An equivalent definition of the histogram of forces: Theoretical and algorithmic implications

Jingbo Ni, Pascal Matsakis*

Computing and Information Science, University of Guelph, Guelph, ON, N1G 2W1, Canada

* Corresponding author.
E-mail address: pmatsaki@uoguelph.ca (P. Matsakis).

Abstract

The relative position between two 2-D spatial regions is often represented quantitatively by a force histogram. In the case of raster data, force histograms are usually computed in $\mathcal{O}(KN\sqrt{N})$ time, where N is the number of pixels in the image and K is the number of directions in which forces are considered. When the regions are defined as fuzzy sets instead of crisp sets, the complexity also depends on the number M of possible membership degrees. In this paper, we show that the force histogram can be defined in a completely different but equivalent way, one which leads to an $\mathcal{O}(N\log N)$ algorithm, with complexity independent of K and M. Moreover, the equivalent definition is better adapted to the solving of theoretical issues. We use it here to determine the behavior of the force histogram towards any invertible affine transformation.

Keywords: relative position; histogram of forces; spatial correlation; histogram of angles; spatial relationships.

1. Introduction

Representations of spatial configurations between physical entities play an important role in many applications, like geographic information systems [1], robot navigation [2] and content-based image retrieval [3]. For example, in order to reach its destination efficiently and safely, an autonomous robot must capture knowledge about its spatial location (through its navigation system); this is often done by generating representations of the relative positions between the robot and landmarks in the environment [4]. Numerous approaches have been proposed to quantitatively represent the relative position between two spatial regions. The angle histogram [5][6] has had significant influence in the field. It is appealing for its simplicity, but suffers from many weaknesses (e.g., excessively long processing times, anisotropy, inability to handle vector data). In [7], Matsakis introduced a generic, more complex representation: the F-histogram. There are different types of F-histograms [8]. Most work, however, has been done on a particular type called force histogram. The force histogram generalizes and overcomes the weaknesses of the angle histogram. It has been used in scene description [9], human-robot communication [10], for classification problems [11], content-based retrieval [12], etc. See [13] for a review of work on and applications of the force histogram.

The spatial regions considered here are fuzzy regions in a 2-D raster image. Fuzzy sets make it possible to encapsulate photometric information, or information regarding the imprecision or the uncertainty in the spatial extent of the regions. In [14], for example, the authors manipulate the 3-D data contained within a LADAR (Laser Radar) range image and create a 2-D version of the scene as seen from above; the areas of uncertainty in the transformed view are then filled in with fuzzy regions from which force histograms are computed. Processing times are in $\mathcal{O}(N^2)$ for angle histograms [5][6] and in $\mathcal{O}(KM^2N\sqrt{N})$ for force histograms [13][15], where N is the number of pixels in the image, K is the number of directions in which forces are considered, and M is the number of membership degrees in the fuzzy regions. In a preliminary work [16], we expressed the idea that the complexity of force histogram computation could be lowered and made independent of K and M if an intermediate computation step was used. Histograms similar (but not identical) to force histograms could be obtained in $\mathcal{O}(N\log N)$. In the present paper, we take the idea a leap further and bring it to fruition. We show that the histogram of forces can be defined in a completely different but equivalent way, with major theoretical and algorithmic implications. The equivalent definition allows us to determine the behavior of the force histogram towards any invertible affine transformation. This is an issue of prime importance—which was, until now, only partially solved [17]—because it is related to the design of affine-invariant descriptors. We are also able to introduce a general $\mathcal{O}(N\log N)$ algorithm for force histogram computation. This is another important result. The traditional algorithm leads to excessively long processing times when forces are computed in more than a few hundred directions, when the number of membership degrees is not single digit, or when the regions are fractal-like regions. We can now state that these weaknesses are overcome.

The force histogram is reviewed in Section 2. The new, equivalent definition of the histogram is presented in Section 3. In Section 4, we examine the behavior of the histogram towards affine transformations. In Section 5, the new algorithm for force histogram computation is introduced. Experiments, in Section 6, validate the theoretical analysis and demonstrate the efficiency of the algorithm. Conclusions and future work are given in Section 7.

2. Notation and background

The notation used throughout the paper is presented in Section 2.1. The traditional definition of the force histogram is given in Section 2.2. The existing approximation algorithms for force histogram computation in the case of raster data are described in Section 2.3.

