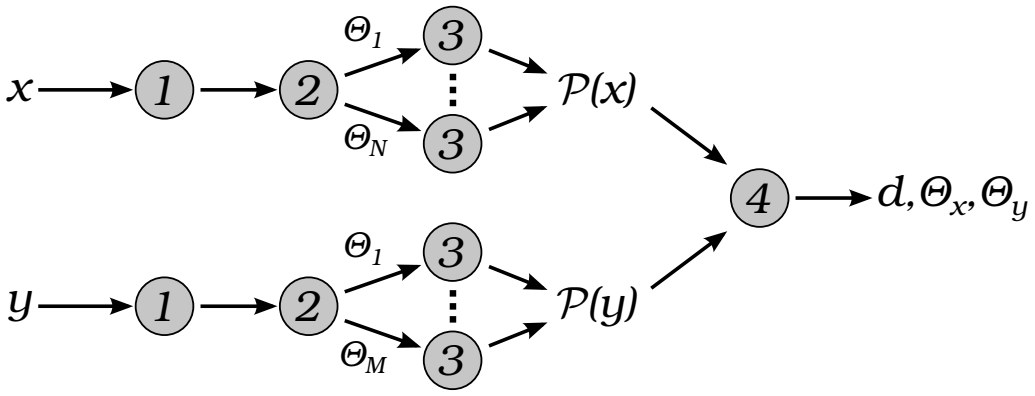

Figure 7: Data flow diagram for the patch similarity (distance) calculation between two given points x and y. The numbers in the nodes correspond to the steps of the outline.

0. **Build regular grid for interpolation**

   prm: $g > 0$, $r > 0$
   out: $\mathcal{G} := \{w_i\}$

   The parameter g controls the resolution of the regular grid, the set $\mathcal{G}$ contains coordinates of the grid nodes. This is a preprocessing step that normally should be done only once and before any patch distance computations. See Section 3.4 for details.

1. **Calculate structure tensor and shape-adaptive patch**

   in: ∇u, x
   prm: $r > 0$, $n_{ST} > 0$
   out: $T_u(x)$, $B_{T_u}(x)$

The parameter n_{ST} controls the number of iterations in the construction scheme for the affine covariant structure tensors and neighborhoods (Section 2.2). The construction of the structure tensors in the discrete setting is shown in Algorithm 1. The algorithm for collecting points belonging to the neighborhood $B_{T_u}(x)$ is explained in Section 3.2.

2. **Estimate dominant orientations**

   in: ∇u, $B_{T_u}(x)$, $T_u(x)$, x
   prm: $n_{bins} > 0$, $\sigma_{DO} > 0$, $\delta > 0$, $n_{DO} > 0$
   out: $\{\Theta_k\}$

   The parameter n_{bins} controls the number of bins in the gradient orientations histogram, the parameter σ_{DO} controls the intra-patch weighting and the parameter δ is the cut-off threshold for local maxima as in [18]. The parameter n_{DO} limits the maximum number of output orientations and $\{\Theta_k\}$ is the set of estimated dominant orientations. See Section 3.3 for details.

3. **Normalize shape-adaptive patch and interpolate it to the grid $\mathcal{G}$ (for each Θ_k)**

   in: u, $T_u(x)$, x, Θ_k, $\mathcal{G}$
   out: $\{\bar{c}_i\}_k$

   The output set contains interpolated color values. For every node w_i of the regular grid $\mathcal{G}$ we obtain the interpolated color value $\bar{c}_i$. See Section 3.4 for details.

   Note: after this step all interpolated candidate normalizations, corresponding to the same point x, are combined in a set $\mathcal{P}(x) := \{(\{\bar{c}_i\}_k, \Theta_k)\}$.

4. **Calculate patch distance between points x and y**

   in: $\mathcal{P}(x)$, $\mathcal{P}(y)$
   out: d, Θ_x, Θ_y

   Every candidate normalization from $\mathcal{P}(x)$ is compared with every candidate normalization from $\mathcal{P}(y)$, and a configuration that gives the smallest distance is returned.

Algorithm 1: Iterative scheme for computation of the affine covariant structure tensors.

Input: ∇u, $x \in \mathbb{R}^2$ *// gradient field, point of interest*
Parameters: r, n_{ST}, threshold *// n_{ST} is 60 by default*
Output: $T_u(x)$

$T_u^{(0)}(x) \leftarrow$ Equation (15) followed by (13) *// compute initial structure tensor*

for $0 < k < n_{ST}$ **do**
  $T_u^{(k)}(x) \leftarrow$ Equation (14) using $T_u^{(k-1)}(x)$ followed by (13) *// compute tensor at next iteration*
  if $||T_u^{(k)}(x) - T_u^{(k-1)}(x)||_F <$ threshold **then**
    break *// stop when change in $T_u(x)$ drops below threshold*

$T_u(x) \leftarrow T_u^{(k)}(x)$

3.2 Affine Covariant Regions

The numerical scheme for the construction of affine covariant structure tensors and neighborhoods is described in Section 2.2. In this section we explain an efficient way to determine a set of points that belong to a given affine covariant region.

