ASIFT: An Algorithm for Fully Affine Invariant Comparison

Guoshen Yu, Jean-Michel Morel

published • 2011-02-24 → BibTeX
reference • Guoshen Yu, and Jean-Michel Morel, *ASIFT: An Algorithm for Fully Affine Invariant Comparison*, Image Processing On Line, 1 (2011). http://dx.doi.org/10.5201/ipol.2011.my-asift

Communicated by Guillermo Sapiro
Demo edited by Nicolas Limare

- Guoshen Yu yu@cmap.polytechnique.fr, CMAP, École Polytechnique
- Jean-Michel Morel morel@cmla.ens-cachan.fr, CMLA, ENS Cachan

Content

Additional materials

- Slides : one-hour version, 20 minutes version, videos (magazine, facade)
- Poster
- Dataset: An image dataset for systematic evaluation of robustness to absolute and transition tilt of the image matching algorithms is available : zip tar/gz.

Source code and executable

This IPOL implementation reproduces exactly the same results as in the online demo.
2010/01/09: updated to version 2.1.2, minor bug corrected.

- source code: tar/gz zip
  The standalone C++ program is compilable on Linux, Mac OSX and Windows systems.
- executable: zip
  Executable versions are available for Windows.

Note from the editor: The source code was updated on April 30, 2011 to resolve a possible copyright issue. Only the SVD functions were replaced.

Overview

If a physical object has a smooth or piecewise smooth boundary, its images obtained by cameras in varying positions undergo smooth apparent deformations. These deformations are locally well approximated by affine transforms of the image plane.

In consequence the solid object recognition problem has often been led back to the computation of **affine invariant** image local features. Such invariant features could be obtained by normalization methods, but no fully affine normalization method exists for the time being. Yet as shown in (see ref. 3) the similarity invariance (invariance to translation, rotation, and zoom) is dealt with rigorously by the SIFT method (see ref. 5). By simulating on both images zooms out and by normalizing translation and rotation, the SIFT method succeeds in being fully invariant to four out of the six parameters of an affine transform.

The method illustrated and demonstrated in this work, **Affine-SIFT (ASIFT)**, simulates a set of sample views of the initial images, obtainable by varying the two camera axis orientation parameters, namely the latitude and the longitude angles, which are not treated by the SIFT method. Then it applies the SIFT

method itself to all images thus generated. Thus, ASIFT covers effectively all six parameters of the affine transform. The method is mathematically proved in (see ref. 1) to be fully affine invariant. And, against any prognosis, simulating a large enough set of sample views depending on the two camera orientation parameters is feasible with no dramatic computational load. The main anamorphosis (deformation) from an image to another caused by applying affine transforms can be measured by the **transition tilt**, a new geometric concept introduced in (see ref. 1) and explained below.

While state-of-the-art methods before ASIFT hardly exceeded transition tilts of 2 (SIFT), 2.5 (Harris-Affine and Hessian-Affine (see ref. 6) and 10 (see ref. 4), ASIFT can handle transition tilts up 32 and higher. MSER can actually deal with transition tilts as high as 10 only when both objects are taken roughly at the same distance. Indeed, contrarily to SIFT, MSER is not scale invariant, because it does not simulate the blur due to an increasing distance to the object. The affine transforms considered in the celebrated comparison paper (see ref. 7) do not have high transition tilts as those that can be handled by ASIFT. As shown by the experiment archive, most scenes with negligible or moderate camera view angle change that match with ASIFT also match with SIFT (with usually fewer matching points). But, when the view angle change becomes important, SIFT and other methods fail while ASIFT continues to work, as we shall see in the examples below.

Disclaimer

The present work publishes only the ASIFT algorithm as described below. It does not publish the SIFT and ORSA subroutines which are called by the ASIFT code. SIFT and ORSA may be later updated or replaced by other subroutines.

References

1. J.M. Morel and G.Yu, ASIFT, *A new framework for fully affine invariant image comparison*. SIAM Journal on Imaging Sciences, 2(2):438-469 (2009). DOI:10.1137/080732730 preprint
2. G. Yu and J.M. Morel, *A Fully Affine Invariant Image Comparison Method*. Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Taipei (2009). DOI:10.1109/ICASSP.2009.4959904 preprint
3. J.M. Morel and G.Yu, *Is SIFT scale invariant?* Inverse Problems and Imaging, vol 5, no. 1 (2011). DOI:10.3934/ipi.2011.5.115 preprint
4. J. Matas, O. Chum, M. Urba, and T. Pajdla. *Robust wide baseline stereo from maximally stable extremal regions*. Proc. of British Machine Vision Conference, pages 384-396 (2002). DOI:10.1016/j.imavis.2004.02.006
5. David G. Lowe. *Distinctive image features from scale-invariant keypoints*. International Journal of Computer Vision, 60, 2, pp. 91-110 (2004). DOI:10.1023/B:VISI.0000029664.99615.94 preprint
6. K. Mikolajczyk, and C. Schmid. *An affine invariant interest point detector*. In Proc. European Conf. Computer Vision, Vol. 2350, pp. 128-142 (2002).
7. K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir and L. Van Gool, *A comparison of affine region detectors*. International Journal of Computer Vision, 65, 1-2 (2005). DOI:10.1007/s11263-005-3848-x
8. L. Moisan, B. Stival. *A probabilistic criterion to detect rigid point matches between two images and estimate the fundamental matrix*. International Journal of Computer Vision, vol. 57:3, pp. 201-218 (2004). DOI:10.1023/B:VISI.0000013094.38752.54 preprint

Affine camera model

Image acquisition model

Figure 1. The projective camera model.

