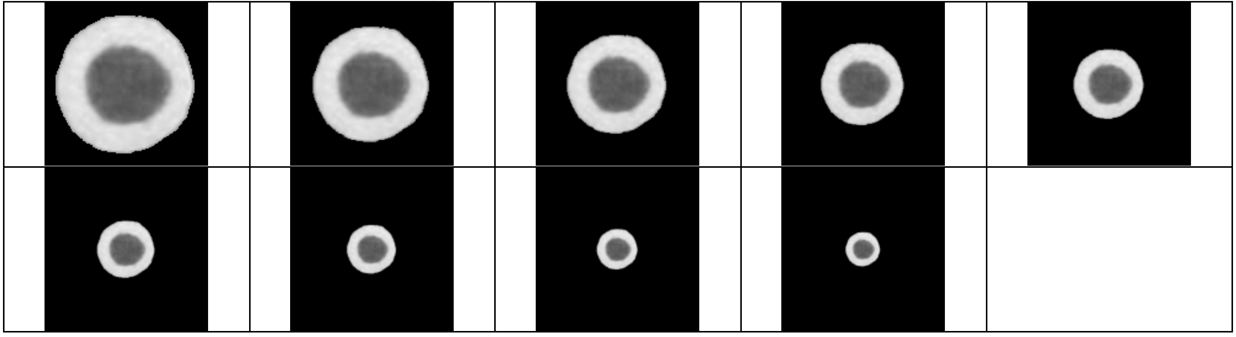
Força Bruta para Obter Invariância à Escala

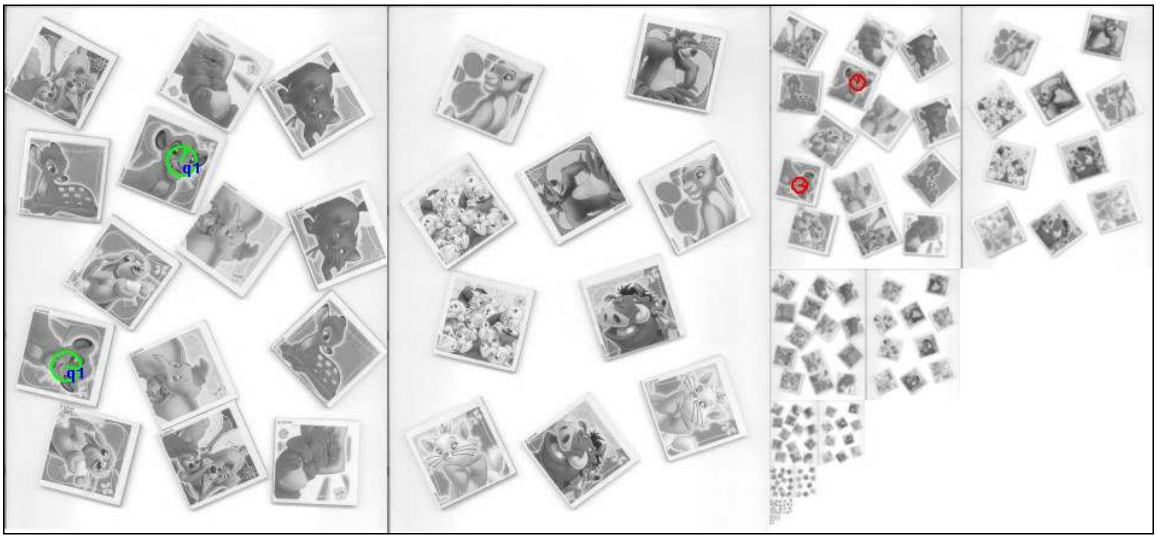
Veja vídeo em www.lps.usp.br/~hae/apostila/mmatch4.avi

Geração de multi-escala em progressão geométrica:



Por que é melhor usar progressão geométrica do que progressão aritmética?

Invariância à escala fica mais simples com uso de pirâmide, onde subir um andar é reduzir a resolução por 2. Se procurar escalas 0,5 a 1,0 na estrutura de pirâmide, está procurando escalas de 0,5 a ∞.



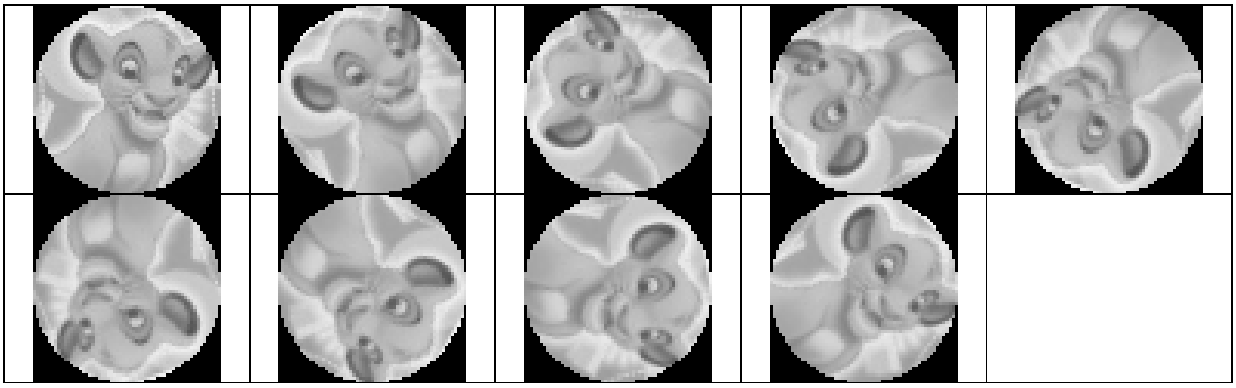
O uso de pirâmide aumenta em menos de 50% o esforço computacional e aumenta invariância de escala de [0,5...1,0[para [0,5... ∞]. Aproximadamente 5 modelos bastam para fazer busca na escala [0,5...1,0[.

Força Bruta para Obter Invariância à Rotação

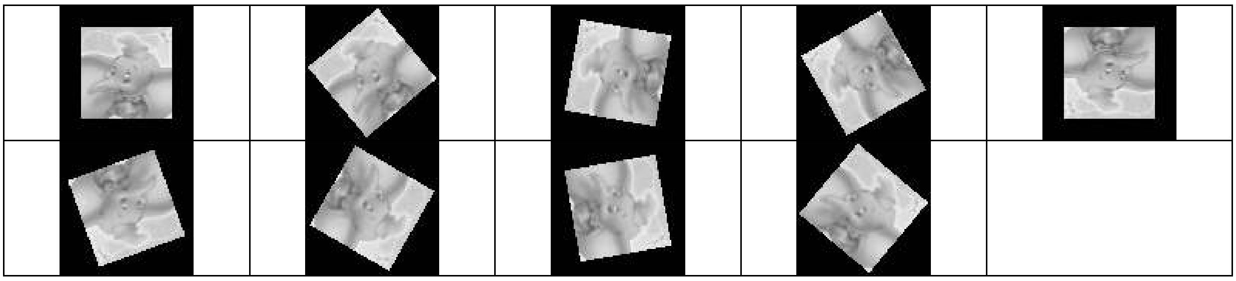
É possível obter invariância à rotação, rotacionando *Q* em diferentes ângulos e procurando-os todos em *A*.

A figura abaixo mostra a imagem-modelo *Q* rotacionada em 9 ângulos. Os pixels pretos não fazem parte da imagem-modelo (são pixels "don't care").

Pergunta: Como implementar template matching com pixels "don't care"?



Método 1



Método 2

O programa:

```
pv ncc_r
```

faz o template invariante à rotação força bruta.

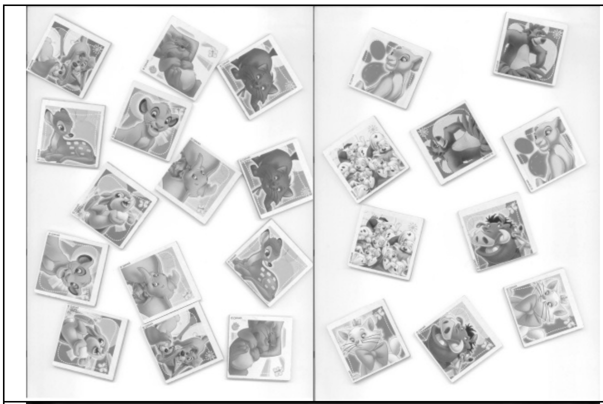
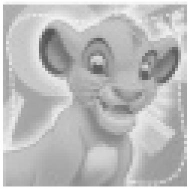
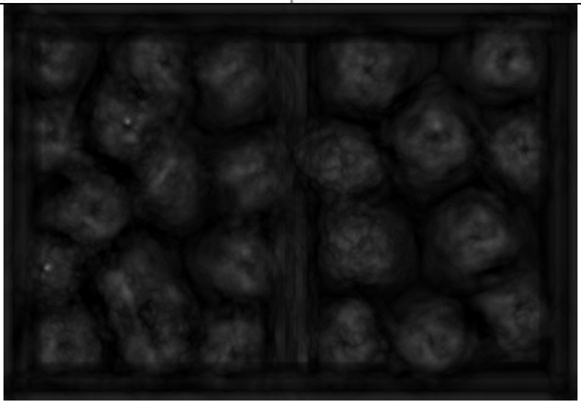
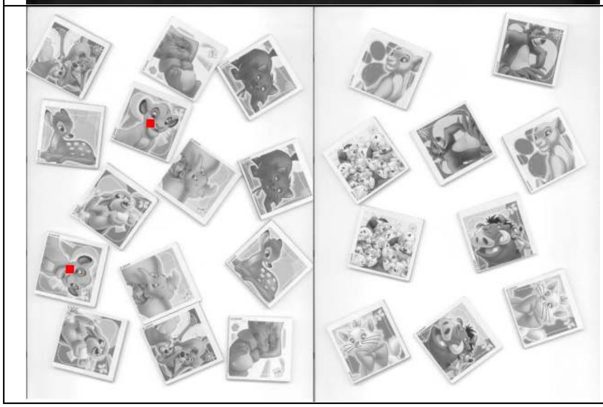
Nota 1: Na verdade, a "força bruta" não é tão bruta assim, pois o template matching cas bibliotecas (ex: OpenCV) são altamente otimizado usando FFT e imagem integral.