Throughout this section we use a different notation for points in images to simplify explicit coordinates indication. Let $\bar{\mathbf{p}}$ be the central point of an affine covariant region $B_{T_u}(\bar{\mathbf{p}})$ and $\mathbf{p}$ be just any point on the image domain of u. A point $\mathbf{p}$ belongs to the affine covariant region $B_{T_u}(\bar{\mathbf{p}})$ if and only if

$$\langle T_u(\bar{\mathbf{p}})(\mathbf{p}-\bar{\mathbf{p}}),(\mathbf{p}-\bar{\mathbf{p}})\rangle \leq r^2.$$

The boundary of an affine covariant region $B_{T_u}(\bar{\mathbf{p}})$ is given by

$$\partial B_{T_u}(\bar{\mathbf{p}}) = \left\{\mathbf{p}:\ \langle T_u(\bar{\mathbf{p}})(\mathbf{p}-\bar{\mathbf{p}}),(\mathbf{p}-\bar{\mathbf{p}})\rangle = r^2\right\}. \tag{21}$$

3.2.1 Affine Covariant Regions in $\mathbb{R}^2$

Let us indicate components of $\bar{\mathbf{p}} \in \mathbb{R}^2$ and $\mathbf{p} \in \mathbb{R}^2$ as $\bar{\mathbf{p}} = [\bar{x},\bar{y}]^t$ and $\mathbf{p} = [x,y]^t$. From Equation (21) we can compute two points on $\partial B_{T_u}(\bar{\mathbf{p}})$ with the biggest and the smallest y coordinates. Let us introduce matrix coordinates for $T_u(\bar{\mathbf{p}})$, namely,

$$T_u(\bar{\mathbf{p}}) = \begin{bmatrix} T_{00} & T_{01} \\ T_{10} & T_{11} \end{bmatrix},$$

where $T_{01} = T_{10}$. By taking partial derivative of (21) with respect to x (and using the implicit function theorem) we obtain

$$T_{00}(x-\bar{x}) + T_{01}(y-\bar{y}) + T_{01}(x-\bar{x})\frac{\partial y}{\partial x} + T_{11}(y-\bar{y})\frac{\partial y}{\partial x} = 0.$$

Then by setting $\frac{\partial y}{\partial x} = 0$ we have

$$x - \bar{x} = -\frac{T_{01}}{T_{00}}(y-\bar{y}).$$

By substituting the above expression into (21) we obtain the y coordinates of the two extreme points (shown in Figure 8)

$$y = \bar{y} \mp r\left(T_{11} - \frac{T_{01}^2}{T_{00}}\right)^{-\frac{1}{2}}. \tag{22}$$

Let us denote them y^- and y^+. To collect the points $\mathbf{p}$ that belong to $B_{T_u}(\bar{\mathbf{p}})$ we traverse rows lying between the extreme points. For every row y (where $y^- \leq y \leq y^+$) we compute the x coordinates of two its intersections with the elliptical boundary (21) as

$$x(y) = \bar{x} - a(y-\bar{y}) \pm \sqrt{b(y-\bar{y})^2 + c}, \tag{23}$$

where the constants are given by

$$a = \frac{T_{01}}{T_{00}}, \qquad b = a^2 - \frac{T_{11}}{T_{00}} \quad \text{and} \quad c = \frac{r^2}{T_{00}}.$$

Then we collect all points $\mathbf{p}$ belonging to that row and located within the boundary of $B_{T_u}(\bar{\mathbf{p}})$. Figure 8 schematically illustrates this traversing.

At some points near high contrast edges, for small values of r, the numerical scheme may yield degenerate structure tensors which are not positive definite, or close to it. This occurs when either $T_u(\bar{\mathbf{p}})$ is close to the zero matrix (e.g. $\bar{\mathbf{p}}$ belongs to a region where u is constant) or $T_u(\bar{\mathbf{p}})$ is not zero but so is its determinant (e.g. all $\nabla u(\mathbf{p})$ in $B_{T_u}(\bar{\mathbf{p}})$ are almost the same, which results in a extremely elongated and narrow region). In both cases we output a region containing only the central point $\bar{\mathbf{p}}$ instead. To check that a matrix is strictly positive definite we check that $det(A) > 0$ and $A_{00} > 0$.

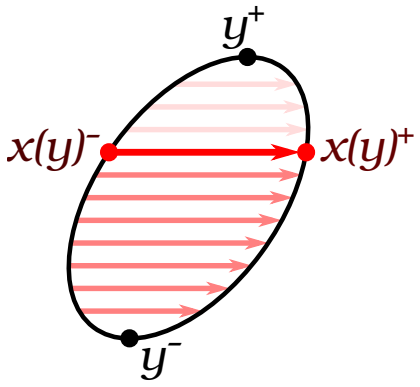

Figure 8: Schematic illustration of traversing of shape-adaptive elliptical patches in $\mathbb{R}^2$.

For the structure tensor, T_{00} is always greater than or equal to 0; thus it is sufficient to check only its determinant. The elongation of an elliptical region can be found using the ratio between eigenvalues of the corresponding structure tensor. As discussed in [18] (Section 4.1), since only the ratio is needed, there is no need to compute the eigenvalues themselves. Let λ_1 and λ_2 be eigenvalues of a structure tensor $T_u(\mathbf{p})$. Then its trace and determinant are

$$\begin{aligned} \mathrm{Tr}(T_u(\mathbf{p})) &= \lambda_1 + \lambda_2, \\ \mathrm{Det}(T_u(\mathbf{p})) &= \lambda_1 \lambda_2. \end{aligned}$$

Let $\alpha = \lambda_1/\lambda_2$ be the eigenvalue ratio. Then

$$\frac{\mathrm{Tr}(T_u(\mathbf{p}))^2}{\mathrm{Det}(T_u(\mathbf{p}))} = \frac{(\lambda_1+\lambda_2)^2}{\lambda_1\lambda_2} = \frac{(\alpha\lambda_2+\lambda_2)^2}{\alpha\lambda_2^2} = \frac{(\alpha+1)^2}{\alpha}.$$

Notice that the previous expression, $f(\alpha) := \frac{(\alpha+1)^2}{\alpha}$, is an increasing function of α (since $f'(\alpha) \geq 0$). Therefore, being $\bar{\alpha}$ a given eigenvalue ratio threshold, we recognize the structure tensor $T_u(\mathbf{p})$ at $\mathbf{p}$ as degenerate if $\alpha = \lambda_1/\lambda_2 > \bar{\alpha}$, which is equivalent to the condition

$$\frac{\mathrm{Tr}(T_u(\mathbf{p}))^2}{\mathrm{Det}(T_u(\mathbf{p}))} > \frac{(\bar{\alpha}+1)^2}{\bar{\alpha}}.$$

Throughout this work we set $\bar{\alpha} = 100$.