2.1. Notation

Numbers, vectors, and points: $\Re$ is the set of real numbers and $m..n$ is the set of all integers between m and n inclusively. The nearest integer to $x\in\Re$ is $\lfloor x \rceil$. The symbol $\vec{\mathcal{P}}$ denotes the Euclidean vector plane. The norm of the vector $\vec{u}\in\vec{\mathcal{P}}$ is $|\vec{u}|$ and its direction is $\angle\vec{u}\in[0,2\pi)$. For any $\theta\in[0,2\pi)$, the symbols $\vec{\theta}$ and $\vec{\theta}^{\perp}$ denote the vectors such that $\angle\vec{\theta}=\theta$ and $(\vec{\theta},\vec{\theta}^{\perp})$ is a positively oriented orthonormal basis. See Fig. 1. The dot product of $\vec{u}$ and $\vec{v}$ is $\vec{u}\bullet\vec{v}$. The symbol $\mathcal{P}$ denotes the Euclidean affine plane. The origin, ω, is the point of coordinates (0,0). If $p\in\mathcal{P}$ is the point of coordinates (x,y) then $\lfloor p \rceil$ is the point of coordinates $(\lfloor x \rceil,\lfloor y \rceil)$. For any $\theta\in[0,2\pi)$ and any $t\in\Re$, the symbol $\Lambda_\theta(t)$ denotes the line $\{\omega + t\,\vec{\theta}^{\perp} + s\,\vec{\theta}\,\}_{s\in\Re}$. Also note that two points p and q define a vector $\overrightarrow{pq}$.

Crisp, fuzzy, and raster objects: An *object* is a nonempty fuzzy subset of $\mathcal{P}$. For any object A and any point p, the membership degree of p in A (i.e., the degree to which p belongs to A) is $A(p)\in[0,1]$. It is assumed that the support of A, i.e., $\{\,p\in\mathcal{P}\mid A(p)\neq 0\,\}$, is bounded. A is a *crisp* object iff (if and only if): $\forall p\in\mathcal{P},\, A(p)\in\{0,1\}$. It is a *raster* object iff: $\forall p\in\mathcal{P},\, A(p)=A(\lfloor p \rceil)$. A pixel is a crisp raster object with area 1.

Affine transformations: Affine transformations are mappings from $\mathcal{P}$ to $\mathcal{P}$ that preserve collinearity (all points lying on a line initially still lay on a line after transformation) and ratios of distances (e.g., the midpoint of a line segment remains the midpoint after transformation). Translations, reflections, rotations, dilations and stretches (Fig. 2) are examples of affine transformations. Any affine transformation *aff* can be written as the composition of a translation *tra* with a linear transformation *lin* (an affine transformation such that $lin(\omega)=\omega$). In other words,

$aff(p) = (tra \circ lin)(p) = tra(lin(p))$, for any point $p \in \mathcal{P}$. It is a common convention to see *tra* as a vector, *lin* as a 2×2 matrix, vectors as 2×1 matrices, etc. We can therefore write, e.g., $aff(p)=lin(p)+tra$ and $lin(\omega+\vec{u})=\omega+lin\cdot\vec{u}$, where + denotes point-vector addition and · denotes matrix multiplication. *aff* is invertible iff *lin* is invertible, i.e., iff det(*lin*)≠0, where det(*lin*) is the determinant of (the matrix) *lin*. In that case, the object A is transformed by *aff* into $aff[A]$, and $aff[A](p)=A(aff^{-1}(p))$. For example, $tra[A](p)=A(p+(-tra))$.

Fig. 1. Notation for numbers, vectors, and points.

Fig. 2. Examples of affine transformations. *(a)* Translation. *(b)* Orthogonal reflection about $\Lambda_\alpha(0)$. *(c)* ρ-angle rotation about ω. *(d)* Dilation with center ω and positive ratio λ. *(e)* X-axis orthogonal stretch with positive ratio k. In all cases, if o is the midpoint of p and q, and if p', o' and q' are the affine transformations of p, o and q, then o' is the midpoint of p' and q'.

2.2. Traditional definition of the histogram of forces

Consider two objects A and B. The points p and q are seen as particles that attract each other: q exerts on p an elementary force whose direction is $\angle\overrightarrow{pq}$ and whose magnitude is $A(p)B(q) \,/\, |\overrightarrow{pq}|^r$, where $r \in \Re$ is a constant (Fig. 3(a)). The *histogram of forces* F_r^{AB} is a representation of the position of B relative to A (Fig. 3(b)). It is a function from $[0,2\pi)$ into $\Re$. The value $\mathrm{F}_r^{AB}(\theta)$ is defined as the sum of the magnitudes of all the elementary forces in direction θ. When r=2, the force exerted by q on p obeys Newton's law of gravity: F_2^{AB} is called a *gravitational force histogram*. As another example, F_0^{AB} is a *constant force histogram*. Let us give a formal definition. A *histogram of forces* is a function $\mathrm{F}_r^{AB} \mid [0,2\pi) \rightarrow \Re$ defined by

