Comparison of Viewpoint-Invariant Template Matchings

Guillermo Angel Pérez López
Escola Politécnica da Universidade de São Paulo
São Paulo, Brazil
guillermo.angel@usp.br

Hae Yong Kim
Escola Politécnica da Universidade de São Paulo
São Paulo, Brazil
hae@lps.usp.br

Abstract — **SIFT, Forapro and Ciratefi are three rotation and scale invariant template matching (or keypoint matching) algorithms. They can become viewpoint-invariant using the view simulation technique, compensating the perspective deformation by simulated affine transformations, yielding the algorithms named ASIFT (Affine-invariant SIFT), Aforapro and Aciratefi. We compare the three algorithms, evaluating their performances in different situations. We focus our attention especially to situations with illumination changes and in the presence of repetitive similar patterns. The results show that all the three algorithms have strengths and weaknesses and the user should choose the best suited algorithm according to the intended application.**

Keywords — Template matching, perspective invariance, viewpoint invariance, affine transformation, ASIFT, Aforapro, Aciratefi.

I. INTRODUCTION

The goal of a template matching algorithm is to find the occurrence of a small query template image Q in a larger image to analyze (also called search image) A. In this paper, we will consider only grayscale images. Some of the difficulties of this process are: (1) the images may appear in different rotations and scales; (2) the images can suffer perspective deformation due to the change of the viewpoint; (3) the images may change due to photometric effects that include brightness/contrast changes; (4) the camera optical device may blur the image (e.g. image out of focus); (5) some sub-patterns of the query image Q may appear repeatedly out of the sought instance Q in A.

One of the authors of this paper have participated in the proposal of two rotation and scale invariant template etchings named Forapro (Fourier coefficients of Radial Projections [1]) and Ciratefi (Circular, Radial and Template matching Filter [2, 3]). A rotation and scale invariant template matching can become invariant to viewpoint changes by using “viewpoint simulations”. It consists in deforming the images by affine transforms and choosing the best match between the deformed images. This technique was first used in the technique named ASIFT [4] to make the well-known keypoint matching algorithm SIFT [5] viewpoint-invariant. We use the same idea to make Forapro and Ciratefi affine-invariant, yielding Aforapro and Aciratefi. We have published preliminary papers on Aforapro in Portuguese in two local conferences [6, 7]. This is the first time we are presenting Aciratefi.

Affine transformations preserve parallelism of lines. Perspective transformations (viewpoint changes) do not. However, any perspective transformation can be approximated locally by an affine transformation. So, any affine-invariant template matching is also automatically robust to viewpoint changes if the algorithm actually matches small sub-templates (or small regions around keypoints) and uses these sub-matchings to find the final matching, usually through the generalized Hough transform [8]. SIFT matches small regions around keypoints; Forapro matches small sub-templates and Ciratefi matches the whole query image at once. So, we expect ASIFT and Aforapro to be more robust to viewpoint changes than Aciratefi. On the other hand, using small sub-templates makes the algorithm weaker in the presence of repetitive patterns (like repeating letters in the text). So, we expect Aciratefi behave more robustly than ASIFT and Aforapro in the presence of repetitive patterns. We will make some empirical experiments to confirm (or reject) these hypotheses. We also test the three algorithms under the illumination changes to test their robustness to brightness/contrast variations.

We stress that ASIFT is not actually a template matching algorithm, but rather an algorithm for finding correspondences between two images. However, it can be used for template matching task by clustering the keypoint matches to detect a final matching. To find the final matching, it is necessary at least two keypoint matches. So, we consider that SIFT has successfully matched the template if there are at least two correct keypoint matches.

This paper is organized as follows: Section 2 presents the desirable properties of template matching algorithms; Section 3 describes briefly Aforapro and Aciratefi algorithms; Section 4 presents experimental results; and Section 5 presents our conclusions.