The whole procedure is summarized in Algorithm 2.

3.3 Dominant Orientations

In Section 2.3 it was shown that a local affine transformation can be estimated from two structure tensors, but only up to a rotation. To compensate for the missing rotation we estimate dominant orientations of the normalized patches using histograms of gradient orientations as in the SIFT keypoints and descriptors of [18]. Notice that there might be several dominant orientations and thus several equivalent options for the additional rotation. The procedure for estimating dominant orientations is described below and summarized in Algorithm 3 for the case of images in $\mathbb{R}^2$. An extension for the $\mathbb{R}^3$ case can be found in [7].

3.3.1 Dominant Orientations in $\mathbb{R}^2$

Recall that in order to compute affine covariant structure tensors we first compute the gradient field ∇u. The same gradient vectors, when transformed appropriately, can be used to estimate the

Algorithm 2: Assembly of affine covariant regions in $\mathbb{R}^2$.

Input: $T_u(\bar{\mathbf{p}})$, $\Omega_u \subset \mathbb{R}^2$, $\bar{\mathbf{p}} = [\bar{x}, \bar{y}]^t$ *// structure tensor, domain of u, point of interest.*
Parameters: r, $\bar{\alpha}$
Output: $B_{T_u}(\bar{\mathbf{p}})$

$T := T_u(\bar{\mathbf{p}})$ *// alias*
$B_{T_u}(\bar{\mathbf{p}}) \leftarrow \{\}$

// Ensure tensor is not degenerate
if $det(T) \leq 0$ **or** $\frac{trace(T)^2}{det(T)} > \frac{(\bar{\alpha}+1)^2}{\bar{\alpha}}$ **then**
  $B_{T_u}(\bar{\mathbf{p}}) \leftarrow B_{T_u}(\bar{\mathbf{p}}) \cup \bar{\mathbf{p}}$
  Stop

// Compute offsets of extreme points from the center
$oy \leftarrow r\left(T_{11} - \frac{T_{01}^2}{T_{00}}\right)^{-\frac{1}{2}}$ *// Equation (22)*
// Compute auxiliary constants
$a \leftarrow \frac{T_{01}}{T_{00}}$
$b \leftarrow a^2 - \frac{T_{11}}{T_{00}}$
$c \leftarrow \frac{r^2}{T_{00}}$
// Traverse row by row
for $\bar{y} - \lfloor oy \rfloor \leq y \leq \bar{y} + \lfloor oy \rfloor$ **do**
  Skip y, if it is outside of Ω_u *// $y \in \mathbb{R}$*
  $x^- \leftarrow \bar{x} - a(y - \bar{y}) - \sqrt{b(y - \bar{y})^2 + c}$ *// left intersection*
  $x^+ \leftarrow \bar{x} - a(y - \bar{y}) + \sqrt{b(y - \bar{y})^2 + c}$ *// right intersection*
  for $\lceil x^- \rceil \leq x \leq \lfloor x^+ \rfloor$ **do**
    $B_{T_u}(\bar{\mathbf{p}}) \leftarrow B_{T_u}(\bar{\mathbf{p}}) \cup \mathbf{p}$ *// $\mathbf{p} = [x, y]^t$*

dominant orientation within normalized patches. Let us simplify the presentation and denote by $\hat{T}(x) := T_u(x)^{\frac{1}{2}}$ the transformation that normalizes an elliptical patch at x to a disk of radius r. Of course, this normalization does not yet take any additional rotation into account. It can be shown that the suitable transformation to apply to the gradient vectors is $(\hat{T}(x)^{-1})^t$.

For every point $y \in B_{T_u}(x)$ we transform the corresponding gradient vector $\nabla u(y)$, compute its direction and magnitude and use them to fill-in the circular histogram of orientations. Note that the magnitude value is additionally weighted by the anisotropic Gaussian intra-patch weight, depending on the distance to the center of the patch, given by

$$\omega_x(y) = exp\left(-\frac{\langle T_u(x)(y-x), (y-x)\rangle}{2\sigma_{DO}^2}\right). \tag{24}$$

The resulting value is then distributed linearly between the nearest bins of the histogram in proportion to the distance to these bins. In that histogram we find the global maximum and all other local maxima which are big enough. In order to improve the accuracy of the estimation, we fit a quadratic function to the histogram values around every such maximum and take its argmax as orientation Θ. Up to n_{DO} highest peaks in the histogram are considered as dominant orientations and gathered into the output set $\{\Theta_k\}$. According to our observations, most commonly the number of dominant orientations does not exceed two.

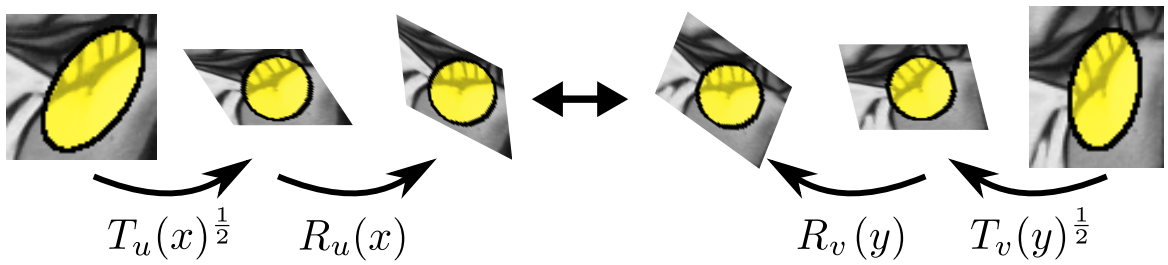

Figure 9: Registration of two shape-adaptive patches for comparison by normalization and additional rotation.