Nota 2: Procurar Q_1, ..., Q_n em A (usando correlação cruzada ou coeficiente de correlação normalizada) pode ser acelerado calculando FFT e imagem integral de A uma única vez.

Nota 3: SIFT, SURF e assemelhados conseguem invariância a rotação e escala detectando pontos-chaves e processando regiões em torno de pontos chaves.

Nota 4: Viola e Jones consegue robustez a deformações utilizando aprendizagem de máquina. Utiliza imagem integral para acelerar as buscas (inclusive busca em múltipla escala).

Exemplo em c:\haepi\algpi\rstmatch\rstmatch\brutag

```
pv ncc_r a01.jpg q01.jpg p01.tga 36
img kmax p00.tga p00.pbm 2 61 9
img sobrmmg a01.jpg p01.ppm p01.pbm
```

```
NCC_R: normalized cross-correlation invariant to rotation
Syntax1: NCC_R analyze.pgm query.pgm sai.ppm nang thresh
Syntax2: NCC_R analyze.pgm query.pgm sai.img nang

kmax ent.img sai.pbm k dist [tamanho]
  Acha k maximo que estao separados por mais de d pixels por heap
  ent.img e' IMGFLT (pode ser grayscale)
  Pixels>=1.0 nao sao levados em conta
    assim a saida pode ter menos que k elementos.

SobrMMG: Sobrepoe multiplas mascaras binarias em IMGGRY.
SobrMMG OriG.tga SaiC.tga mas1.bmp mas2.bmp mas3.bmp ...
SobrMMG OriG.tga SaiC.tga mas?.bmp
  Imagens mas1-mas6 sao pintados red, green, blue, cyan, mag, yel
  As demais sao pintadas com cores aleatorias
```

Ciratefi (circular, radial and template matching filter)

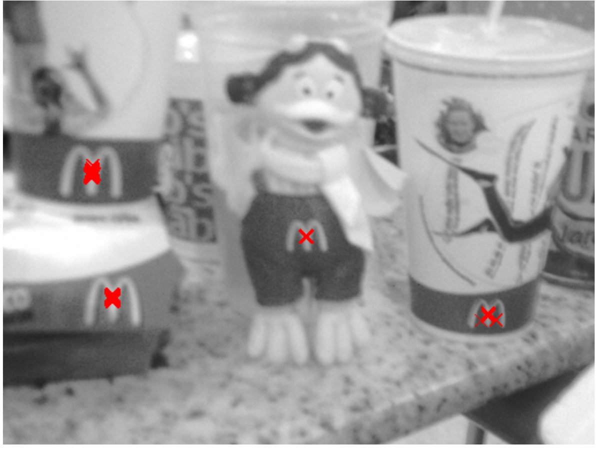
1. Introduction

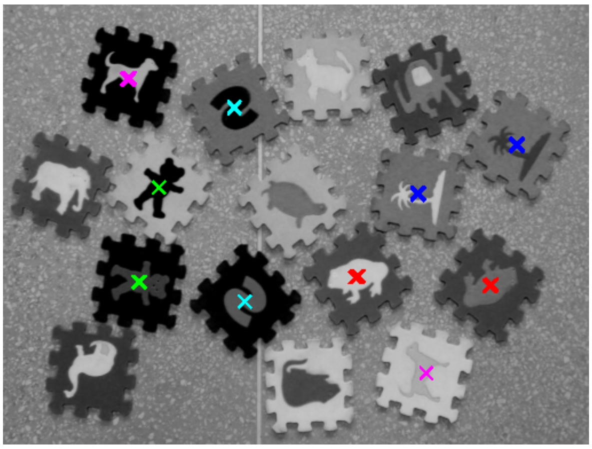
Goal: Find a grayscale query image Q in another image to analyze A, invariant to RSTBC:

- Rotation
- Scaling
- Translation
- Brightness
- Contrast

The proposed solution: Ciratefi (Circular, Radial and Template matching Filter) consists of three cascaded filters. Each filter excludes pixels that have no chance of matching the template.

Examples:





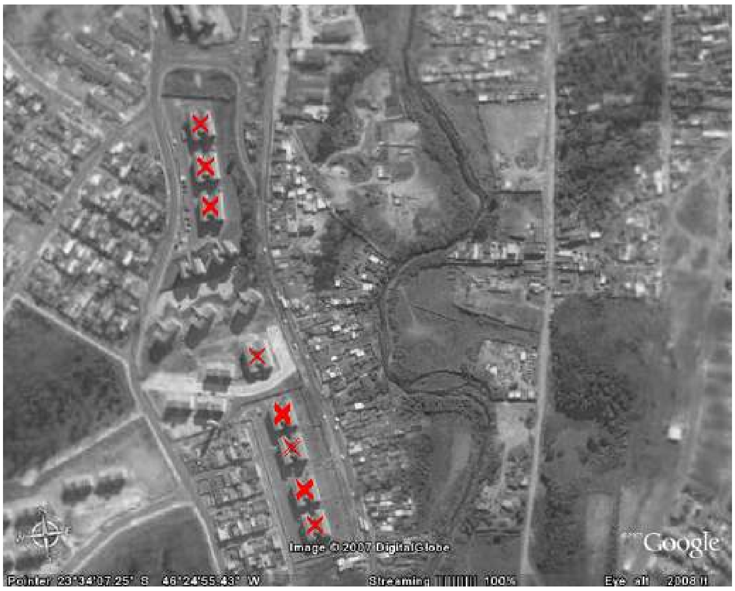


Image © 2007 DigitalGlobe
Google
Pointer 23°34'07.25" S 46°24'55.43" W
Streaming 100%
Eye alt 2008 ft

Exemplos em:

c:\haepi\algpi\rstmatch\rstmatch\ciratefi2014\1instance
rodar runc.bat e rung.bat

c:\haepi\algpi\rstmatch\rstmatch\ciratefi2014\2instances
rodar runc.bat e rung.bat

Agora, roda com OpenMP.

2.1 Brightness/contrast-invariant template matching

We use the correlation coefficient r_{xy}, because it is invariant to brightness and contrast.

There is a matching if $r_{xy} > t_B$ (or $|r_{xy}| > t_B$, to allow matching negative template instances).

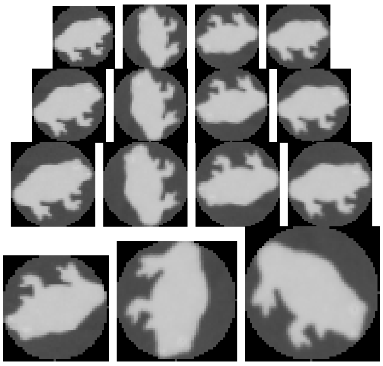
We assume that the correlation is zero if the brightness correction γ or the contrast correction β is above some threshold.

Brightness/contrast-aware correlation:

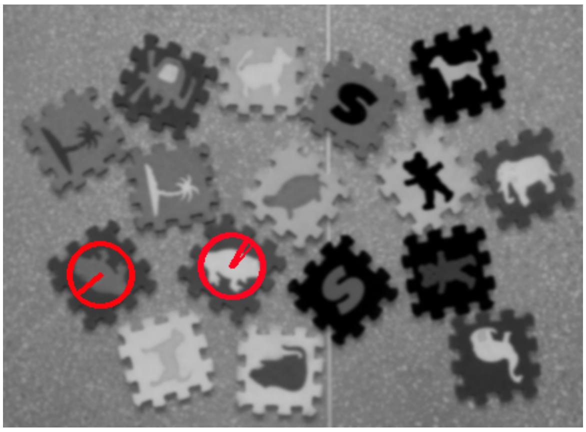
$$\mathrm{Corr}(\mathbf{x},\mathbf{y}) = \begin{cases} 0, \text{ if } |\beta| \le t_\beta, 1/t_\beta \le |\beta| \text{ or } |\gamma| > t_\gamma \\ r_{\mathbf{xy}}, \text{ otherwise} \end{cases}$$

2.2 Brute force RSTBC-invariant template matching

To achieve RSTBC-invariance, template Q must be rotated and scaled by all angles and factors and matched against all pixels of A. In practice, discrete sets of angles and scale factors are used. Brute force template matching is too slow.



Some of the scaled and rotated query images.



RSTBC-invariant brute-force template matching takes 2 hours, while Ciratefi takes 8 seconds.

3. Circular sampling filter (Cifi)

Ciratefi consists of 3 cascaded filters: Cifi, Rafi and Tefi.

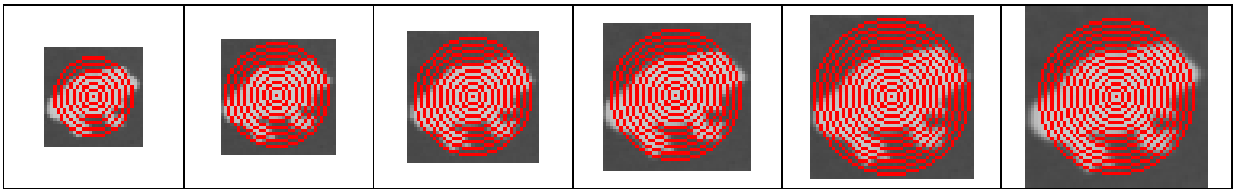
Cifi uses the projections of A and Q on a set of rings to detect the pixels that have chance of matching the template, named “first grade candidate pixels” and their scale factors.