II. DESIRABLE PROPERTIES

A. *Robustness to Viewpoint Changes*

The viewpoint is the spatial position of the camera in relation to the scene, at the time of the image capture. An object can be photographed from multiple viewpoints. The change in the camera viewpoint introduces a perspective deformation in images that can hamper finding the template.

This work was funded partially by CAPES.

The following property can be used to make a template matching algorithm robust to viewpoint changes:

Property: *The perspective deformation, induced by the change of viewpoint, can be locally approximated by an affine transformation* [4, 9].

So, an affine-invariant template matching is automatically viewpoint-robust template matching if the algorithm uses many small sub-templates to find the final matching, because the perspective deformation of each sub-template can be approximated by an affine transformation.

The affine transformation of an image I_1 is a mapping that leads to another image I_2:

$$I_2(x_2, y_2) = A * I_1(x_1, y_1) + T \qquad (1)$$

where A is an invertible 2x2 matrix and $T \in R^2$ is the translation vector. The affine transformation can be decomposed:

$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} 1 & k \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\phi & -sen\phi \\ sen\phi & \cos\phi \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} \qquad (2)$$

where S_x and S_y are the scale factors of axes x and y, k is a non-orthogonality factor between the axes, ϕ is the rotation angle on the plane, and e and f are the translation factors. In other words, the image affine transformation is a linear transformation with six degrees of freedom.

There are two techniques to achieve the viewpoint invariance:

(a) The first one uses affine-invariant local descriptors, as in [9]. This technique has limitations because the perspective deformation is not completely switchable with the distortion introduced by the camera optical device, as noted in [4].

(b) The second one uses view simulation technique proposed by Morel and Yu [4] to extend the popular SIFT (scale invariant feature transform [5]) making it affine-invariant. The robustness obtained by ASIFT stands out as one of the best affine-invariant algorithms. We use the view simulation technique to make Forapro and Ciratefi affine-invariant.

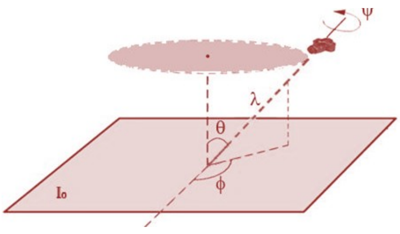


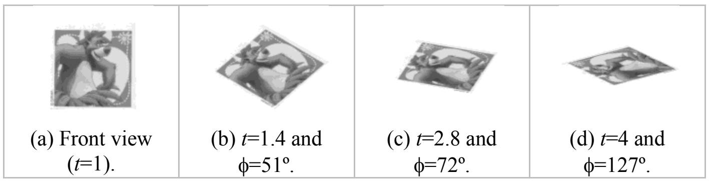
Figure 1. Affine model for image capture parameters (modified from [4]).

The camera orientation parameters of the affine model are depicted in Fig. 1. The plane I_o is the front view, ψ is the camera rotation angle on its optical axis, λ is the scale parameter (the camera distance to the center point), ϕ is the camera rotation angle on the image plane, and θ is the tilt angle measured between the normal vector of the image plane and the camera optical axis. The front viewpoint is determined by $\theta = \phi = \psi = 0$. This model transforms Equation 1 in:

$$\mathrm{A} = H_\lambda R_1(\psi) T_t R_2(\phi) = \lambda \begin{bmatrix} \cos\psi & -sen\psi \\ sen\psi & \cos\psi \end{bmatrix} \begin{bmatrix} t & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\phi & -sen\phi \\ sen\phi & \cos\phi \end{bmatrix} \qquad (3)$$

where H_λ is the scale matrix with $\lambda>0$, R_i are the rotations, and T_t is the inclination angle change with $t = 1/\cos\theta$. The parameter t is the tilt (viewpoint inclination) and can be: absolute tilt, that is, the inclination difference of the image from the front view; and transition tilt, that is, the slope measure between two different viewpoints of the same image. We use only the absolute tilt in Aforapro and Aciratefi. That is, we assume that query image Q was photographed from approximately front viewpoint. This assumption can be satisfied in most practical template matching applications, because the user normally has some control on the query image obtaining process. On the other hand, ASIFT considers both absolute and transition tilts. Note that ASIFT is not primarily intended for template matching, but to find the correspondences between two images.