The additional rotation $R_u(x)$ can be computed as

$$R_u(x) := R(\Theta) = \begin{bmatrix} \cos\Theta & \sin\Theta \\ -\sin\Theta & \cos\Theta \end{bmatrix}, \tag{25}$$

for Θ being a given dominant orientation.

In our experiments we set the number of bins in the histogram $n_{bins} = 72$, the cut-off threshold $\delta = 45\%$ (see [18]), the weighting parameter $\sigma_{DO} = 0.2$ and the limit on the number of orientations $n_{DO} = 3$.

3.4 Normalization and Interpolation

In this section we delve into the details of registration of elliptical patches for comparison. In our specific setting there are two approaches to registration: either one of the elliptical patches is transformed using (18), or both elliptical patches are normalized to circular patches as shown in Figure 9. The first approach can be exploited for image synthesis, for example, in exemplar-based inpainting and denoising. Meanwhile the second option is particularly suitable for patch comparison, because it allows intermediate data caching. That is, normalized patches can be computed once and stored in memory, then every normalized patch can participate in multiple comparison operations without any additional transformations required.

In order to normalize an elliptical patch we need a proper transformation. As was shown previously, an initial transformation is obtained directly from a corresponding structure tensor. This preliminary normalization is used to estimate dominant orientations within the normalized patch. Several dominant orientation might be estimated for every patch; therefore, there can be several options for the normalizing transformations. Let us denote them here by $A_k(x) = R(\Theta_k)T_u(x)^{\frac{1}{2}}$, where $R(\Theta_k)$ is an additional rotation given by (25). Obviously, all the options are equivalent and we should consider each of them.

Since in practice we deal with digital images which are discrete, after normalization every patch turns into a set of scattered points (as illustrated by the blue dots in Figure 10, right). In principle, to compare two normalized patches we could interpolate one set of scattered points directly to another; however, we propose to instead interpolate these sets to a regular grid $\mathcal{G}$ (shown by the red asterisks on the right side of Figure 10). The usage of the intermediate regular grid allows us to precompute all normalized patches and store them in memory. If we store all possible normalized versions for every elliptical patch, together with their corresponding candidate transformations $A_k(x)$, the calculation of the patch distance between two points boils down to several sums of squared differences. We consider the resolution of the regular grid to be constant throughout a single run of any experiment; thus, it can be built in the preprocessing phase. In terms of data structure, the grid is represented by a set of real-valued coordinates of its nodes. Any normalized patch is contained within a circle of radius r; therefore, the regular grid is built in such a way that its nodes evenly cover this circle

Algorithm 3: Estimation of dominant orientations in $\mathbb{R}^2$.

Input: ∇u, $B_{T_u}(x)$, $T_u(x)$, $x \in \mathbb{R}^2$ *// gradient field, neighbourhood, structure tensor, point*
Parameters: n_{bins}, σ_{DO}, δ, n_{DO}
Output: $\{\Theta_k\}$

$A \leftarrow ((T_u(x)^{\frac{1}{2}})^{-1})^t$ *// transformation for gradient vectors*
$H \leftarrow$ array of $(n_{bins}+2)$ elements, all set to 0 *// histogram*

// Fill-in histogram H
foreach $y \in B_{T_u}(x)$ **do**
  $\widetilde{\nabla u}(y) \leftarrow A\nabla u(y)$
  $\alpha \leftarrow exp\left(-\frac{\langle T_u(x)(y-x),(y-x)\rangle}{2\sigma_{DO}^2}\right)\left\|\widetilde{\nabla u}(y)\right\|$
  $\gamma \leftarrow$ angle between $\widetilde{\nabla u}(y)$ and X axis in range $[0, 2\pi]$
  $p \leftarrow \frac{\gamma\ n_{bins}}{2\pi}$ *// real-valued position in the histogram*
  $i \leftarrow \lfloor p - 0.5 \rfloor + 1$ *// index of the left closest bin*
  $d \leftarrow p - i + 0.5$ *// distance to the center of i-th bin*
  $H_i \leftarrow H_i + (1-d)\ \alpha$
  $H_{i+1} \leftarrow H_{i+1} + d\ \alpha$

// Merge boundary values, because H should be circular
$H_0 \leftarrow H_0 + H_{-2}$ *// H_{-2} is the penultimate element*
$H_{-1} \leftarrow H_{-1} + H_1$ *// H_{-1} is the last element*
$H_{-2} \leftarrow H_0$
$H_1 \leftarrow H_{-1}$

Smooth H by convolving it six times with $\left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right]$ kernel

// Collect peaks that are big enough
$C \leftarrow \{\}$ *// set of candidate orientations*
for $1 \le i \le n_{bins}$ **do**
  if $H_i \ge \delta\ max(H)$ **and** $H_i > H_{i-1}$ **and** $H_i > H_{i+1}$ **then**
    $C \leftarrow C \cup (H_i, i)$ *// append tuple (H_i, i) to C*

Sort C in descending order by histogram values

// Refine and collect dominant orientations
$\Theta \leftarrow \{\}$
for $0 \le j < min(n_{DO},\ size(C))$ **do**
  $i \leftarrow$ second element from tuple C_j
  $\Theta \leftarrow \Theta \cup \frac{2\pi}{n_{bins}}\left(i + 0.5 + 0.5\frac{H_{i-1}-H_{i+1}}{H_{i-1}-2H_i+H_{i+1}}\right)$ *// refine by fitting a parabola*