Circular sampling *Cis* is the average grayscale of the pixels of an image B situated at distance r from pixel (x,y):

$$\mathrm{Cis}_B(x,y,r) = \frac{1}{2\pi r}\int_0^{2\pi} B(x + r\cos\theta, y + r\sin\theta)\, d\theta$$

C_Q is a matrix of multi-scale, rotation-invariant features of Q:

$$C_Q[i,k] = \mathrm{Cis}_{Q_i}(x_0, y_0, r_k)\,,\ 0 \le i < n \ \text{ and }\ 0 \le k < l$$



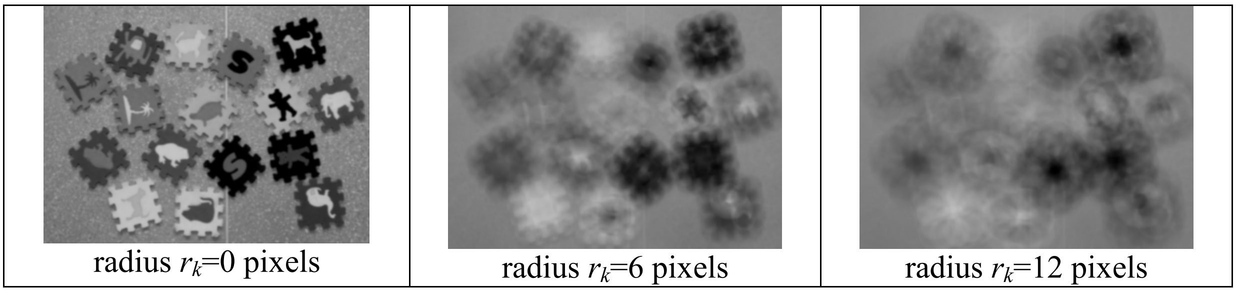
The average grayscales on red circles form the matrix C_Q.

Ex: $$C_Q[i,k] = \begin{bmatrix} 200 & 200 & 200 & 200 & 180 & 160 & 140 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ 200 & 200 & 200 & 200 & 200 & 180 & 160 & 140 & 120 & -1 & -1 & -1 & -1 & -1 \\ 200 & & & & & & & & & & & & & -1 \\ 200 & & & & & \cdots & \cdots & \cdots & \cdots & & & & & -1 \\ 200 & & & & & & & & & & & & & 70 \\ 200 & 200 & 200 & 200 & 200 & 200 & 200 & 200 & 180 & 160 & 140 & 120 & 100 & 80 \end{bmatrix}$$

C_A is multi-scale 3-D image with rotation-invariant features of A:

$$C_A[x,y,k] = \mathrm{Cis}_A(x,y,r_k),\quad 0 \le k < l \ \text{ and }\ (x,y) \in \mathrm{domain}(A)$$

Ex: $C_A[x,y,k] = \begin{bmatrix}100 & 100 & 100 & 100 & 100 & 90 & 80 & 70 & 60 & 30 & 240 & 30 & 200 & 70\end{bmatrix}$



radius r_k=0 pixels

radius r_k=6 pixels

radius r_k=12 pixels

CisCorr is the best correlation of circular samplings C_Q and C_A:

$$\mathrm{CisCorr}_{A,Q}(x,y) = \operatorname*{MAX}_{i=0}^{n-1}\Big[\big|\mathrm{Corr}(C_Q[i], C_A[x,y])\big|\Big]$$

A pixel (x,y) is a first grade candidate pixel if $\mathrm{CisCorr}_{A,Q}(x,y) \ge t_1$.

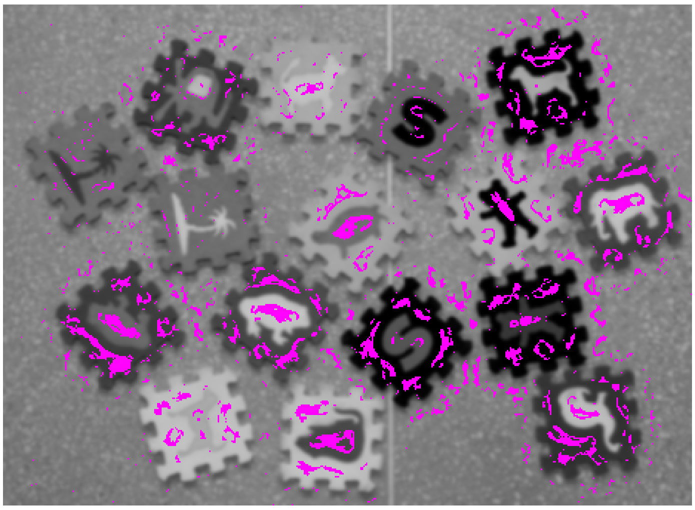
Note: -1 is not taken into account.

Note: Instead of using a fixed threshold t_1, it is better to consider a percentage of pixels (like 2%) as candidate pixels.

The probable scale *CisPS* is the scale of the best correlation:

$$\text{CisPS}_{A,Q}(x,y)=\underset{i=0}{\overset{n-1}{\text{ARGMAX}}}\left[\left|\text{Corr}(C_Q[i],C_A[x,y])\right|\right]$$

Ex: $\text{CisPS}_{A,Q}(x,y)=1$



The output of Cifi, with first grade candidate pixels in magenta. Each candidate pixel has an associated probable scale.

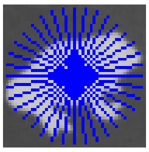
4. Radial sampling filter (Rafi)

Rafi uses the projections of *A* and *Q* on radial lines to upgrade some first grade pixels to the second grade. Radial sampling *Ras* is the average grayscale of the pixels of image *B* on a radial line:

$$\mathrm{Ras}_B^{\lambda}(x,y,\alpha)=\frac{1}{\lambda}\int_0^{\lambda} B(x+t\cos\alpha, y+t\sin\alpha)\,dt$$

Vector R_Q of radial samplings of *Q* is:

$$R_Q[j]=\mathrm{Ras}_Q^{r_{l-1}}(x_0,y_0,\alpha_j),\ 0\le j<m$$



Ex: $R_Q[j]=\begin{bmatrix}80 & 40 & 90 & 70 & 120 & 10 & 110 & 70 & 50\end{bmatrix}$

At each first grade pixel (x,y), *A* is radially sampled at its probable scale s_i:

$$R_A[x,y,j]=\mathrm{Ras}_R^{s_i r_{l-1}}(x,y,\alpha_j),\quad 0\le j<m \text{ and } (x,y)\in \mathrm{f\_gr\_cand}(A)$$

Ex: $R_A[x,y,j]=\begin{bmatrix}240 & 20 & 220 & 140 & 100 & 160 & 80 & 180 & 140\end{bmatrix}$

RasCorr is the largest correlation between the radial samplings R_Q and R_A:

$$\mathrm{RasCorr}_{A,Q}(x,y)=\overset{m-1}{\underset{j=0}{\mathrm{MAX}}}\left[\left|\,\mathrm{Corr}\big(R_A[x,y],\mathrm{cshift}_j(R_Q)\big)\right|\,\right],(x,y)\in \mathrm{f\_gr\_cand}(A)$$

where $cshift_j$ means circular shifting *j* positions.

A first grade pixel (x,y) is upgraded to the second grade if:

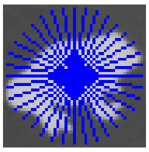
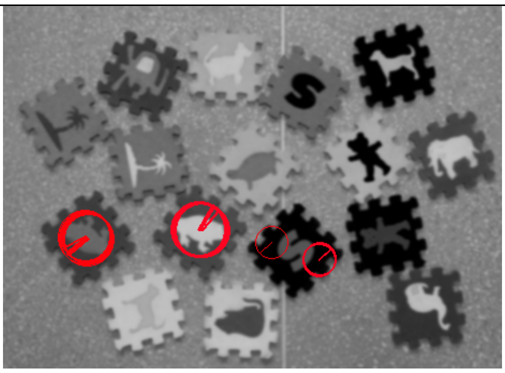
$$\mathrm{RasCorr}_{A,Q}(x,y)\ge t_2$$

Note: Instead of using a fixed threshold t_2, it is better to consider a percentage of pixels (like 1%) as candidate pixels.

The probable rotation angle at (x,y) is the angle that yields the largest correlation:

$$\mathrm{RasAng}_{A,Q}(x,y)=\ \overset{m-1}{\underset{j=0}{\mathrm{ARGMAX}}}\left[\left|\mathrm{Corr}\big(R_A[x,y],\mathrm{cshift}_j(R_Q)\big)\right|\right]$$

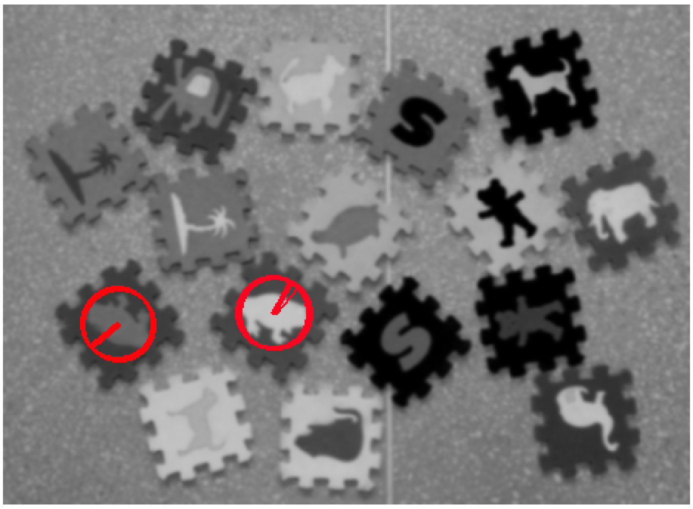
Ex: $\mathrm{RasAng}_{A,Q}(x,y)=5$

 Radial sampling lines	 The output of Rafi. Each second grade candidate pixel has an associated scale and angle.

5 Template matching filter (Tefi)

Tefi is the template matching applied at the second grade pixels, using the scales and angles determined by Cifi and Rafi.

If the correlation of the template matching is above t_3, Q is considered to be found.
Note: Instead of using a fixed threshold t_3, it is better to consider a number of matchings (like 2) separated by a minimal distance (like 50 pixels).



The output of Tefi.

Melhorias:
1) Escolher limiares “automaticamente”.
2) Usar pirâmide para melhorar invariância à escala.
3) Usar SSIM em vez de coeficiente de correlação

Ciratefi colorida:
1) Usa CSSIM para avaliar diferença.