The viewpoint simulation modifies the values of θ and ϕ in discrete steps to simulate the possible distortions caused by the camera position changes. The proper selection of the set of values for both parameters ensures that a small number of the simulated images are enough to have correspondence with all the distortions present in the captured images. Figure 2 shows some of these simulations. Note that of the six freedom degrees of affine transformation, we are simulating only two. Forapro and Ciratefi will take care of the remaining four, because they are rotation, scale and translation invariant.



(a) Front view (t=1). (b) t=1.4 and ϕ=51°. (c) t=2.8 and ϕ=72°. (d) t=4 and ϕ=127°.

Figure 2. Viewpoint simulation.

B. Robustness to Brightness/Contrast Changes

Brightness and contrast of the images may change due to the change of illumination, background or shadows. We define that two images, $\boldsymbol{I}_1$ and $\boldsymbol{I}_2$, are equivalent under brightness/contrast changes if there is a contrast correction factor $\beta>0$ and a brightness correction factor γ such that $\boldsymbol{I}_2 = \beta \boldsymbol{I}_1 + \gamma \boldsymbol{1}$, where $\boldsymbol{1}$ is the array of 1's [2]. We consider that a template matching algorithm is brightness/contrast invariant if it can find instances of Q in A, independently of the brightness/contrast of the instance.

In theory, all the three algorithms are brightness and contrast invariant. Moreover, when a shadow changes the brightness/contrast of only some region of the image, we expect that ASIFT and Aforapro would succeed more than Aciratefi, because the formers match small sub-templates, while the latter matches the whole template at once. Surprisingly, this is not what we observed in the experiments.

C. *Robustness to Repetitive Patterns*

An image of real world scenes may contain many repeating elements (e.g., written characters or tiles on the floor). In this case, the query image may be similar to the background structure or similar to other objects present in the scene. These repeated patterns may obstruct the template matching process, increasing the errors, because they are locally indistinguishable [10].

Given a query image Q and an image to analyze A, where n instances of Q appear inside A, we say that A has repetitive patterns of Q if some sub-parts of Q appear in A outside the n instances of Q. We define that a template matching algorithm is robust to repetitive patterns if it can find the n instances of Q in A, even when A has repetitive patterns of Q.

III. Algorithm Descriptions

Given two images Q and A, we assume that image Q presents the front viewpoint and the image A contains an instance of Q photographed from a random viewpoint. Differently from ASIFT, we do not consider the "transition tilt" where both Q and A can be viewed from different directions. This assumption is not very restrictive in practice, because normally the user has some control over the acquisition process of query image Q. Our algorithms can be summarized as follows:

A. *Aforapro*

```
Aforapro Algorithm
Input: Images $Q$, $A$ and the number of simulations $n$.
Output: The best matching $Q_i$, together with its position, angle, scale
and affine parameters in $A$.
1. Use affine transformations to simulate the view of $Q$ from
   different viewpoints ($Q \rightarrow Q_1, Q_2, \dots, Q_n$).
2. Find each simulated image $Q_i$ in $A$ using Forapro algorithm [1]:
   a. Forapro computes rotation-invariant features for each pixel of $A$.
   b. Forapro extracts from $Q_i$ smaller sub-templates $T_i^j$, computes
      the rotation-invariant features of $T_i^j$, and uses them to find the
      pixels in $A$ that best match $T_i^j$.
   c. Forapro uses the matchings of sub-templates $T_i^j$ to find the
      final matching of $Q_i$ in $A$, using the generalized Hough
      transform.
3. Forapro chooses the best $Q_i$ with the largest value in Hough
   transform as the final match of $Q$ in $A$.
```