$$\mathrm{F}_r^{AB}(\theta) = \int_{-\infty}^{+\infty} \mathrm{F}_r(A,B,t,\theta)\,\mathrm{d}t\,, \tag{1}$$

where

$$\mathrm{F}_r(A,B,t,\theta) = \int_{p\in\Lambda_\theta(t)}\int_{q\in\Lambda_\theta(t)} A(p)\, B(q)\, \varphi_r(\overrightarrow{pq}\bullet\vec{\theta})\, \mathrm{d}p\, \mathrm{d}q \qquad (2)$$

and

$$\varphi_r(k) = 1/k^r \text{ if the real number } k \text{ is positive and } \varphi_r(k)=0 \text{ otherwise.} \qquad (3)$$

In the case of raster data, the handling of fuzzy objects can be reduced to that of crisp objects, using the general double sum scheme introduced in [18]. The approach is recommended (since it leads to shorter processing times) when the number of possible membership degrees is small. An alternative scheme is the single sum scheme [5]. Equation (2), however, is then implicitly replaced with Eq. (4). See [7] (Chapter II, Section 6.1) and [15] for details.

$$\mathrm{F}_r(A,B,t,\theta) = \int_{p\in\Lambda_\theta(t)}\int_{q\in\Lambda_\theta(t)} \min\{A(p), B(q)\}\, \varphi_r(\overrightarrow{pq}\bullet\vec{\theta})\, \mathrm{d}p\, \mathrm{d}q \qquad (4)$$

It can be shown that the constant force histogram F_0^{AB} and the angle histogram are fundamentally equivalent. They are, nonetheless, different functions, computed in different ways, and F_0^{AB} offers stronger theoretical guarantees. In the end, the histogram of forces generalizes and supersedes the histogram of angles because of two unique characteristics: each object is seen as a surface (not as a discrete cloud of points) and distance information is explicitly taken into account (through r). See [7] (Chapter II, Section 5) and [15].

Fig. 3. The histogram of forces. *(a)* Non-raster objects can be handled. The darker the area, the higher the membership value. Notice the elementary force exerted by q on p. *(b)* Example.

2.3. Existing algorithms for force histogram computation

As far as we know, there exist three different algorithms for the computation of force histograms in the case of raster data. Practically, only a finite number of histogram values can be calculated. Only a finite number K of directions θ are, therefore, considered. A reasonable option is to pick a set of evenly distributed reference directions, such as $\{2\pi i/K\}_{i\in 0..K-1}$. For a given θ, the three algorithms approximate $\mathrm{F}_r^{AB}(\theta)$ (Eq. (1)) by a Riemann sum:

$$\mathrm{F}_r^{AB}(\theta)=\int_{-\infty}^{+\infty}\mathrm{F}_r(A,B,t,\theta)\,\mathrm{d}t\approx(\cos\theta)\sum_{-\infty}^{+\infty}\mathrm{F}_r(A,B,k\cos\theta,\theta)\,. \qquad (5)$$

The value $\mathrm{F}_r(A,B,k\cos\theta,\theta)$ is the sum of the elementary forces in direction θ exerted by the points of $B\cap\Lambda_\theta(k\cos\theta)$ on those of $A\cap\Lambda_\theta(k\cos\theta)$. For the sake of simplicity, let us assume that A and B are crisp, and that θ belongs to [0,π/4]. The line $\Lambda_\theta(k\cos\theta)$ is rasterized into a set of pixels using Bresenham's algorithm [19]. See Fig. 4(a). Some of these pixels belong to A, and their projection along the Y-axis onto $\Lambda_\theta(k\cos\theta)$ defines pairwise disjoint segments I_i. Some pixels belong to B and their projection defines segments J_j. We have $A\cap\Lambda_\theta(k\cos\theta)\approx\cup I_i$ and $B\cap\Lambda_\theta(k\cos\theta)\approx\cup J_j$. In other words,

$$\mathrm{F}_r(A,B,k\cos\theta,\theta)\approx\int_{p\in\cup I_i}\int_{q\in\cup J_j}\varphi_r(\overrightarrow{pq}\bullet\vec{\theta})\,\mathrm{d}p\,\mathrm{d}q=\sum_i\sum_j\int_{p\in I_i}\int_{q\in J_j}\varphi_r(\overrightarrow{pq}\bullet\vec{\theta})\,\mathrm{d}p\,\mathrm{d}q. \quad (6)$$

Compare Eq. (6) with Eq. (2). Symbolic calculation of the double integral over I_i and J_j yields a set of algebraic expressions that are hard-coded. This is the traditional algorithm [7][15]. In the worst-case scenario, A and B are fractal-like objects: the number of segments I_i and J_j may then be very high; the algorithm runs in $\mathcal{O}(KN\sqrt{N})$ time, where N denotes the number of pixels in the image. In the best-case scenario, A and B are convex objects: the total number of segments I_i and J_j included in $\Lambda_\theta(k\cos\theta)$ is at most two; the algorithm runs in $\mathcal{O}(KN)$ time.