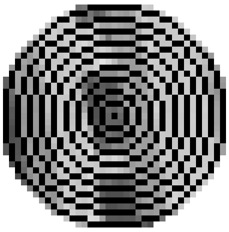
Forapro (Fourier coefficients of Radial Projections)

Exemplo em:
    c:\haepi\algpi\rstmatch\rstmatch\forapro2014
    rodar runtest-a runtest-b runtest-c
Ainda não usa OpenMP - daria para acelerar 4 vezes.

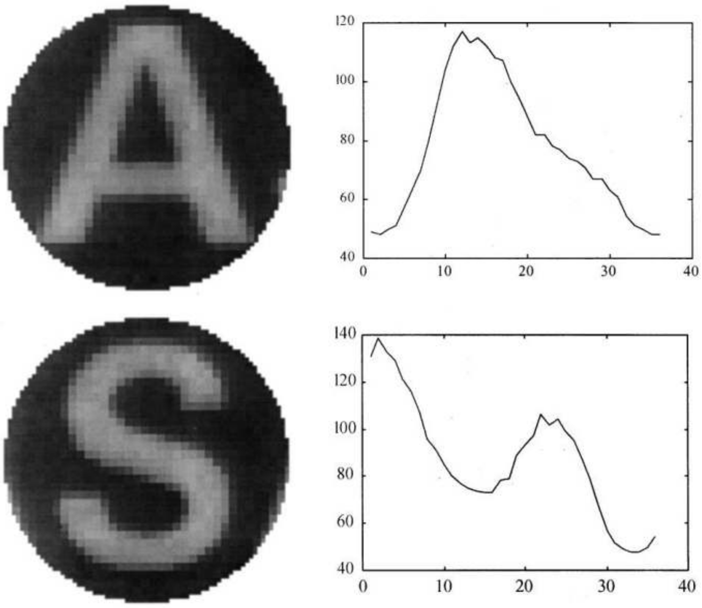
A idéia é calcular rapidamente algumas caracteríscas invariantes à rotação, brilho e contraste para cada pixel de A. Essas características serão usadas para fazer template matching com vários Qs.

Circular projection

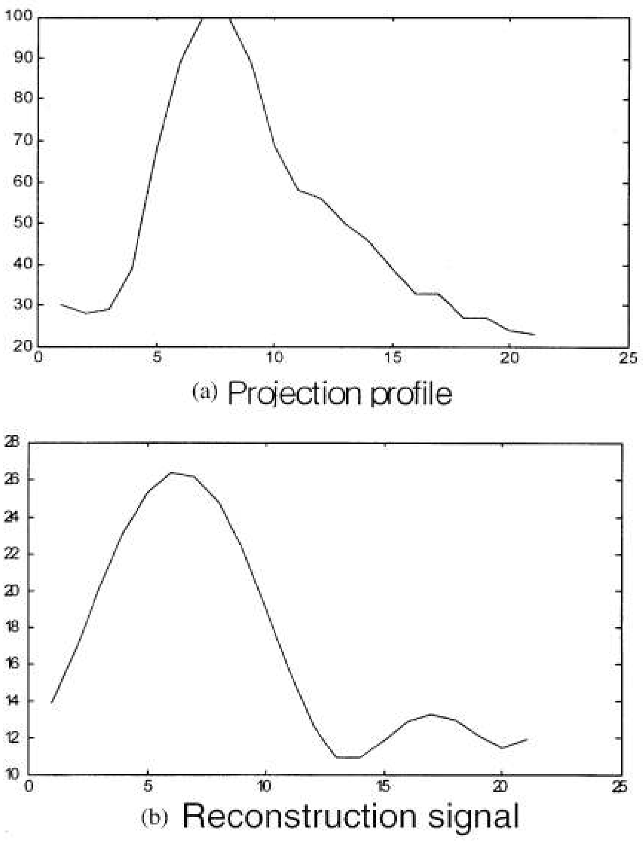
Choi and Kim (2000) have proposed to compute circular projections for each pixel (x,y) in A forming a one-dimensional vector $C(A(x,y))$. It computes the first low-frequency complex Fourier coefficients c_1, c_2, ... of $C(A(x,y))$, and uses them as rotation-invariant features. Actually, this technique computes the Fourier coefficients directly, by convolving A with appropriate kernels via FFT. The features of Q are compared with the features of each pixel $A(x,y)$ to select the pixels candidate for the matching. A secondary filter based on the rotation-invariant Zernike moments is used to further test the candidate pixels.



Circular projection is the mean grayscale on circular rings.



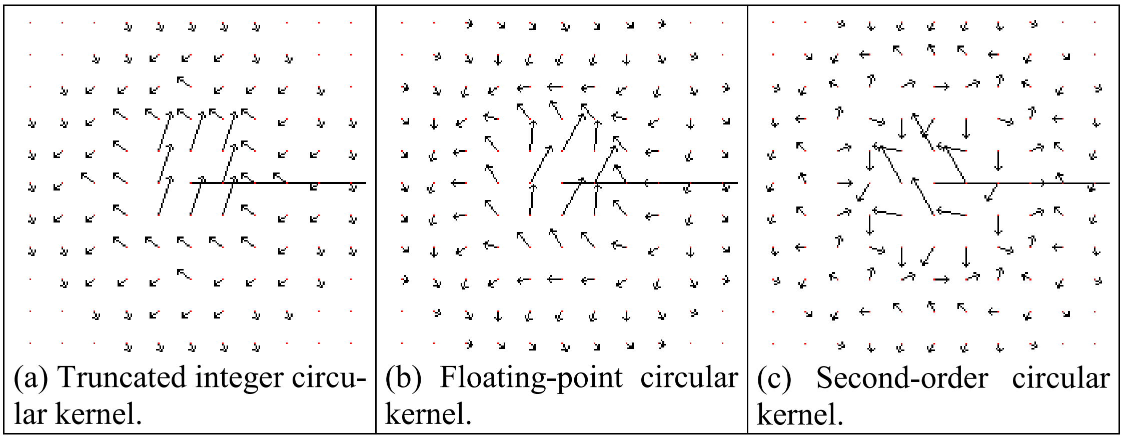
Projeção circular (retirado de Choi and Kim, 2000).



(a) Projection profile

(b) Reconstruction signal

Reconstrução do sinal usando primeiros 3 coeficientes de Fourier (retirado de Choi and Kim, 2000).



(a) Truncated integer circular kernel.

(b) Floating-point circular kernel.

(c) Second-order circular kernel.

Circular kernels.

Problema: Há várias figuras diferentes que levam à projeções circulares semelhantes.Tem que criar mais features invariantes à rotação, brilho e contraste.

Radial IDFT Coefficients

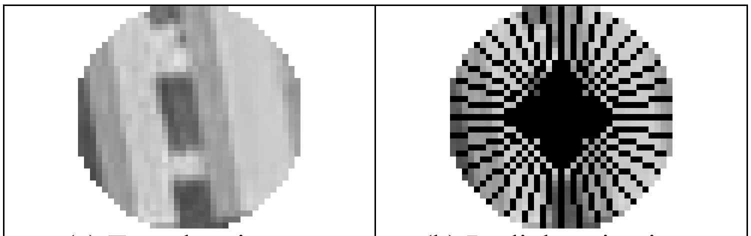
The radial projection $R_\alpha^\lambda(A(x,y))$ is the average grayscale of the pixels of *A* located on the radial line with one vertex at (*x*,*y*), length λ and inclination α:

$$R_\alpha^\lambda(A(x,y)) = \frac{1}{\lambda}\int_0^\lambda A\big(x + t\cos(\alpha), y + t\sin(\alpha)\big)dt \quad . \tag{1}$$

In practice, a sum must replace the integral. The vector of *M* discrete radial projections at pixel *A*(*x*,*y*) with radius λ is obtained by varying the angle α:

$$R_{A(x,y)}^\lambda[m] = R_{2\pi m/M}^\lambda(A(x,y)), \quad 0 \le m < M \quad . \tag{2}$$

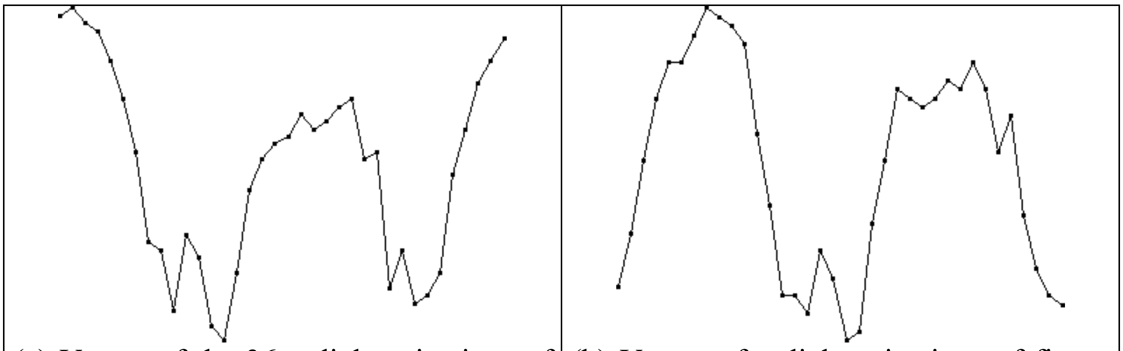
Figures 1(b) and 2(a) depict *M*=36 radial projections at the central pixel of figure 1(a).



(a) Template image

(b) Radial projections

Fig. 1: A radial projection is the mean grayscale on a radial line.



(a) Vector of the 36 radial projections of figure 1(a).

(b) Vector of radial projections of figure 1(a) rotated 60°

Fig. 2: Rotation of the image causes circular shifting in the vector of radial projections.

Vector of radial projections $R_{A(x,y)}^\lambda[m]$ characterizes the neighborhood of *A*(*x*,*y*) of radius λ. If *A* rotates, then this vector shifts circularly (figure 2).