The simulations number n must be enough to cover all the possible viewpoints of the camera. Following Morel and Yu's suggestion, we adopted $\Delta t = t_{k+1}/t_k = \sqrt{2}$ and $\Delta\emptyset = 72^\circ/t$, and consequently:

- $t = \left\{\sqrt{2}^0, \sqrt{2}^1, \sqrt{2}^2, \sqrt{2}^3, \sqrt{2}^4\right\}$, where $t = 1/\cos\theta$
- $\emptyset = \{0, b/t, \dots, kb/t\}$, where $b = 72^\circ$ e $kb/t < 180^\circ$

Thus, Aforapro generates 27 simulated images Q_i and hits a tilt up to $\theta \approx 76^\circ$. The amount of simulated images can be modified according to the application. Current Forapro implementation has some shortcomings: (1) it has many parameters that must be set manually; (2) it uses the "brute-force algorithm" to search the most similar feature. Thus, it is clearly possible to improve the current Forapro implementation.

B. *Aciratefi*

```
Aciratefi Algorithm
Input: Images $Q$, $A$ and the number of simulations $n$.
Output: The best matching $Q_i$, together with its position, angle, scale
and affine parameters in $A$.
1. Use affine transformations to simulate the view of $Q$ from
   different viewpoints ($Q \rightarrow Q_1, Q_2, \dots, Q_n$).
2. Find each simulated image $Q_i$ in $A$ using Ciratefi algorithm [2, 3]:
   a. Ciratefi successively filters out the pixels in $A$ that cannot
      match $Q_i$.
   b. Ciratefi returns the pixel in $A$ that best matches $Q_i$, together
      with its normalized correlation coefficient.
3. Choose $Q_i$ with the largest normalized correlation coefficient as
   the final match of $Q$ in $A$.
```

We used the same view simulation parameters as in Aforapro. Almost all parameters of the current Ciratefi implementation are set automatically.

Note that Aforapro actually finds small sub-templates in A, while Aciratefi matches the whole simulated query image Q at once.

IV. Experimental Results

To compare the three algorithms, we built a database with 247 images divided into three sets. Each set was further subdivided into groups. A group consists of a query image Q and multiple search images A. In each search image A, an instance of Q appears photographed from some viewpoint.

Set 1: This set consists of 47 classic images frequently used to analyze viewpoint-invariant template matching algorithms, including ASIFT. These images were obtained from the databases created by Morel and Yu [12] and Mikolajczyk [13].

Set 2: This set consists of 100 images chosen to test the robustness to brightness/contrast changes. These images contain objects of common use such as calculator and magazine, divided into 10 groups. Changes in brightness/contrast were induced using lighting effects and shadows. The scenario was a laboratory room.

Set 3: This set consists of 100 images chosen to test the robustness to repetitive patterns. These images contain (for example) Google maps, signs, advertisements, silhouettes and bushes. They were subdivided into 10 groups with viewpoint changes.

We used two different assessment criteria because ASIFT indicates the matching using correspondence lines between keypoints but Aforapro and Aciratefi indicate the matching by drawing a rectangle or circle (Figure 3). So, we consider that a matching is correct if: (a) For ASIFT, if there are at least 2 lines indicating correctly the correspondence between the query and the instance; (b) For Aforapro and Aciratefi, if the rectangle or circle intersects at least 60% of the sought instance area.

A. Viewpoint invariance

We used the image set 1 to test robustness to viewpoint changes. In this set, query images Q have typically 50x50 or 70x70 pixels and search images A have typically 500x375 pixels. We used the ASIFT implementation provided by the authors [12]. Figure 3 depicts some of our tests and Table I shows the obtained matching success rates and average processing times.