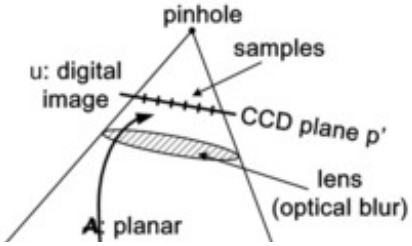
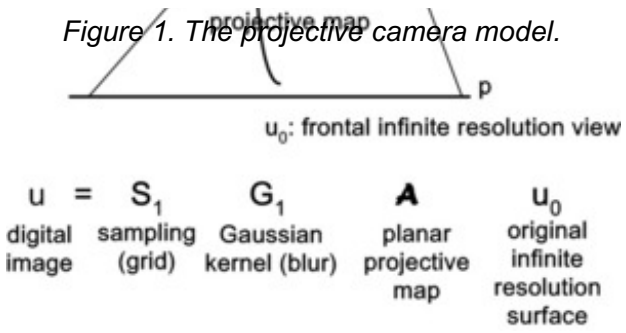


Figure 1. The projective camera model.



As illustrated by the camera model in Figure 1, digital image acquisition of a flat object can be described as

$$\mathbf{u} = \mathbf{S}_1 G_1 A \mathcal{T} u_0$$

where $\mathbf{u}$ is a digital image and u_0 is an (ideal) infinite resolution frontal view of the flat object. The parameters $\mathcal{T}$ and A are respectively a plane translation and a planar projective map due to the camera motion. G_1] is a Gaussian convolution modeling the optical blur, and $\mathbf{S}_1$ is the standard sampling operator on a regular grid with mesh 1. The Gaussian kernel is assumed to be broad enough to ensure no aliasing by the 1-sampling, therefore with a Shannon-Whittaker interpolation I, one can recover the continuous image from its discrete version: $I\mathbf{S}_1 G_1 A \mathcal{T} u_0 = G_1 A \mathcal{T} u_0$. $\mathbf{S}_1$ will be thus omitted in the following.

Affine local approximation

Figure 2. Affine local approximation.

We shall proceed to a further simplification of the above model, by reducing A to an affine map. Figure 2 shows one of the first perspectively correct Renaissance paintings by Paolo Uccello. The perspective on the ground is strongly projective: the rectangular pavement of the room becomes a trapezoid. However, each tile on the pavement is almost a parallelogram. This illustrates the local tangency of perspective deformations to affine maps. Indeed, by the first order Taylor formula, any planar smooth deformation can be approximated around each point by an affine map. The perspective deformation of a plane object induced by a camera motion is a planar homographic transform, which is smooth, and therefore locally tangent to affine transforms $u(x,y) \rightarrow u(ax+by+e, cx+dy+f)$ in each image region.

Affine map decomposition, with geometric interpretation as a camera motion

Any affine map A with strictly positive determinant which is not a similarity has a unique decomposition

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix} = H_\lambda R_1(\psi) T_t R_2(\phi) = \lambda \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} \begin{bmatrix} t & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}$$

where $\lambda > 0$, λ t is the determinant of A, R_i are rotations, $\varphi \in [0, \pi)$, and T_t is a tilt, namely a diagonal matrix with first eigenvalue $t > 1$ and the second one equal to 1.

Figure 3. Geometric interpretation of affine decomposition.

Figure 3. Geometric interpretation of affine decomposition.

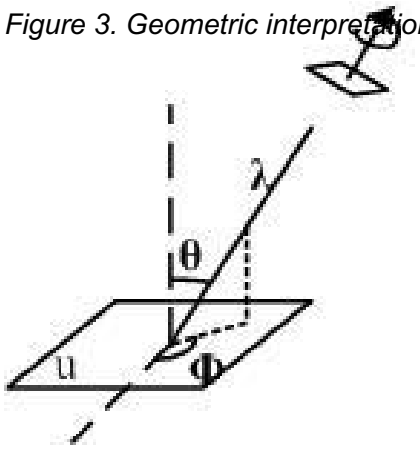


Figure 3 shows a camera motion interpretation of the affine decomposition: φ and $\theta = \arccos 1/t$ are the viewpoint angles, ψ parameterizes the camera spin and λ corresponds to the zoom. The camera (the small parallelogram on the top-right) is assumed to stay far away from the image u and starts from a frontal view u, i.e., $\lambda = 1$, $t = 1$, $\varphi = \psi = 0$. The camera can first move parallel to the object's plane: this motion induces a translation T that is eliminated by assuming without loss of generality that the camera axis meets the image plane at a fixed point. The plane containing the normal and the optical axis makes an angle φ with a fixed vertical plane. This angle is called "*longitude*". Its optical axis then makes a θ angle with the normal to the image plane u. This parameter is called "*latitude*". Both parameters are classical coordinates on the "*observation hemisphere*". The camera can rotate around its optical axis (rotation parameter ψ). Last but not least, the camera can move forward or backward, as measured by the zoom parameter λ. We have seen that the affine model is enough to give an account of local projective deformations. If the camera were not assumed to be far away, the plane image deformation under a camera motion would be a homography. Yet, as explained above, a homography is locally tangent to an affine map.