The *k*-th Fourier coefficient of a vector of radial projections *R* is (we omit indices *A*(*x*,*y*) and λ):

$$r[k] = \sum_{m=0}^{M-1} R[m]\exp(-j2\pi km/M), \quad 0 \le k < M \; . \tag{3}$$

The Fourier coefficients of a vector of radial projections can be computed directly convolving *A* with an appropriate kernel *K*, without explicitly calculating the radial projections. Figure 3(a) depicts the "sparse DFT kernel" *K* (with *M*=8 angles) such that the convolution $A * \breve{K}$ yields the first Fourier coefficient of the radial projections (where $\breve{K}(x,y) = K(-x,-y)$ is the double reflection of *K*):

$$(A * \breve{K})(x,y) = \sum_p \sum_q A(p,q)\breve{K}(x-p, y-q) = \sum_p \sum_q A(p,q)K(p-x, q-y) \; . \tag{4}$$

The “dense DFT kernel” (figure 3(b)) can be used. The non-zero elements of this kernel are defined as:

$$\Re_k[x,y] = \exp(-jk\angle(x+yj)), \tag{6}$$

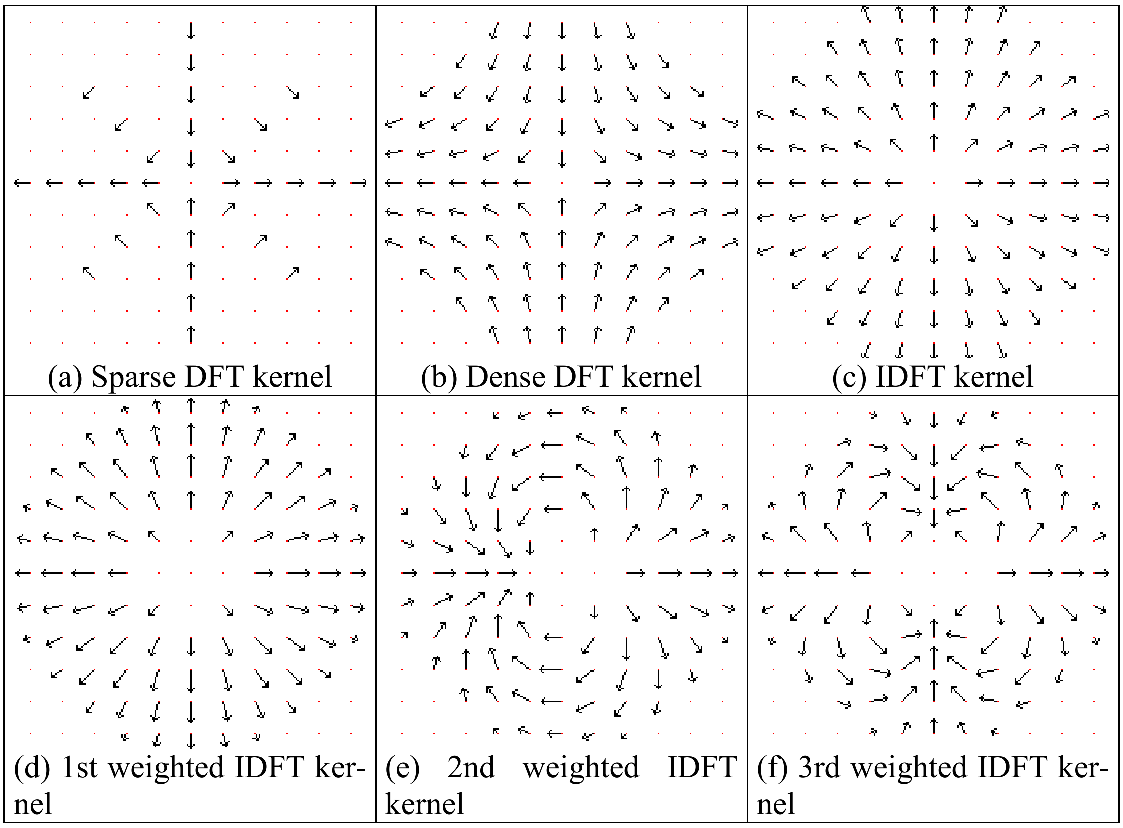
where k is the order of the Fourier coefficient and $\angle(.)$ is the angle of the complex number.

Using the inverse DFT, the result of the convolution acquires a meaning: it becomes analogous to the gradient.

In order to make the kernel more “stable,” we assign less weight to the pixels in the outer and central regions (figure 3(d)). The resulting radial kernel is:

$$\Re_k[x,y] = \sqrt{r(\lambda - r)}\,\exp(jk\angle(x+yj)), \tag{7}$$

where $r = \sqrt{x^2 + y^2}$ and λ is the radius of the kernel. In order to make the weights sum to one, all weights may be divided by the sum of the weights.



(a) Sparse DFT kernel

(b) Dense DFT kernel

(c) IDFT kernel

(d) 1st weighted IDFT kernel

(e) 2nd weighted IDFT kernel

(f) 3rd weighted IDFT kernel

Fig. 3: Radial kernels

Canonical Orientation

The "canonical orientation," is the first radial angle $\angle r_1$. If A rotates, canonical orientation $\angle r_1(A(x,y))$ rotates by the same angle. Brightness/contrast change does not alter the canonical orientation.

Rotation-Invariant "Vector of Radial Magnitudes"

Radial magnitudes are invariant to rotation and brightness.

The ratios between radial magnitudes are invariant to contrast because a contrast alteration multiplies by the same factor all the radial coefficients. We use the following vector of radial magnitudes v_{rm}:

$$v_{rm} = \upsilon\left[|r_1|, |r_2|, \ldots, |r_K|\right], \tag{10}$$

where υ means L^1-versor. This is invariant to rotation, brightness and contrast.

Rotation-Invariant "Vector of Radial Angles"

If A rotates θ radians, then the vector of radial projections $R^{\lambda}_{A(x,y)}[m]$ shifts circularly θ radians and the k-th radial coefficient $r_k(A(x,y))$ is multiplied by $\exp(jk\theta)$. A brightness or contrast change does not alter $\angle r_k$. Thus, the difference between $\angle r_k$ and $k\ \angle r_1$ is rotation- and brightness/contrast-invariant. We call this feature "difference of radial angles" k and 1:

$$\mathrm{dra}_k = \mathrm{mod}(\angle r_k - k\ \angle r_1, 2\pi), \quad k \geq 2\,. \tag{13}$$

This feature is computed modulo 2π, because it is an angle. We packed the dra's in vector of radial angles v_{ra}:

$$v_{ra} = \left[\mathrm{dra}_2, \mathrm{dra}_3, \ldots, \mathrm{dra}_K\right]. \tag{14}$$

We define the distance Λ between two v_{ra}'s as the weighted average of the angle differences:

$$\begin{aligned}
&\Lambda\left[v_{ra}(A(x,y)), v_{ra}(T(x_o,y_o))\right] = \\
&\quad (w_2/(\pi w_t))\ \Phi\left[\mathrm{dra}_2(A(x,y)), \mathrm{dra}_2(T(x_o,y_o))\right] + \\
&\quad (w_3/(\pi w_t))\ \Phi\left[\mathrm{dra}_3(A(x,y)), \mathrm{dra}_3(T(x_o,y_o))\right] + \\
&\quad \cdots + \\
&\quad (w_K/(\pi w_t))\ \Phi\left[\mathrm{dra}_K(A(x,y)), \mathrm{dra}_K(T(x_o,y_o))\right]
\end{aligned} \tag{15}$$

where $w_k = 1/k\ (2 \leq k \leq K)$, $w_t = w_2 + w_3 + \ldots + w_K$ and Φ is the difference of angles.

Rotation-Invariant "Vector of Circular Features"

We will use DFT coefficients of the circular projections, together with the radial features. The resulting circular kernel is:

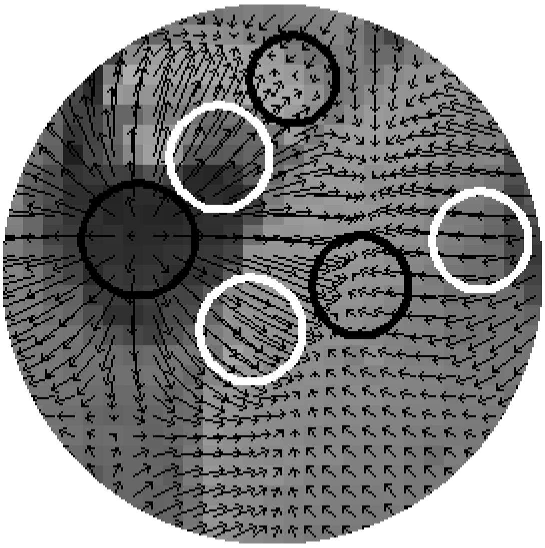
$$\Im_l[x,y] = \begin{cases} \dfrac{1}{2\pi r}\exp\left(\dfrac{jlr}{\lambda}\right), & \text{if } r > 0 \\ 0.73, & \text{if } r = 0 \end{cases} \tag{16}$$

where $r = \sqrt{x^2 + y^2}$ and λ is the radius of the kernel. In order to make the weights sum to one, all weights must be divided by the sum of the weights.