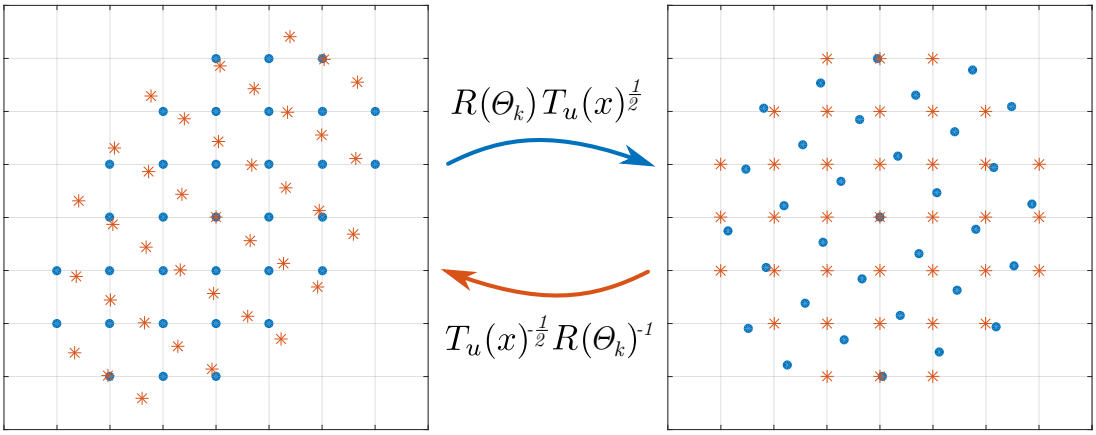

Figure 10: Left to right (and in blue): transformation of an elliptical patch onto the normalized circular patch; right to left (and in red): mapping of the regular grid $\mathcal{G}$ onto an elliptical patch. Points of an elliptical patch have color values associated with them and are shown as blue dots. Nodes of $\mathcal{G}$ are shown as red asterisks. These are the locations where color values have to be interpolated.

with a given resolution. It is convenient to specify the resolution of the grid by the number of nodes that should fit along the diameter of the circle. We denote this free parameter of the method by g. Algorithm 4 illustrates the construction of the regular grid.

A straightforward approach to interpolation would be to transform points of an elliptical patch by $R(\Theta_k)T_u(x)^{\frac{1}{2}}$ and then apply some kernel-based estimator, for instance, the simple Nadaraya-Watson estimator [21, 34] with Gaussian kernel. This approach is illustrated in Figure 10, where blue dots are the ones belonging to an elliptical patch and having color values associated with them, red asterisks are the nodes of the regular grid $\mathcal{G}$ for which the color values have to be computed, and the transformation brings color values from the elliptical patch on the left side into the normalized patch on the right side. Immediately it can be seen that there is another approach. Using the inverse transformation $T_u(x)^{-\frac{1}{2}}R(\Theta_k)^{-1}$ coordinates of the grid nodes can be transferred onto the elliptical patch. In this case the known color values are located on the regular grid of a discrete image, while points to be interpolated are off the grid. Then the computationally less expensive bilinear interpolation can be applied. Algorithm 5 describes this approach to interpolation of elliptical patches in a more formal way.

Algorithm 4: Construction of a regular grid in $\mathbb{R}^2$.

Parameters: g, r
Output: $\mathcal{G}$

$s \leftarrow \frac{2r}{g}$ *// grid step*
$\mathcal{G} \leftarrow \{\}$ *// set of grid nodes*
for $0 \leq i < g$ **do**
  for $0 \leq j < g$ **do**
    $\omega \leftarrow [j\,s + 0.5\,s - r,\ i\,s + 0.5\,s - r]^t$ *// $\omega \in \mathbb{R}^2$*
    if $\|\omega\|_2^2 \leq r^2$ **then**
      $\mathcal{G} \leftarrow \mathcal{G} \cup \omega$

Algorithm 5: Interpolation of normalized patches to a regular grid.

Input: u, $T_u(x)$, $x \in \mathbb{R}^2$, Θ_k, $\mathcal{G}$ *// image, structure tensor, point, angle, set of grid nodes*

Output: $\bar{C} := \{\bar{c}_i\}_k$

$$R \leftarrow \begin{bmatrix} cos\,\Theta_k & sin\,\Theta_k \\ -sin\,\Theta_k & cos\,\Theta_k \end{bmatrix}$$ *// additional rotation*

$A \leftarrow T_u(x)^{-\frac{1}{2}}\ R^{-1}$

$\bar{C} \leftarrow \{\}$

foreach $\omega \in \mathcal{G}$ **do**

$\bar{\omega} = A\omega + x$

$y = \lfloor \bar{\omega} \rfloor$

$\Delta y = \bar{\omega} - y$

$$\bar{c} = u\,(y)\,(1-\Delta y_1)(1-\Delta y_2) + u\left(y + \begin{bmatrix}1\\0\end{bmatrix}\right)\Delta y_1(1-\Delta y_2) +$$
$$+u\left(y + \begin{bmatrix}0\\1\end{bmatrix}\right)(1-\Delta y_1)\Delta y_2 + u\left(y + \begin{bmatrix}1\\1\end{bmatrix}\right)\Delta y_1 \Delta y_2$$ *// bilinear interpolation*

$\bar{C} \leftarrow \bar{C} \cup \bar{c}$

4 Examples of Similarity Maps

The source code provided together with this paper implements the computation of the affine covariant structure tensors, affine covariant regions, and affine invariant patch similarity. The implementation is highly modular; therefore, it can be either used through the provided console applications, or included as an integral part of some other project. In this section we demonstrate a few similarity maps obtained with the provided implementation.