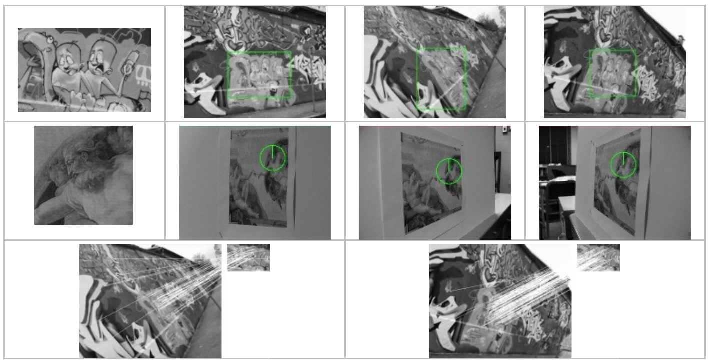


Figure 3. Some of the experiments made to test the robustness to viewpoint changes. The two upper leftmost images are the query images. In the remaining images, a green rectangle (Aforapro), green circle (Aciratefi) and white lines (ASIFT) indicate the localized instance.

TABLE I. MATCHING SUCCESS RATES AND AVERAGE PROCESSING TIMES IN SET 1 (VIEWPOINT CHANGES).

Group	**Potential matches**	**ASIFT**		**Aforapro**		**Aciratefi**	
		(%)	(s)	(%)	(s)	(%)	(s)
Adam	16	100	45.2	100	255.4	100	210.6
Lepoint	23	100	38.1	100	257.9	91	305.2
Grafite	8	100	56.1	100	259.5	100	247.4
Total	**47**	**100**	**46.5**	**100**	**257.3**	**97**	**254.4**

Aciratefi failed to localize some templates, especially with large viewpoint changes ($\theta \approx 80^\circ$). Aciratefi fails in these cases probably because it searches for the whole template at once, while ASIFT searches for regions around keypoints and Aforapro search for sub-templates and summarizes the matching using the generalized Hough transform. These tests also give us an indirect comparison with three other affine invariant algorithms related in the literature (MSER [11], Harris-affine [9], Hessian-affine [9]) because, according to Morel and Yu, none of these methods successfully matches with viewpoint change larger than $\theta \geq 70^\circ$ using the same images. ASIFT was the fastest method.

B. Robustness to brightness/contrast changes

We used the set 2 to test the robustness to brightness/contrast changes. In this set, query images Q have typically 50x50 pixels and search images A have typically 500x375 pixels. Figure 4 depicts some of our tests and Table II shows the obtained matching success rates and average processing times.

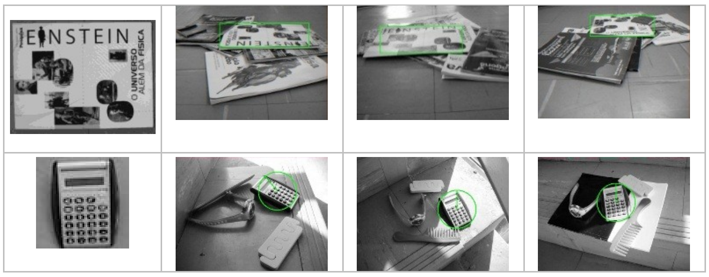


Figure 4. Some of the experiments made to test robustness to illumination changes. The leftmost images are the query images.

TABLE II. MATCHING SUCCESS RATES AND AVERAGE PROCESSING TIMES IN SET 2 (ILLUMINATION CHANGES).