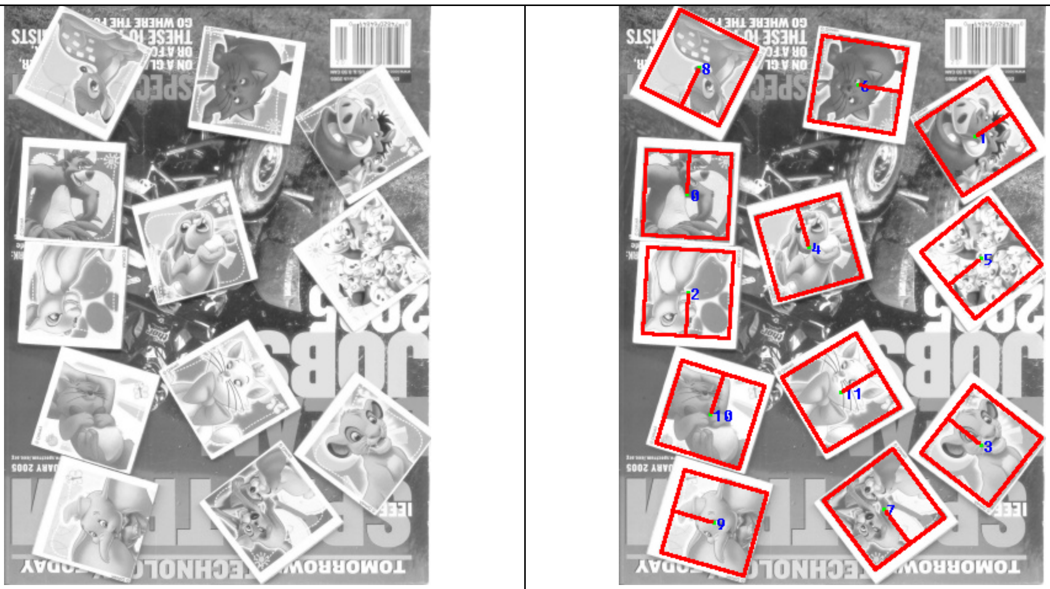
We use the following "vector of circular features" that takes into account the real and imaginary components of all circular coefficients up to degree L:

$$\text{vcf} = \upsilon\left[\text{re}(c_1), \text{im}(c_1), \text{re}(c_2), \text{im}(c_2), \ldots, \text{re}(c_L), \text{im}(c_L)\right], \tag{17}$$

where υ means L^1-versor, and "re" and "im" are respectively the real and imaginary parts of the complex number.



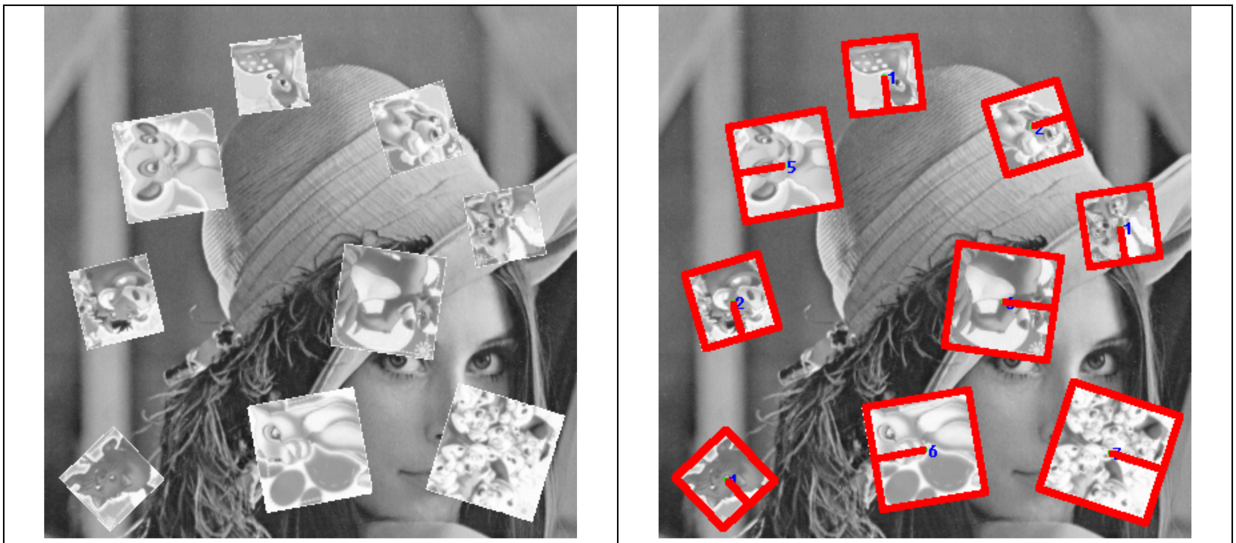
Canonical orientation



(a) One of the eight images to be analyzed.

(b) The output of our algothm.

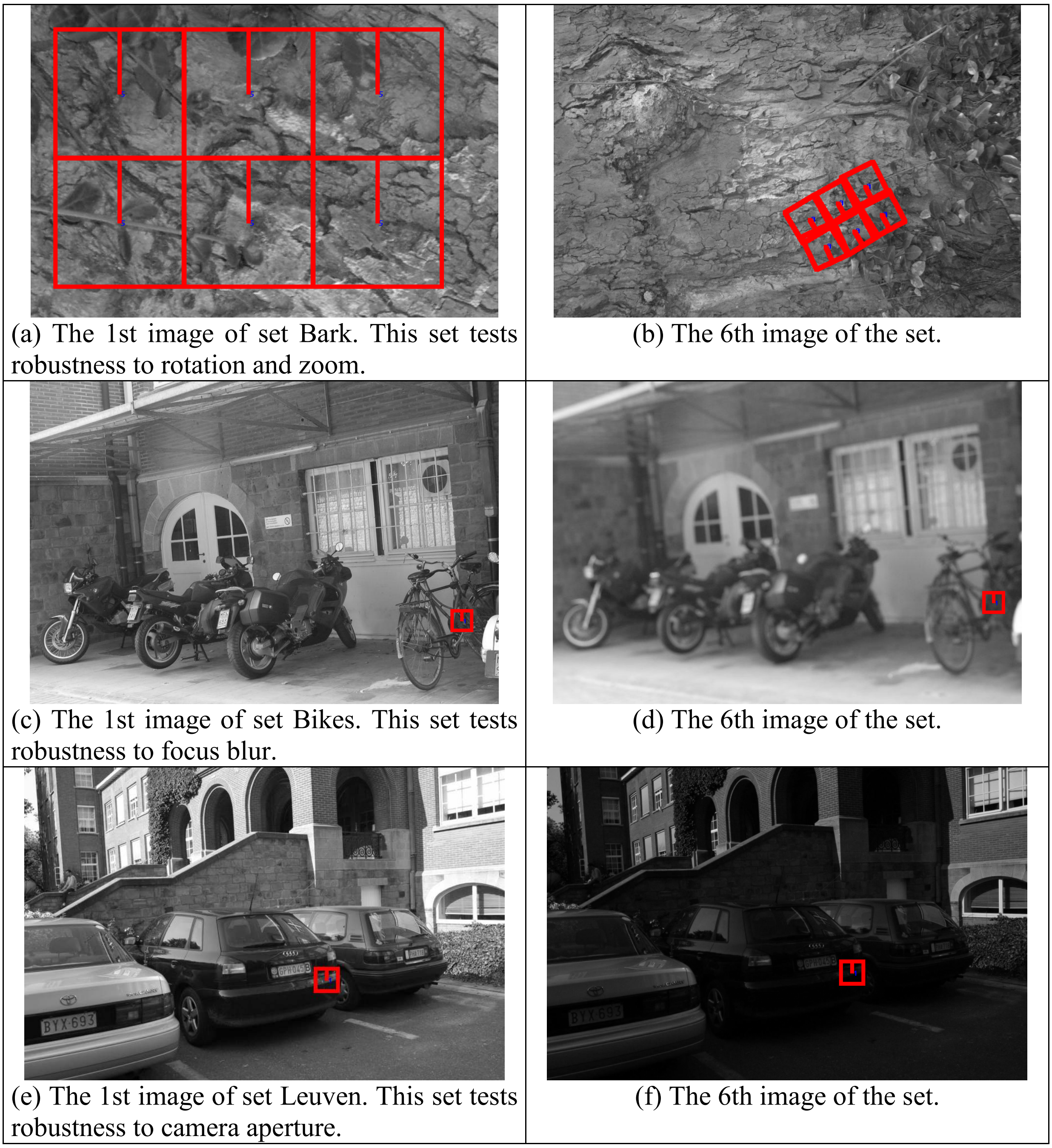
Fig. 7: One of the images used to test Forapro, with the respective output.



(a) An image to be analyzed.

(b) The output of our algorithm.

Fig. 9: A test to verify the robustness to scaling.



(a) The 1st image of set Bark. This set tests robustness to rotation and zoom.

(b) The 6th image of the set.

(c) The 1st image of set Bikes. This set tests robustness to focus blur.