In the examples shown in Figure 11 each similarity map depicts similarity values between a given point (let's say x) and all the points y in an image. In other words, such a map identifies the locations which are considered to be similar to a selected point of interest. For visualization purposes, the distances $\mathcal{D}^{\mathrm{a}}(t,x,y)$ are converted into similarities and color-coded using the map

$$c(x,y,t) = 255 \exp\left(-\frac{(\mathcal{D}^{\mathrm{a}}(t,x,y) - \mathcal{D}^{\mathrm{a}}_{min})^2}{2\sigma^2}\right), \tag{26}$$

where x and y are two pixels, $t > 0$, $\sigma = \frac{\mathcal{D}^{\mathrm{a}}_{max} - \mathcal{D}^{\mathrm{a}}_{min}}{\gamma}$, $\mathcal{D}^{\mathrm{a}}_{max}$ and $\mathcal{D}^{\mathrm{a}}_{min}$ are the maximum and minimum patch distance values, respectively, and $\gamma > 0$ is a visualization parameter. Notice that higher values of $c(x,y,t)$ correspond to more similar patches around x and y.

Finally, Figure 12 illustrates the multiscale nature of the proposed affine invariant patch similarity measure (20). The selection of a bigger t (or r) in (20) results in bigger patches used in the patch comparison. Figure 12 shows three similarity maps $c(x,y,t)$ for $x = (89, 178)$ (i.e., the point shown in green in Figure 11), all y in the image domain and, from left to right, with increasing scale value t. The scale is set as $t = r/\hat{t}$, where radius $r = 300$, and $\hat{t} = 6$, $\hat{t} = 3$, $\hat{t} = 1$ (from left to right, respectively). Let us notice that, as the patch size increases, the patch comparison allows to identify similar texture content at a bigger scale. A more detailed study of the effects of r and t parameters can be found in [9].

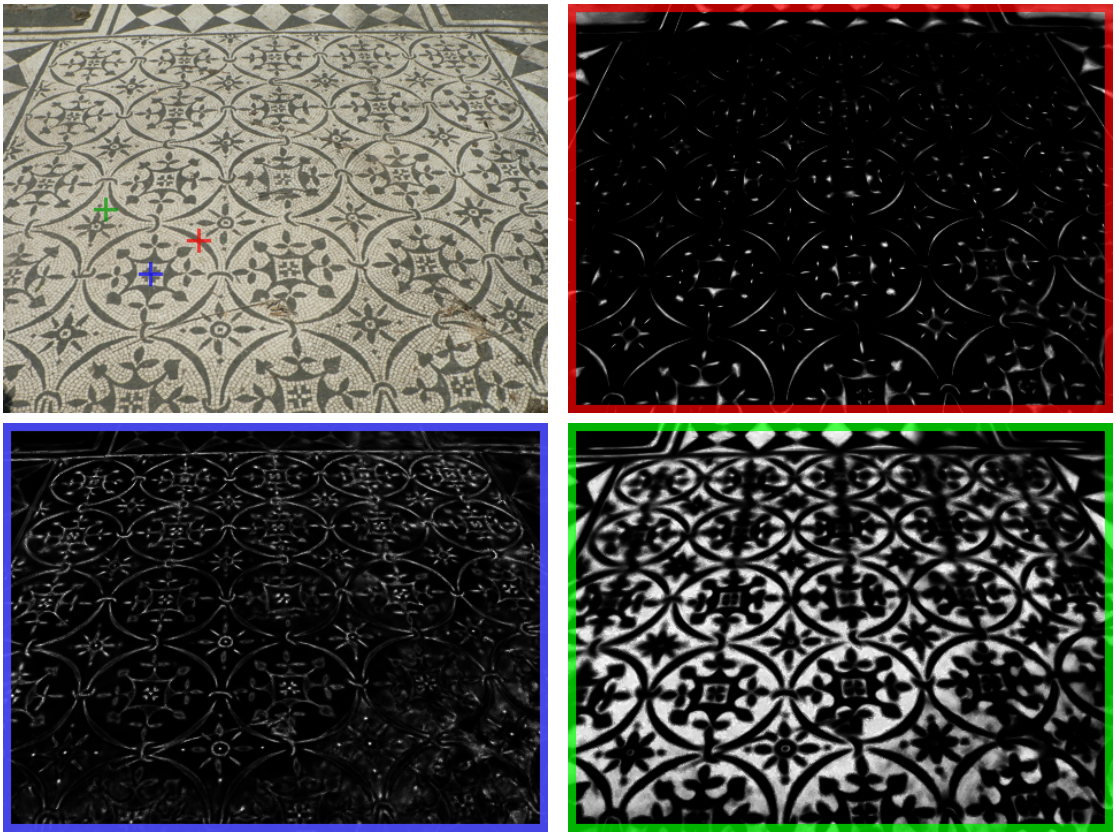

Figure 11: Top-left is the original image with three points x of interest shown in different colors: red is $x = (170, 205)$, green is $x = (89, 178)$, blue is $x = (128, 234)$. The remaining three images show similarity maps (using Equation (26)) computed for each point of interest x and all y in the image domain. The map corresponding to each x is highlighted with the same color.