Group	**Potential matches**	**ASIFT**		**Aforapro**		**Aciratefi**	
		(%)	(s)	(%)	(s)	(%)	(s)
A1	10	60	120.4	100	230.8	80	181.3
B1	10	40	100.7	100	150.1	100	202.4
C1	10	50	54.2	80	164.8	90	216.0
D1	10	20	56.6	90	204.3	100	189.7
E1	10	30	49.0	100	162.5	70	174.5
F1	10	40	66.3	90	207.3	90	168.7
G1	10	60	59.8	100	176.0	100	210.1
J1	10	100	57.1	100	208.4	90	185.3
K1	10	80	70.6	100	223.7	100	180.8
L1	10	50	90.2	90	180.5	100	120.5
Total	**100**	**53%**	**72.5**	**95%**	**190.8**	**92%**	**182.9**

The results in Table II show that Aforapro and Aciratefi outperform ASIFT under large illuminations variations (matching rates respectively 95%, 92% and 53%). ASIFT is still the fastest of the three algorithms. However, the processing times of Aforapro and Aciratefi have decreased, because in this case we used only small query images (50x50 pixels) while the processing time of ASIFT has increased because the involved images have many textures, yielding many keypoints to be matched. These results show that Aforapro and Aciratefi are indeed brightness/contrast invariant, and that they outperform ASIFT in situations with large illumination variations.

C. Robustness to repetitive patterns

We used the set 3 to test the robustness to repetitive patterns. In this set, query images Q have typically 50x50 or 70x70 pixels and search images A have typically 500x375 pixels. Figure 5 depicts some of our tests and Table III shows the obtained matching success rates and average processing times. With repetitive patterns, ASIFT presents very low successful matching rate (35%), while Aforapro a Aciratefi present relatively high success rates (92% and 91%). ASIFT matches small regions around the keypoints and consequently it is very difficult to successfully match these regions when there are many similar repetitive patterns. Meanwhile, Aforapro matches usually larger sub-templates and Aciratefi matches the whole template at one, thus the repetitive patterns do not affect these methods so much. ASIFT is, once again, the fastest method.

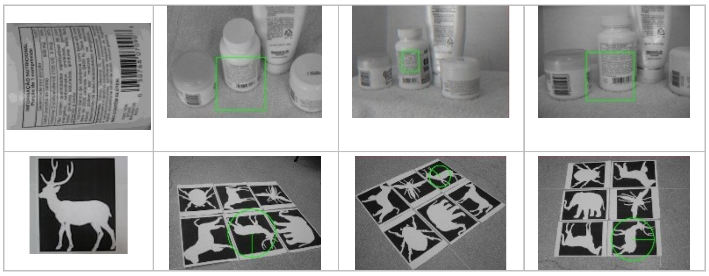


Figure 5. Some of the experiments made to test robustness to repetitive patterns. The leftmost images are the query images.

TABLE III. MATCHING SUCCESS RATES AND AVERAGE PROCESSING TIMES IN SET 3 (REPETITIVE PATTERNS).

Group	Potential matches	ASIFT		Aforapro		Aciratefi	
		(%)	(s)	(%)	(s)	(%)	(s)
A2	10	10	50.3	90	230.4	100	180.7
B2	10	10	56.9	80	218.9	90	210.5
C2	10	40	110.5	70	183.1	90	204.8
D2	10	60	168.1	100	240.4	100	194.1
E2	10	30	60.6	90	220.8	100	209.3
F2	10	40	54.3	100	190.3	80	225.4
G2	10	50	140.6	100	205.4	80	183.2
J2	10	30	58.7	100	228.0	90	246.4
K2	10	20	50.8	100	259.5	100	239.6
L2	10	60	150.7	90	216.2	80	195.2
Total	**100**	**35%**	**90.1**	**92%**	**219.3**	**91%**	**208.9**

D. Results summary

We summarize in Table IV the results obtained so far. Actually, ASIFT is a keypoint matching algorithm while Aforapro is a sub-template matching algorithm and Aciratefi is a "traditional" a template matching algorithm. ASIFT is the only algorithm that can match a slanted view to another slanted view (transition tilt), while Aforapro and Aciratefi only matches frontal view to slanted view.