Transition tilts, why they can be so high?

The parameter t defined above is called *absolute tilt*, since it measures the tilt between the *frontal* view and a *slanted* view. In real applications, both compared images are usually slanted views. The *transition tilt* is designed to quantify the amount of tilt between two such images. Assume that $v(x,y) = u(A(x,y))$ and $w(x,y) = u(B(x,y))$ are two slanted views of an flat scene whose image is $u(x,y)$, where A and B are two affine maps. Then $v(x,y) = w(AB^{-1}(x,y))$. The transition tilt between v and w is defined as the absolute tilt associated with the affine map AB^{-1}. Let t and t' the absolute tilts of two images u and u', and let φ and φ' be their longitude angles. The *transition tilt* $\tau(u,u')$ between the two images depends on the absolute tilts and the longitude angles, and satisfies

$$t/t' \leq \tau(u, u') = \tau(u',u) \leq t\, t'$$

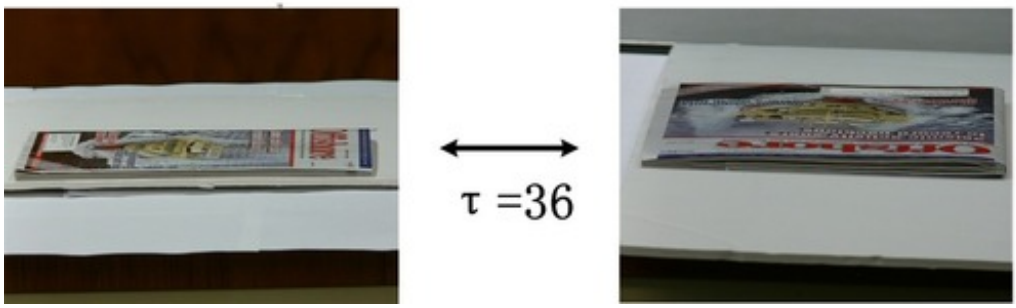
where we assume $t \geq t'$. The transition tilt can therefore be be much higher than an absolute tilt. Hence, it is important for an image matching algorithm to be invariant to high transition tilts.

Figure 4 illustrates an example of high transition tilt. The frontal image (above) is squeezed in one direction on the left image by a slanted view, and squeezed in an orthogonal direction by another slanted view. The *absolute tilt* (compression factor) is about 6 in each view. The resulting compression factor, or *transition tilt* from left to right is actually 6x6=36.

Figure 4. High transition tilt.

$t=6,\ \phi=0$ $\qquad$ $t'=6,\ \phi'=\pi/2$

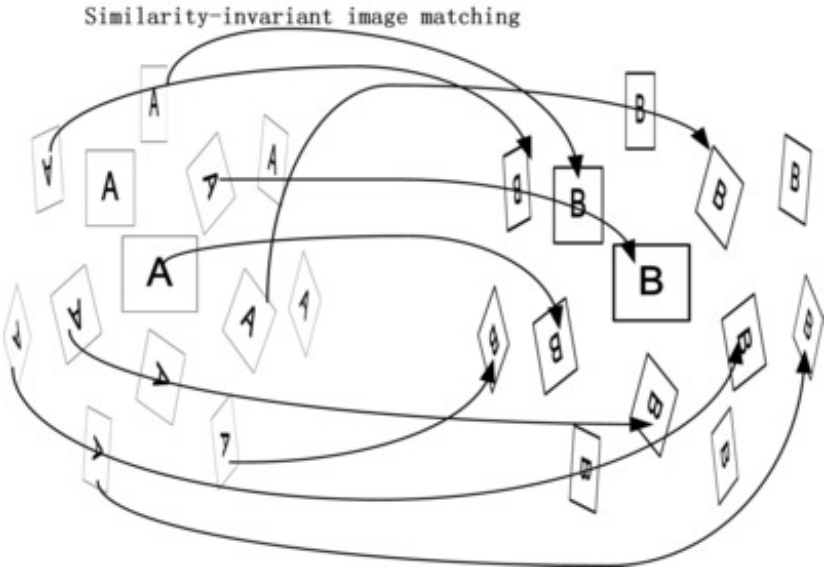
Figure 4. High transition tilt.



Description of the ASIFT algorithm

A fully affine invariant image matching algorithm needs to cover the 6 affine parameters. The SIFT method covers 4 parameters by normalizing *rotations* and *translations*, and simulating all *zooms out* of the query and of the search images.