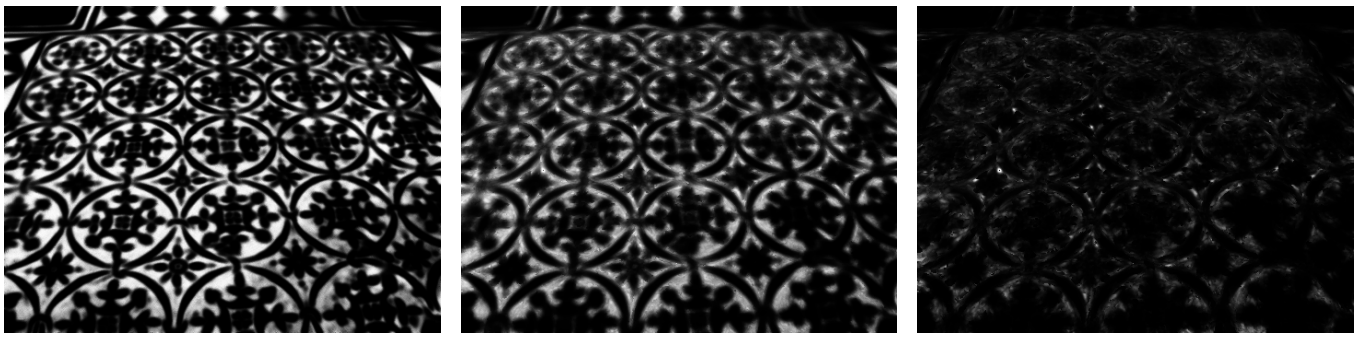

Figure 12: Similarity maps $c(x, y, t)$ for the point $x = (89, 178)$ (shown in green in Figure 11) computed at different increasing scales t. The scale is set as $t = r/\hat{t}$, where radius $r = 300$, and $\hat{t} = 6$, $\hat{t} = 3$, $\hat{t} = 1$ (from left to right, respectively). In other words, from left to right the increasing patch size used in the patch comparison allows to identify similar texture content at a bigger scale.

5 Conclusions

In this work we have presented a detailed description and implementation of affine invariant patch comparison by means of the similarity measure proposed in [9, 7]. We have briefly summarized the theoretical premises of this similarity measure which exploits the affine covariant structure tensors for defining and transforming shape-adaptive patches. For the sake of completeness we have also recalled the particular definition and the iterative computational scheme for the affine covariant structure

tensors and regions. In the rest of the paper we have documented at a high level the accompanying open source implementation of the discussed affine invariant patch comparison.

Acknowledgements

The authors acknowledge partial support by MINECO/FEDER UE project, reference TIN2015-70410-C2-1-R, by GRC reference 2014 SGR 1301 Generalitat de Catalunya, and by H2020-MSCA-RISE-2017 project, reference 777826 NoMADS.

Image Credits

 Digital image courtesy of the Getty's Open Content Program[2]

 London Mosaic[3] (for non-commerical use only)

 Middlebury Stereo Datasets 2003[4] [29]

 Derived from an image[5] licensed under Creative Commons BY-SA 2.0 license

 "Affine Covariant Regions" dataset[6]

 "Affine Covariant Regions" dataset[6]

 Meredith Cutler[7]

All other images by the authors.

References

[1] P. Arias, G. Facciolo, V. Caselles, and G. Sapiro, *A variational framework for exemplar-based image inpainting*, International Journal of Computer Vision, 93 (2011), pp. 319–347. https://doi.org/10.1007/s11263-010-0418-7.

[2] C. Ballester, F. Calderero, V. Caselles, and G. Facciolo, *Multiscale analysis of similarities between images on Riemannian manifolds*, SIAM Journal Multiscale Modeling and Simulation, 12 (2014), pp. 616–649. https://doi.org/10.1137/130926833.

[3] T. Brox, R. Boomgaard, F. Lauze, J. Weijer, J. Weickert, P. Mrázek, and P. Kornprobst, *Adaptive structure tensors and their applications*, Visualization and Processing of Tensor Fields, (2006), pp. 17–47. https://doi.org/10.1007/3-540-31272-2_2.

[4] T. Brox, J. Weickert, B. Burgeth, and P. Mrázek, *Nonlinear structure tensors*, Image and Vision Computing, 24 (2006), pp. 41–55. https://doi.org/10.1016/j.imavis.2005.09.010.

[2] http://www.getty.edu/about/whatwedo/opencontent.html
[3] https://www.londonmosaic.com
[4] http://vision.middlebury.edu/stereo/data/scenes2003/
[5] https://commons.wikimedia.org/wiki/File:Roman_mosaic_in_the_Jewry_Wall_Museum.jpg
[6] http://www.robots.ox.ac.uk/~vgg/research/affine/
[7] http://www.meredithcutler.com/

[5] A. Buades, B. Coll, and J.-M. Morel, *A non-local algorithm for image denoising*, in IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, June 2005, pp. 60–65. https://doi.org/10.1109/CVPR.2005.38.

[6] H. Farid and E.P Simoncelli, *Optimally rotation-equivariant directional derivative kernels*, in International Conference on Computer Analysis of Images and Patterns, Springer, 1997, pp. 207–214. https://doi.org/10.1007/3-540-63460-6_119.

[7] V. Fedorov, *Affine Invariant Image Comparison and Its Applications*, PhD. Thesis, 2016.

[8] V. Fedorov, P. Arias, G. Facciolo, and C. Ballester, *Affine invariant self-similarity for exemplar-based inpainting*, in Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, 2016, pp. 48–58. https://doi.org/10.5220/0005728100480058.

[9] V. Fedorov, P. Arias, R. Sadek, G. Facciolo, and C. Ballester, *Linear multiscale analysis of similarities between images on Riemannian manifolds: Practical formula and affine covariant metrics*, SIAM Journal on Imaging Sciences, 8 (2015), pp. 2021–2069. https://doi.org/10.1137/141000002.

[10] V. Fedorov and C. Ballester, *Affine non-local means image denoising*, IEEE Transactions on Image Processing, 26 (2017), pp. 2137–2148. https://doi.org/10.1109/TIP.2017.2681421.

[11] H. Foroosh, J. B. Zerubia, and M. Berthod, *Extension of phase correlation to subpixel registration*, IEEE Transactions on Image Processing, 11 (2002), pp. 188–200. https://doi.org/10.1109/83.988953.