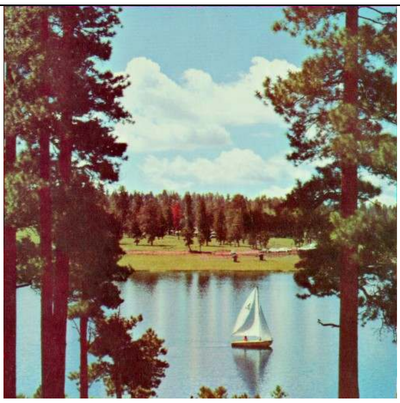
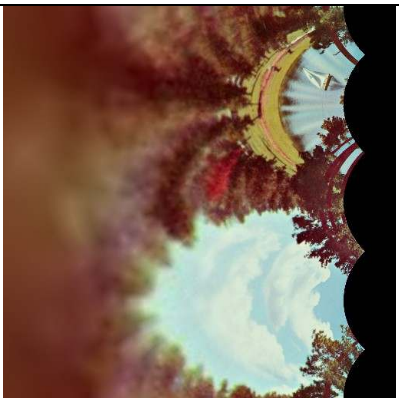
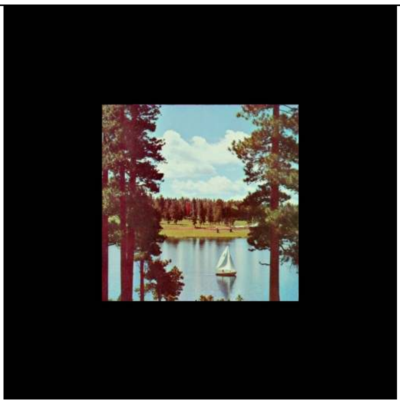
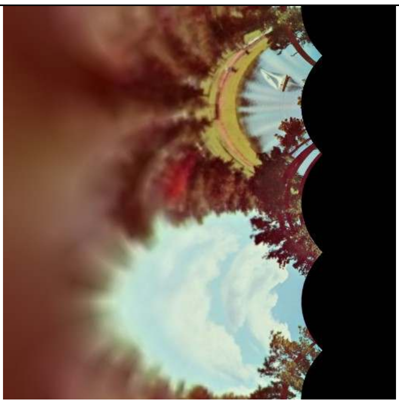
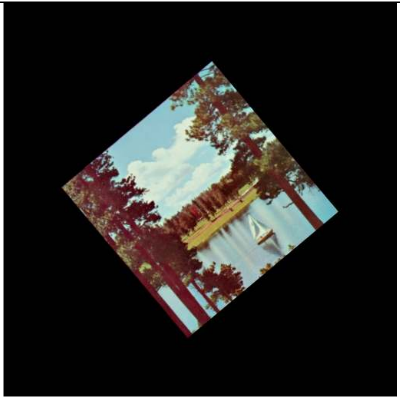
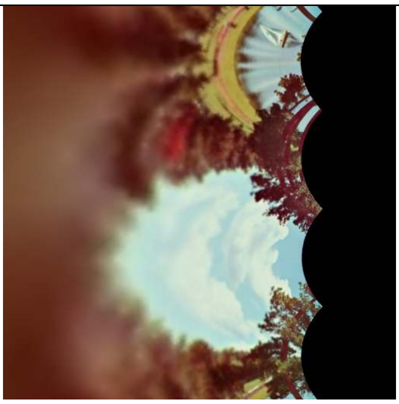
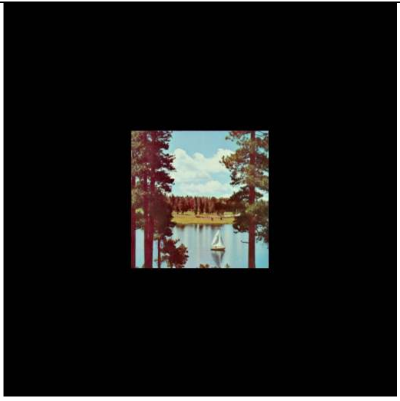
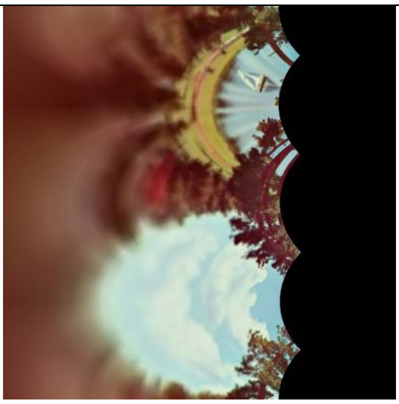
(d) The 6th image of the set.

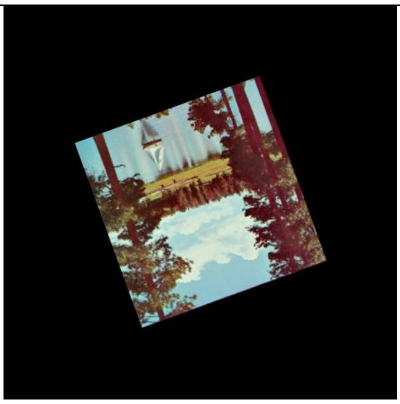
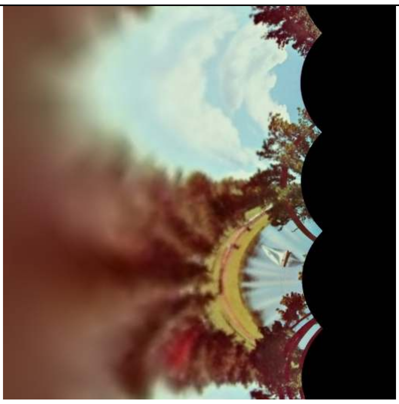
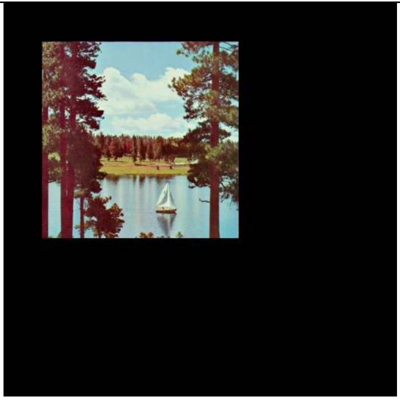
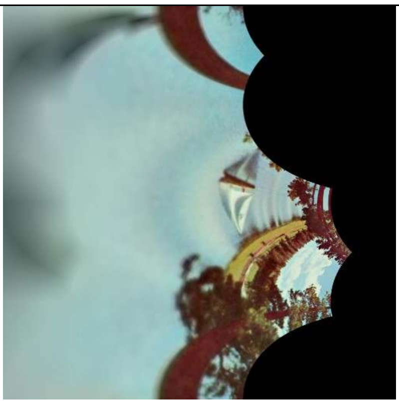
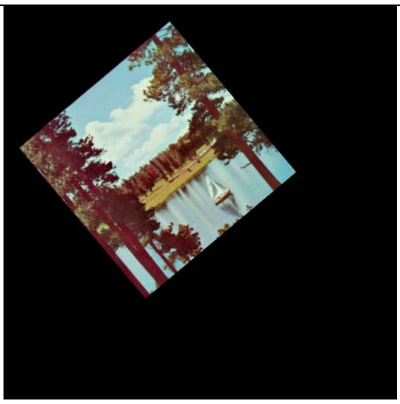
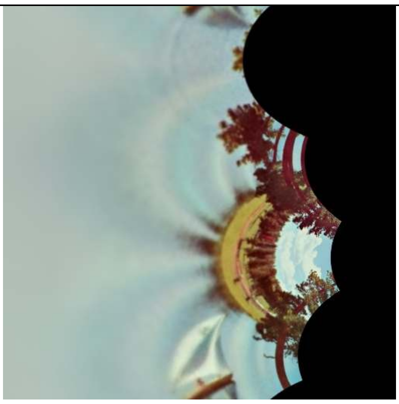
(e) The 1st image of set Leuven. This set tests robustness to camera aperture.

(f) The 6th image of the set.

Fig. 11: Some of the images used to compare SIFT with Forapro. Both SIFT and Forapro localized correctly 300 queries in set Bark. SIFT made substantially more errors than Forapro when the local textures could not be easily extracted, like in sets Bikes and Leuven.

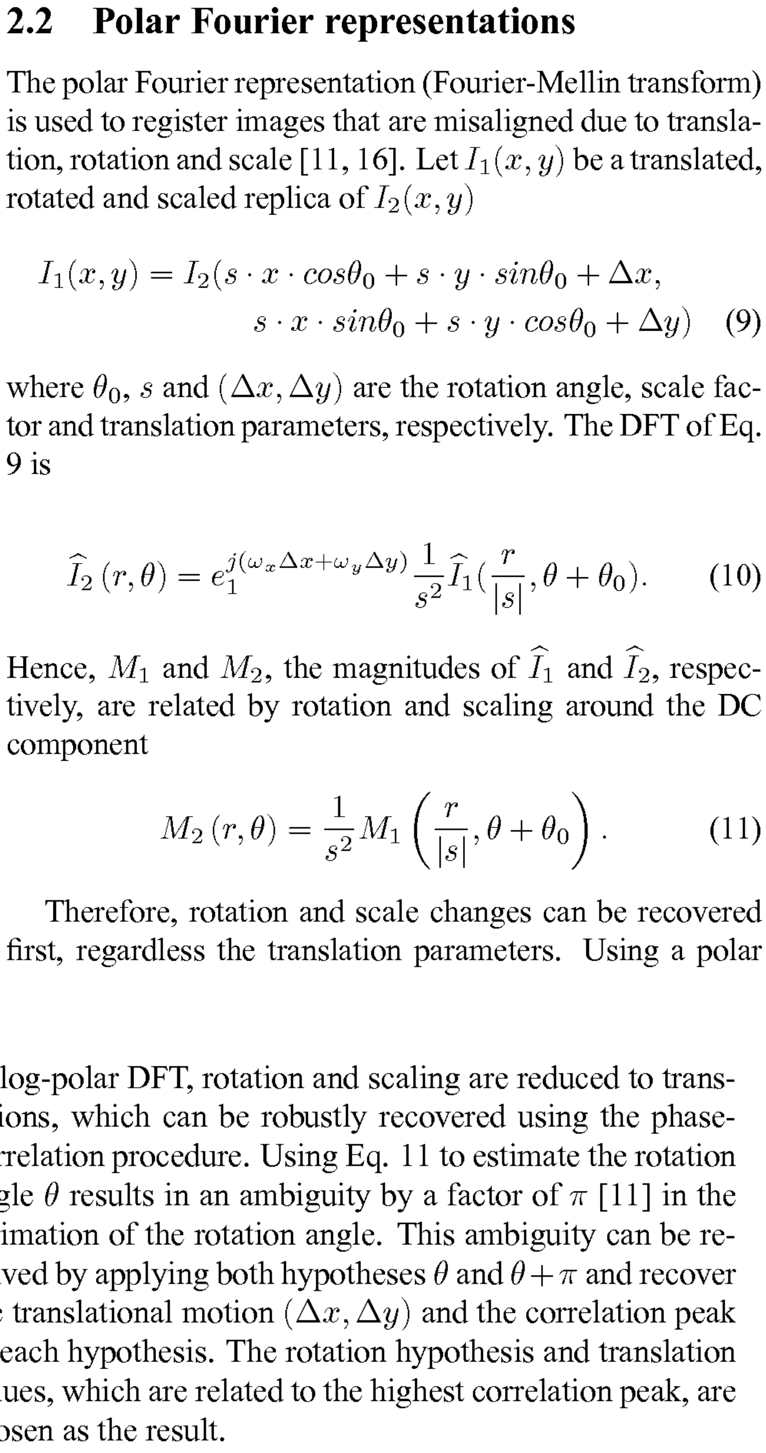
Log-polar no domínio espacial

 s00	 l00 – observe o barco
 s01 – mudança de escala em torno do centro	 l01 – o barco não mudou de forma
 s03 – escala e rotação em torno do centro	 l03 – o barco não mudou de forma
 s05 – escala em torno do centro	 l05 – o barco não mudou de forma

 s06 – escala e rotação em torno do centro	 l06 – o barco não mudou de forma
 s02 – escala e deslocamento	 l02 – o barco mudou de forma
 s04 – escala, rotação e deslocamento	 l04 – o barco mudou de forma

Log-polar DFT

Registro de imagens usando log-polar DFT:
As idéias deste resumo estão na seção 2.2 do artigo
[Keller, 2005] Y. Keller, A. Averbuch, M. Israeli, "Pseudo-polar based estimation of large translations rotations and scalings in images," IEEE Workshop on Motion and Video Compuyting, 2005.