Figure 5. ASIFT algorithm overview.



As illustrated in Figure 5, ASIFT complements SIFT by simulating the two parameters that model the camera optical axis direction (the original and simulated images are represented respectively by squares and parallelograms), and then applies the SIFT method to compare the simulated images, so that all the 6 parameters are covered. In other words, ASIFT **simulates three parameters**: the scale, the camera *longitude angle* and the *latitude angle* (which is equivalent to the *tilt*) and **normalizes the other three** (*translation* and *rotation*). ASIFT can thus be mathematically shown to be fully affine invariant (see ref. 1). Against any prognosis, simulating the whole affine space is not prohibitive at all with the proposed affine space sampling (see ref. 1 ref. 2).

Algorithm description

ASIFT proceeds by the following steps.

1. Each image is transformed by simulating all possible affine distortions caused by the change of camera optical axis orientation from a frontal position. These distortions depend upon two parameters: the longitude φ and the latitude θ. The images undergo rotations with angle φ followed by tilts with parameter $t = 1/|\cos\theta|$ (a tilt by t in the direction of x is the operation $u(x,y) \rightarrow u(tx,y)$). For digital images, the tilt is performed by a directional t-subsampling. It therefore requires the previous application of an antialiasing filter in the direction of x, namely the convolution by a Gaussian with standard deviation $\mathbf{c}\sqrt{t^2-1}$. The value $\mathbf{c} = 0.8$ is the value shown in (see ref. 3) to ensure a very small aliasing error. These rotations and tilts are performed for a finite and small number of latitude and longitude angles, the sampling steps of these parameters ensuring that the simulated images keep close to any other possible view generated by other values of φ and θ (see below).
2. All simulated images are compared by a similarity invariant matching algorithm (SIFT). SIFT can

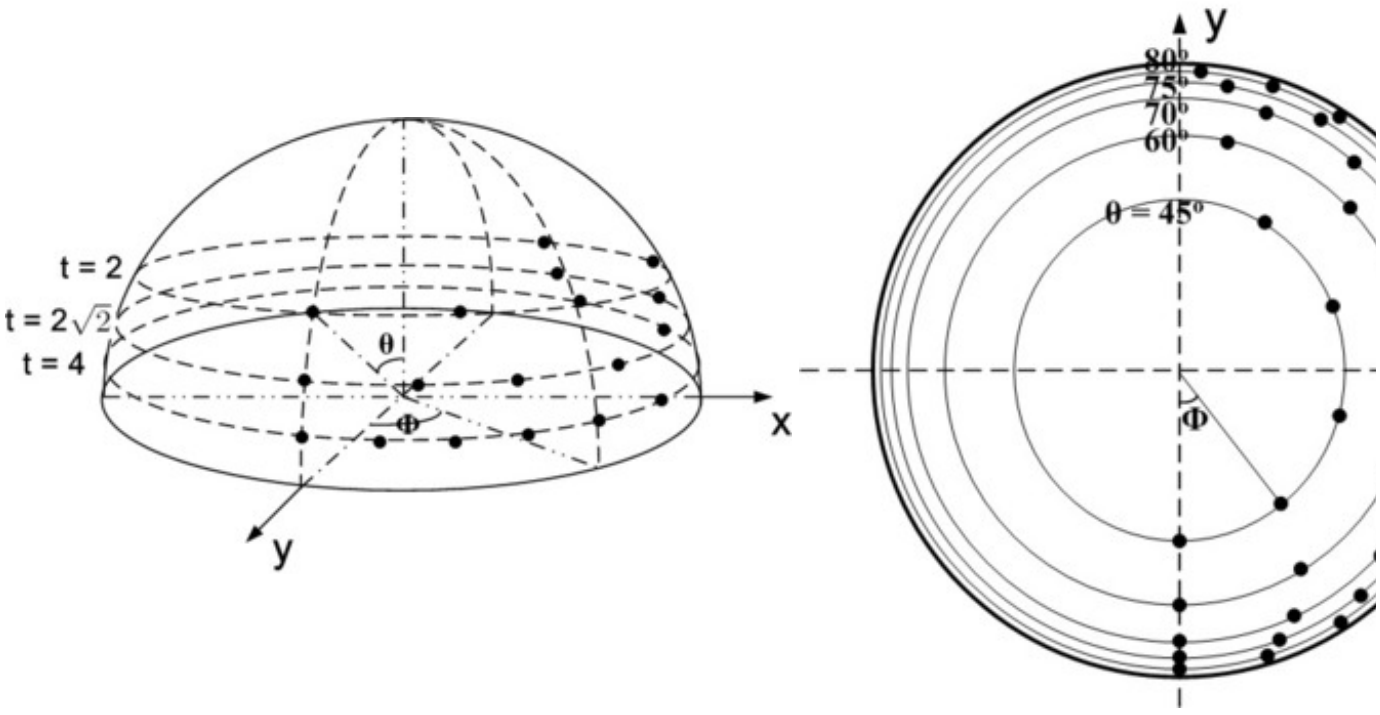
be replaced by any other similarity invariant matching method. (There are many such variants of SIFT.) It is therefore not the object of this article to describe the SIFT method.