[12] J. Garding and T. Lindeberg, *Direct estimation of local surface shape in a fixating binocular vision system*, in European Conference on Computer Vision, Jan-Olof Eklundh, ed., vol. 800 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 1994, pp. 365–376. https://doi.org/10.1007/3-540-57956-7_40.

[13] T. Kadir, A. Zisserman, and M. Brady, *An affine invariant salient region detector*, in European Conference on Computer Vision, Springer, 2004, pp. 228–241. https://doi.org/10.1007/978-3-540-24670-1_18.

[14] A. Kheradmand and P. Milanfar, *A general framework for regularized, similarity-based image restoration*, IEEE Transactions on Image Processing, 23 (2014), pp. 5136–5151. https://doi.org/10.1109/TIP.2014.2362059.

[15] R. Kimmel, R. Malladi, and N. Sochen, *Images as embedded maps and minimal surfaces: movies, color, texture, and volumetric medical images*, International Journal of Computer Vision, 39 (2000), pp. 111–129. https://doi.org/10.1023/A:1008171026419.

[16] M. Lebrun, A. Buades, and J. M. Morel, *A nonlocal Bayesian image denoising algorithm*, SIAM Journal on Imaging Sciences, 6 (2013), pp. 1665–1688. https://doi.org/10.1137/120874989.

[17] J.P. Lewis, *Fast normalized cross-correlation*, Vision Interface, 10 (1995), pp. 120–123.

[18] D. Lowe, *Distinctive image features from scale-invariant keypoints*, International Journal of Computer Vision, 60 (2004), pp. 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94.

[19] J. Matas, O. Chum, M. Urban, and T. Pajdla, *Robust wide-baseline stereo from maximally stable extremal regions*, Image and Vision computing, 22 (2004), pp. 761–767. https://doi.org/10.1016/j.imavis.2004.02.006.

[20] K. Mikolajczyk and C. Schmid, *Scale & affine invariant interest point detectors*, International journal of computer vision, 60 (2004), pp. 63–86. https://doi.org/10.1023/B:VISI.0000027790.02288.f2.

[21] E. A. Nadaraya, *On estimating regression*, Theory of Probability & Its Applications, 9 (1964), pp. 141–142. http://dx.doi.org/10.1137/1109020.

[22] M. Oliver, *Scene Understanding from Image and Video: Segmentation, Depth Configuration and Inpainting*, PhD. Thesis, 2018.

[23] M. Oliver, G. Haro, V. Fedorov, and C. Ballester, *L1 patch-based image partitioning into homogeneous textured regions*, in Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2018, pp. 1558–1562. https://doi.org/10.1109/ICASSP.2018.8462594.

[24] G. Peyré, *Manifold models for signals and images*, Computer Vision and Image Understanding, 113 (2009), pp. 249–260. https://doi.org/10.1016/j.cviu.2008.09.003.

[25] N. Pierazzo, M. Lebrun, M. E. Rais, J. M. Morel, and G. Facciolo, *Non-local dual image denoising*, in Proceedings of IEEE International Conference on Image Processing (ICIP), 2014. https://doi.org/10.1109/ICIP.2014.7025163.

[26] L. Pizarro, P. Mrázek, S. Didas, S. Grewenig, and J. Weickert, *Generalised nonlocal image smoothing*, International Journal of Computer Vision, 90 (2010), pp. 62–87. https://doi.org/10.1007/s11263-010-0337-7.

[27] M. Protter, M. Elad, H. Takeda, and P. Milanfar, *Generalizing the non-local-means to super-resolution reconstruction*, IEEE Transactions on Image Processing, 18 (2009), pp. 36–51. https://doi.org/10.1109/TIP.2008.2008067.

[28] H. Scharr, *Optimal filters for extended optical flow*, in Complex Motion, B. Jähne, R. Mester, E. Barth, and H. Scharr, eds., Berlin, Heidelberg, 2007, Springer Berlin Heidelberg, pp. 14–29. http://doi.org/10.1007/978-3-540-69866-1_2.

[29] D. Scharstein and R. Szeliski, *High-accuracy stereo depth maps using structured light*, in Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, June 2003, pp. I–I. http://dx.doi.org/10.1109/CVPR.2003.1211354.

[30] T. Tuytelaars and K. Mikolajczyk, *Local invariant feature detectors: A survey*, Foundations and Trends in Computer Graphics and Vision, 3 (2008), pp. 177–280. https://doi.org/10.1561/0600000017.

[31] T. Tuytelaars and L. Van Gool, *Matching widely separated views based on affine invariant regions*, International Journal of Computer Vision, 59 (2004), pp. 61–85. https://doi.org/10.1023/B:VISI.0000020671.28016.e8.

[32] P. Vitoria, V. Fedorov, and C. Ballester, *Spatio-temporal tube segmentation through a video metrics-based patch similarity measure*, in Proceedings of the Irish Machine Vision and Image Processing Conference (IMVIP), 2017.

[33] L. Wang, Y. Zhang, and J. Feng, *On the Euclidean distance of images*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 27 (2005), pp. 1334–1339. https://doi.org/10.1109/TPAMI.2005.165.

[34] G.S. Watson, *Smooth regression analysis*, Sankhya: The Indian Journal of Statistics, Series A (1961-2002), 26 (1964), pp. 359–372. http://www.jstor.org/stable/25049340.

[35] J. Weickert, *Anisotropic diffusion in image processing*, vol. 1, Teubner Stuttgart, 1998.

[36] ——, *Coherence-enhancing diffusion filtering*, International Journal of Computer Vision, 31 (1999), pp. 111–127. https://doi.org/10.1023/A:1008009714131.