Our tests with set 1 indicate that ASIFT and Aforapro are more robust to viewpoint changes than Aciratefi. Our tests with set 2 indicate that ASIFT is less robust than Aforapro and Aciratefi when there are large illumination changes. Our tests with set 3 indicate that ASIFT is less robust than Aforapro and Aciratefi in the presence of repetitive similar patterns.

On the other hand, ASIFT was the fastest method in all tests. Finally, the current Aforapro implementation has many parameters to set manually, while ASIFT and Aciratefi choose almost all parameters automatically.

TABLE IV. SUMMARY OF COMPARISON RESULTS, USING RATES: EXCELLENT, GOOD, REGULAR AND POOR.

	ASIFT	Aforapro	Aciratefi
Specialization	keypoint matching	sub-template matching	template matching
Transition tilt	yes	no	no
Robustness to viewpoint changes	excellent	excellent	good
Robustness to illum. changes	regular	good	good
Robustness to repetitive patterns	poor	good	good
Processing time	good	poor	poor
Automatic parameters setting	good	poor	good

V. CONCLUSIONS

In this paper, we have compared three viewpoint-invariant template matching algorithms (ASIFT, Aforapro and Aciratefi) in different situations. Among them, we have proposed the last two algorithms. ASIFT was the fastest algorithm in all tests. On the other hand, Aforapro and Aciratefi presented better matching success rates with illumination changes and in the presence of repetitive patterns. We conclude that the three algorithms have strengths and weaknesses, and the user should choose the best suited one for each application.

REFERENCES

[1] H. Y. Kim, "Rotation-Discriminating Template Matching Based on Fourier Coefficients of Radial Projections with Robustness to Scaling and Partial Occlusion," *Pattern Recognition*, 43(3), 859-872, 2010.

[2] H. Y. Kim and S. A. Araújo, "Grayscale Template-Matching Invariant to Rotation, Scale, Translation, Brightness and Contrast," *IEEE Pacific-Rim Symp. Image and Video Tech*, LNCS, vol. 4872, 100-113, 2007.

[3] S. A. Araújo and H. Y. Kim, "Ciratefi: An RST-Invariant Template Matching with Extension to Color Images," *Integrated Computer-Aided Engineering*, 18(1), 75-90, 2011.

[4] J. Morel and G. Yu, "ASIFT: A new framework for fully affine invariant image comparison," *SIAM Journal on Imaging Sciences*, 2(2), 438-469, 2009.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Computer Vision* 60(2), 91-110, 2004.

[6] H. Y. Kim, G. A. P. López, "Reconhecimento de Padrões Invariante Afim Baseado nos Coeficientes de Fourier de Projeções Radiais," *VI Workshop de Visão Computacional*, 2010.

[7] G. A. P. López, H. Y. Kim, "Novo Algoritmo para Reconhecimento de Objetos Invariante Afim," *XXIX Simp. Bras. Telecomunicações - SBrT'11*, 2011.

[8] D. H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes," *Pattern Recognition*, 13(2), 111-122, 1981.

[9] K. Mikolajczyk and C. Schmid, "Scale & affine invariant interest point detectors," *Int. J. Computer Vision*, 60(1), 63-86, 2004.

[10] P. Doubek, J. Matas, M. Perdoch, and O. Chum, "Image matching and retrieval by repetitive patterns," In *IEEE Int. Conf. Pattern Recognition (ICPR), 2010.*

[11] J. Matas. O. Chum. M. Urban and T. Pajdla. "Robust wide-baseline stereo from maximally stable extremal regions," Image Vision Computing. 22(10). pp. 761-767. 2004.

[12] J. Morel and G. Yu, SIFT and ASIFT: IPOL Database. <http://www.cmap.polytechnique.fr/~yu/research/ASIFT/>. Accessed in Feb. 15, 2014.

[13] K. Mikolajczyk, Harris-affine and Hessian-affine algorithms. Database. <http://www.robots.ox.ac.uk/~vgg/research/affine/>. Accessed in Feb. 15, 2014.