3. The SIFT method has its own wrong match elimination criterion. Nonetheless, it generally leaves behind false matches, even in image pairs that do not correspond to the same scene. ASIFT, by comparing many pairs, can therefore accumulate many wrong matches. It is important to filter out these matches. The criterion used is that the retained matches must be compatible with an epipolar geometry. We use to that goal the ORSA method (see ref. 8), which is considered the most reliable method, robust to more outliers than a classic RANSAC procedure. It is not the goal of this article to present ORSA. It is simply used to filter out the matches given by both SIFT and ASIFT. Thus, it may occur that two images have no match left at all. This does not necessarily mean that there are no ASIFT matches; the matches may be all eliminated as incompatible with an epipolar geometry.

Parameter sampling

Figure 6(a). Sampling of the parameters.
The samples are the black dots.
Perspective illustration of the observation hemisphere.

Figure 6(b). Sampling of the param
The samples are the black dots
Zenith view of the observation hemis



The sampling precision of the latitude and longitude angles should increase with θ, since the image distortion caused by a fixed latitude or longitude angle displacement is more drastic at larger θ. As described in Figure 6, the sampling of the latitude and longitude angles is specified below.

- The latitudes θ are sampled so that the associated tilts follow a geometric series 1, a, a^2, …, a^n, with $a > 1$. The choice $a = \sqrt{2}$ is a good compromise between accuracy and sparsity. In the present implementation, the value n goes up to 5. In consequence transition tilts going up to 32 (and even a little bit more) can be explored.

- The longitudes φ are for each tilt an arithmetic series 0, b / t, …, kb / t, where $b \simeq 72°$ seems again a good compromise, and k is the last integer such that $kb / t < 180°$.

Computational complexity

Estimating the ASIFT complexity boils down to calculate the image area simulated by ASIFT. The complexity of the ASIFT feature computation is proportional to the image area under test. With the parameter sample steps $\Delta t = \sqrt{2}$, $\Delta\varphi = 72°/t$, as proposed above, the simulated image area is proportional to the number of tilts that are simulated. With the proposed simulated tilt range $[t_{min},t_{max}] = [1,4\sqrt{2}]$, which allows to cover a transition tilt as high as 32, the total simulated area is about 13.5 times the area of the original image. The ASIFT feature computation complexity is therefore 13.5 times the complexity for computing SIFT features. The complexity growth is "*linear*" and thus marginal with respect to the "*exponential*" growth of transition tilt invariance.

Since ASIFT simulates 13.5 times the area of the original images, it generates about 13.5 times more features on both the query and the search images. The complexity of ASIFT feature comparison is therefore $13.5^2 \simeq 180$ times as much as that of SIFT.

Note that on typical images, the ASIFT feature computation dominates the computational complexity with respect to feature comparison. So the total ASIFT computation complexity typically is about 13.5 times that of SIFT when comparing only a few images. If instead the problem is to compare an image to a huge database, this complexity is no more negligible, and having 180 times more comparisons to perform is a serious limitation.

Coarse-to-fine acceleration

An easy coarse-to-fine acceleration described in (see ref. 1) reduces respectively the complexity of ASIFT feature computation and comparison to 1.5 and 2.25 times that of SIFT. This acceleration is not used here.

Parallelization

In addition, the SIFT subroutines (feature computation and comparison) in ASIFT are independent and can easily be implemented in parallel. The online demo uses this possibility.

Implementation

ASIFT feature computation

Implemented in the C++ source file `compute_asift_keypoints.cpp`.

```
Input:
image u
Input parameters:
1. Tilt sampling step: \delta_t = \sqrt{2}
2. Tilt sampling range: n = 5
3. Rotation sampling step factor: b = 72
Output: ASIFT keypoints (referenced by tilt and rotation values)

// loop over tilts
for t = 1, \delta_t, \delta_t^2, ..., \delta_t^n
{
    // when t = 1 (no tilt), no need to simulate rotation
    if ( t == 1 )
    {
        theta = 0;
        // calculate scale-, rotation- and translation-invariant features on the origina
l image
        key(t, theta) = SIFT(u) (--C++ routine: compute_sift_keypoints)
    }
    else
    {
        // loop over rotations (angle in degree)
        for theta = 0, b/t, 2b/t, ..., kb/t (such that kb/t < 180)
        {
            // Rotate image (with bilinear interpolation)
            u_r = rot(u, theta)
            // Anti-aliasing filtering (G_{0.8t} is a Gaussian convolution with kernel s
tandard deviation equal to 0.8t)
            // This 1-dimensional convolution is made in the vertical direction, before
sub-sampling in the same direction.
            u_f = G_{0.8t} u_r
            // Tilt image (subsample in vertical direction by a factor of t)
            u_t = tilt(u_f, t)
           // calculate scale-, rotation- and translation-invariant features
            key(t, theta) = SIFT(u_t) (--C++ routine: compute_sift_keypoints)

            Remove the keypoints close to the boundary of the rotated and tilted image s
upport (parallelogram) in u_t.
            The distance threshold is set equal to 6\sqrt{2} times the scale of each key
point.

            Normalize the coordinates of the keypoints from the rotated and tilted image
 u_t to the original image u.
```

```
        }
    }
}
```