2.2 Polar Fourier representations

The polar Fourier representation (Fourier-Mellin transform) is used to register images that are misaligned due to translation, rotation and scale [11, 16]. Let $I_1(x,y)$ be a translated, rotated and scaled replica of $I_2(x,y)$

$$I_1(x,y) = I_2(s \cdot x \cdot cos\theta_0 + s \cdot y \cdot sin\theta_0 + \Delta x, \\ s \cdot x \cdot sin\theta_0 + s \cdot y \cdot cos\theta_0 + \Delta y) \quad (9)$$

where θ_0, s and $(\Delta x, \Delta y)$ are the rotation angle, scale factor and translation parameters, respectively. The DFT of Eq. 9 is

$$\widehat{I_2}(r,\theta) = e_1^{j(\omega_x \Delta x + \omega_y \Delta y)} \frac{1}{s^2} \widehat{I_1}(\frac{r}{|s|}, \theta + \theta_0). \quad (10)$$

Hence, M_1 and M_2, the magnitudes of $\widehat{I_1}$ and $\widehat{I_2}$, respectively, are related by rotation and scaling around the DC component

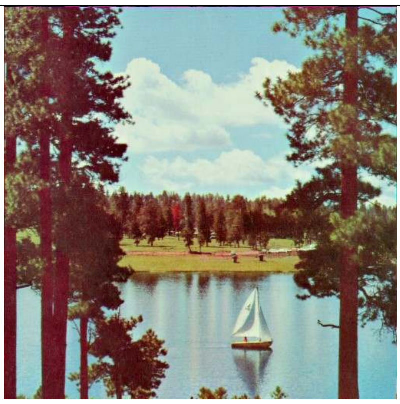
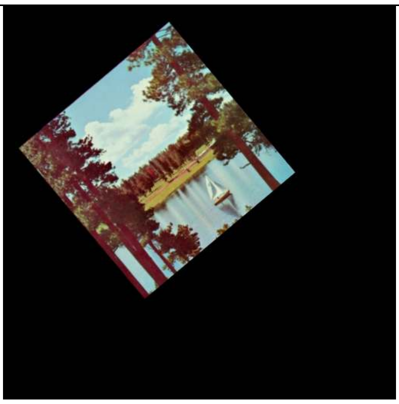
$$M_2(r,\theta) = \frac{1}{s^2} M_1 \left(\frac{r}{|s|}, \theta + \theta_0 \right). \quad (11)$$

Therefore, rotation and scale changes can be recovered first, regardless the translation parameters. Using a polar or log-polar DFT, rotation and scaling are reduced to translations, which can be robustly recovered using the phase-correlation procedure. Using Eq. 11 to estimate the rotation angle θ results in an ambiguity by a factor of π [11] in the estimation of the rotation angle. This ambiguity can be resolved by applying both hypotheses θ and $\theta + \pi$ and recover the translational motion $(\Delta x, \Delta y)$ and the correlation peak of each hypothesis. The rotation hypothesis and translation values, which are related to the highest correlation peak, are chosen as the result.

O problema é que o esquema acima só pode ser aplicado se as duas imagens estiverem quase inteiras, isto é, para registro.

Não é possível usá-lo para procurar template *T* pequeno em imagem *A* grande.

Só é possível usá-lo se T e A estiverem mais ou menos inteiros. T pode ter alguma parte de A ausente. Mas este pedaço deve estar coberto por zero (preto).

 s00	 s04 – escala, rotação e deslocamento

Consegue-se achar os parâmetros de rotação, escala e translação se fora da imagem estiver preto. Caso contrário, ruído diminuirá o pico da correlação.

Granulometria baseada em correlação

[Tenho que escrever a explicação]

A granulometria baseada em correlação para objetos granulares circulares é apresentada em:

- [Ri10] Ricardo Hitoshi Maruta, Hae Yong Kim, Danilo Roque Huanca and Walter Jaimes Salcedo, "A New Correlation-Based Granulometry Algorithm with Application in Characterizing Porous Silicon Nanomaterials," *ECS Transactions*, vol. 31, pp. 273-280, 2010.

A granulometria baseada em correlação para objetos granulares de qualquer formato é apresentada em:

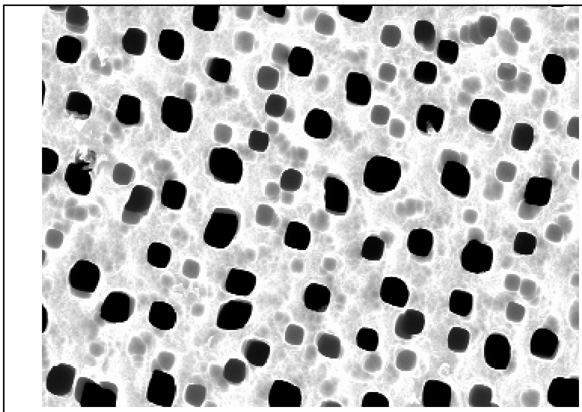
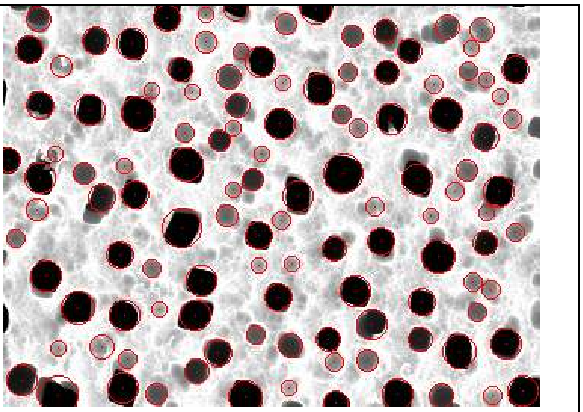
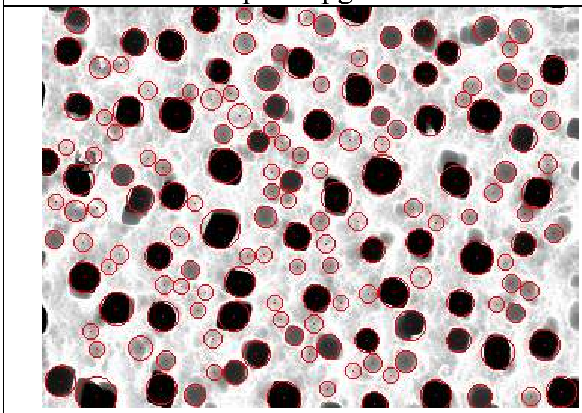
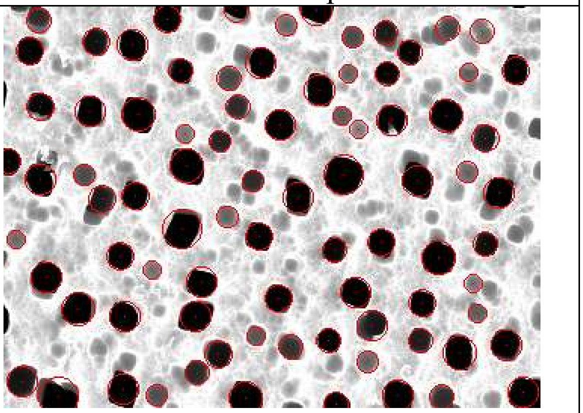
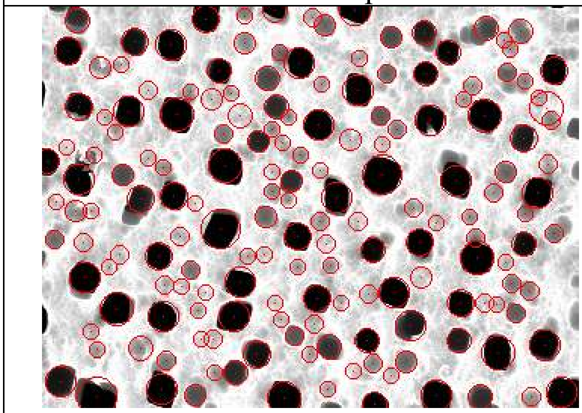
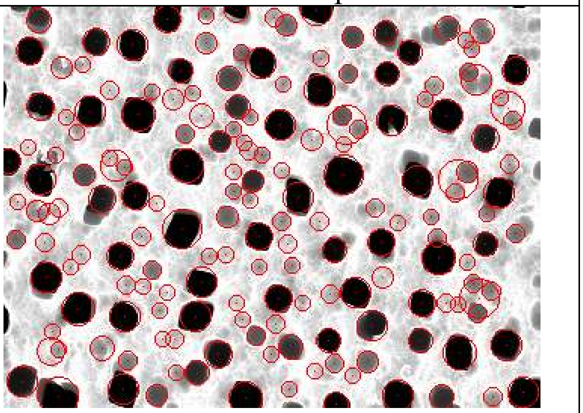
- [Ri15] H. Y. Kim, R. H. Maruta, D. R. Huanca and W. J. Salcedo, "Correlation-Based Multi-Shape Granulometry with Application in Porous Silicon Nanomaterial Characterization," *Journal of Porous Materials*, April 2013, Vol. 20, No. 2, pp 375-385.

Os programas estão disponíveis em:

http://www.lps.usp.br/hae/granul/index.html

Executando:

```
c:\diretorio>granul corrcirc poros.pgm poros.hol 9 5 6
```

 poros.pgm	 Corr=8 Sobrep=40
 Corr=4 Sobrep=40	 Corr=16 Sobrep=40
 Corr=4 Sobrep=75	 Corr=4 Sobrep=99