ASIFT feature comparison

Implemented in the C++ source file `compute_asift_matches.cpp`.

```
Input:
ASIFT keypoints of the two images: key1 and key2
Output:
Matched keypoints: matchinglist

// loop over tilts on image 1
for t1 = 1, \delta_t, \delta_t^2, ..., \delta_t^n
{
    // when t1 = 1 (no tilt), no rotation simulated
    if ( t1 == 1 )
    {
        theta1_all = [0]
    }
    else
    {
        theta1_all = [0, b/t1, 2b/t1, ..., kb/t1] (such that kb/t1 < 180)
    }
    // loop over rotations on image 1 (angle in degree)
    for theta1 = theta1_all[1], ..., theta1_all[end]
        // loop over tilts on image 2
        for t2 = 1, \delta_t, \delta_t^2, ..., \delta_t^n
        {
            // when t2 = 1 (no tilt), no rotation simulated
            if ( t2 == 1 )
            {
                theta2_all = [0]
            }
            else
            {
                theta2_all = [0, b/t2, 2b/t2, ..., kb/t2] (such that kb/t2 < 180)
            }
            // loop over rotations on image 2 (angle in degree)
            for theta2 = theta2_all[1], ..., theta2_all[end]
            {
                // Matching the keypoints between the two simulated images
                matchinglist(t1, theta1, t2, theta2) = SIFT_match(keys1(t1, theta1), key
s2(t2, theta2))
                (--C++ routine: compute_sift_matches)
            }
        }
}

Reshape matchinglist to a 1D vector (the tilt and rotation parameters are no longer usef
ul).

// matchinglist may contain identical matches which were obtained in different simulated
 image pairs.
Remove the identical matches in matchinglist (retain only 1).
(Two matches are considered identical if the distances between the points in both ends a
re small than \sqrt{2}.)

// On interpolated images, SIFT is sometimes subject to an artifact:  one
// keypoint in one image is matched with multiple keypoints in the other image.
// These are  false matches which must be detected and  removed.
Remove all the matches such that a keypoint in one image is matched with multiple keypoi
nts in another image.
(A keypoint is considered matched with multiple keypoints if the distance between the po
ints is smaller than 1
in one end and larger than 2 in the other.)

// Eliminate false matches by epipolar geometry filtering  [8]
matchinglist  = ORSA(matchinglist) (--C++ routine: orsa)
```

Examples

ASIFT is compared with the four state-of-the-art algorithms the SIFT, Harris-Affine, Hessian-Affine and MSER detectors, all coded with the SIFT descriptor. Various types of images (size 600×450) were used for the experiments. The SIFT software is from D. Lowe. The Harris-Affine, Hessian-Affine and MSER

programs are from the web site of K. Mikolajczyk.

- video tracking
- planar objects
- monuments and constructions
- 3D objects
- complex scenes
- object deformation

The matches are connected by white segments.

Video tracking

ASIFT is compared with SIFT on video tracking . In each video, ASIFT and SIFT tracking are shown respectively on the top and bottom. If you prefer high-resolution video, please download the video files by clicking the links below.

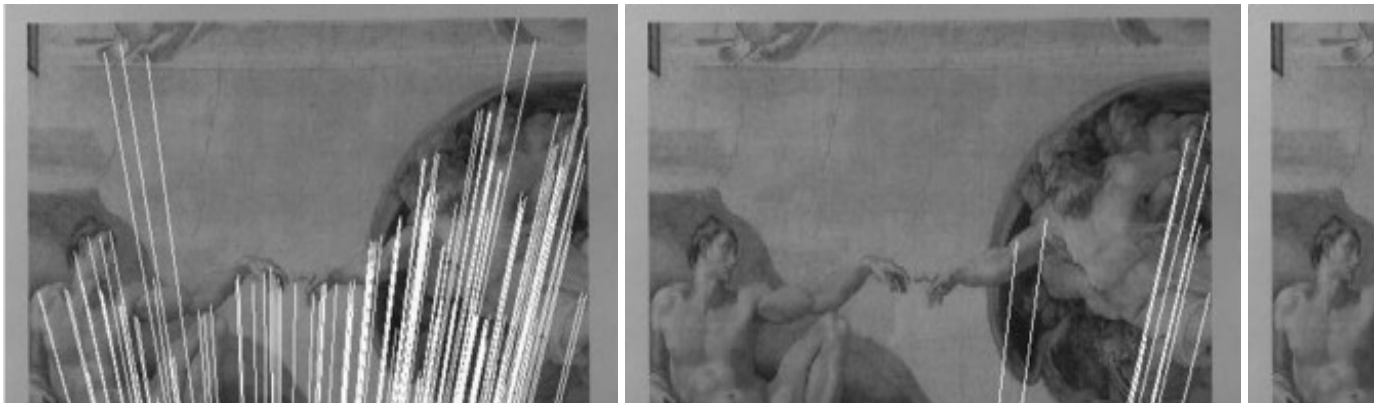
facade (youtube page, high-resolution video) *magazine (you*

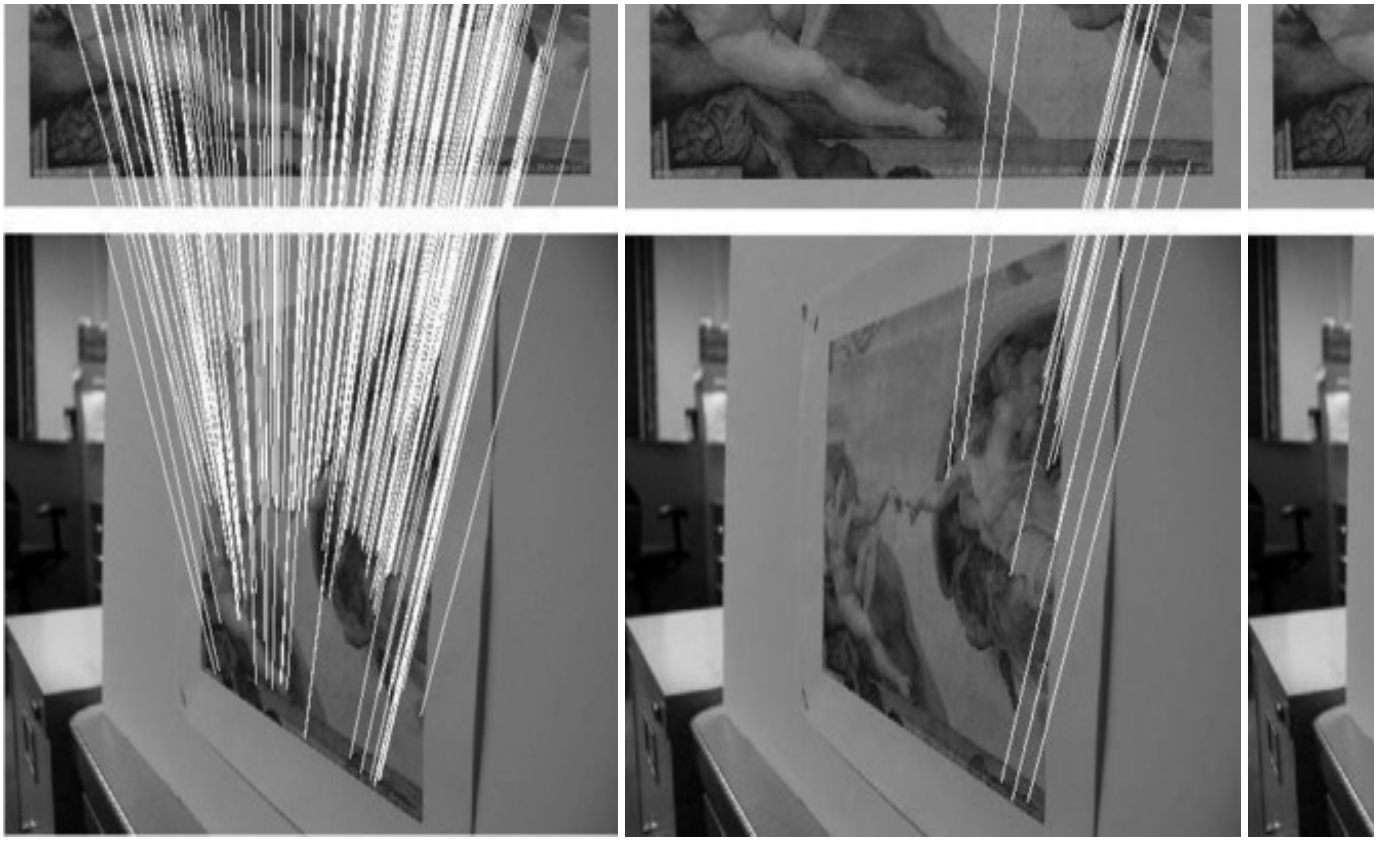
Planar objects

- Adam short distance (zoom ×1) at frontal view and at 75 degree angle, absolute tilt t = 4 (middle), <4 (left), >4 (right)

Not shown Harris-Affine: 3 matches. Hessian-Affine: 1 match

ASIFT: 202 matches *SIFT: 15 matches*





- Adam short distance (zoom ×10) at frontal view and at 65 degree angle, absolute tilt t = 2.4
- Adam short distance (zoom ×10) at frontal view and at 80 degree angle, absolute tilt t = 5.8
- magazine middle distance (zoom ×4) at frontal view and at 80 degree angle, absolute tilt t = 5.8
- magazine absolute tilt t1= t2= 2, with longitude angles phi 1= 0 deg, phi2= 50 deg, transition tilt t = 3
- magazine absolute tilt t1= t2= 4, with longitude angles phi1= 0 deg, phi2= 90 deg, transition tilt t = 16
- facade frontal view and at 75 degree angle, absolute tilt t = 3.8
- graffiti no.1 vs no. 6 (images from K. Mikolajczyk), transition tilt t ~ 3.2
- direction transition tilt t ~ 2.6
- parkings transition tilt t ~ 15
- stump transition tilt t ~ 2.6

Monuments and constructions

- round building transition tilt t ~ [1.8, inf.)
- palace of Versailles transition tilt t ~ 1.8
- École Polytechnique pictures at frontal view and at 65 degree angle, absolute tilt t = 2.4

3D objects

- statue transition tilt t ~ [1.6, inf.)
- can transition tilt t ~ [2.3, inf.)

Complex scenes

- office transition tilt t ~ 3
- coffee room transition tilt t ~ [1.5, 3.3]

Object deformation

Images from Ling and Jacobs.

- Sponge Bob
- CVPR
- flag
- girl
- toy

Failure Cases

Strong relief effect

All methods fail!

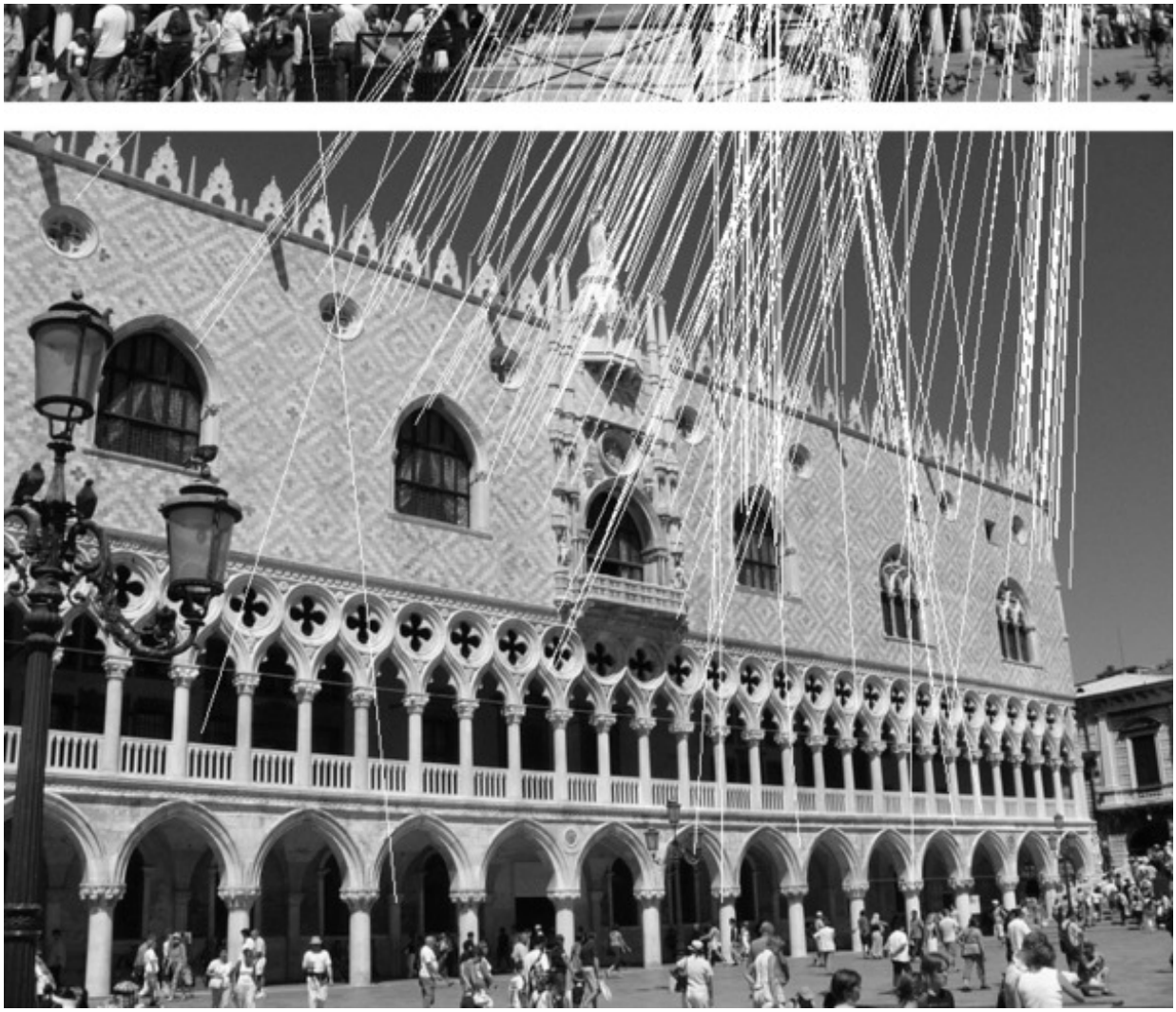
ASIFT, SIFT, Harris-Affine, Hessian-Affine, MSER: 0 match.

The image on the right in close-up shows strong relief effect.

Repetitive shapes

"Good" false matches.

Matches can be arbitrary among repetitive shapes.



ASIFT: 171 matches, many are "good" false matches (for example the matches between some windows).

image credits