Now, let us cut $\Lambda_\theta(k\cos\theta)$ in two, as in Fig. 4(b). The left part yields segments I_{1i} and J_{1j}, and the right part yields I_{2i} and J_{2j}. When r is 0 (and only then), the force between $\cup I_i$ and $\cup J_j$

can be expressed in terms of the force between $\cup I_{1i}$ and $\cup J_{1j}$ and the force between $\cup I_{2i}$ and $\cup J_{2j}$. The value $\mathrm{F}_0(A,B,k\cos\theta,\theta)$ can therefore be calculated recursively. The approach leads to an algorithm in $\mathcal{O}(KN)$ time for the computation of constant force histograms [20].

When $\Lambda_\theta(k\cos\theta)$ is cut in two, each half-line can be adjusted so that it runs through the center of the leftmost and rightmost pixels (Fig. 4(c)). In that case, the rasterization of each half-line may yield a different set of pixels (notice the darker pixel in Fig. 4(c); compare with Fig. 4(b)) and, therefore, different segments I_{1i}, J_{1j}, I_{2i}, J_{2j}. This slightly different approach leads to an algorithm in $\mathcal{O}(N\log N)$ time [20] which computes constant forces in $8\sqrt{N}-8$ directions.

Fig. 4. Key principles of the existing algorithms for force histogram computation. Pixels marked *"A"* (resp. *"B"*; *"AB"*; *"X"*) belong to *A* and not *B* (resp. *B* and not *A*; *A* and *B*; neither *A* nor *B*). *(a)* Computing the histogram value $\mathrm{F}_r^{AB}(\theta)$ comes down to computing the force between two unions of segments, such as $I_1\cup I_2$ and $J_1\cup J_2$. *(b)* When *r* is 0, this force can be computed recursively. *(c)* A variant.

3. An equivalent definition of the histogram of forces

Consider the function $\psi^{AB} \mid \vec{\mathcal{P}} \rightarrow \Re$ defined by

$$\psi^{AB}(\vec{v}) = \int_{\vec{u}\,\in\vec{\mathcal{P}}} A(\omega+\vec{u})\,B(\omega+\vec{u}+\vec{v})\,\mathrm{d}\vec{u}\,. \tag{7}$$

It is a mathematical correlation that provides raw information about the relative position between the objects *A* and *B*. We call it the *spatial correlation* between *A* and *B*. Using coordinate notation, Eq. (7) becomes:

$$\psi^{AB}(s,t) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} A(x,y)\,B(x+s,y+t)\,\mathrm{d}x\,\mathrm{d}y\,. \tag{8}$$

Now, consider the function $\hat{\mathrm{F}}_r^{AB} \mid [0,2\pi) \rightarrow \Re$ defined by:

$$\hat{\mathrm{F}}_r^{AB}(\theta) = \int_0^{+\infty} [\,\psi^{AB}(k\vec{\theta})\,/\,k^r\,]\,\mathrm{d}k\,. \tag{9}$$

Or, using coordinate notation:

$$\hat{\mathrm{F}}_r^{AB}(\theta) = \int_0^{+\infty} [\psi^{AB}(k\cos\theta, k\sin\theta) / k^r]\,\mathrm{d}k\,. \tag{10}$$

We have $\hat{\mathrm{F}}_r^{AB} = \mathrm{F}_r^{AB}$, where F_r^{AB} is the force histogram defined through Eqs. (1-3). The proof is given in Appendix A.

4. Geometric properties

Many computer vision tasks require the design of robust descriptors: these descriptors should not be sensitive to the position of the camera with respect to the photographed scene; they should not be affected by variations in size, translation and rotation. We examine here the behavior of the spatial correlation (Section 4.1) and of the force histogram (Section 4.2) towards invertible affine transformations. This is an important step towards the design of affine-invariant relative position descriptors [17].

4.1. Properties of ψ^{AB}

Consider two objects *A* and *B* and two translations tra_1 and tra_2. As shown in Appendix B:

$$\psi^{tra_1[A]\,tra_2[B]}(\vec{v}) = \psi^{AB}(\vec{v} + tra_1 - tra_2). \tag{11}$$

In other words, the spatial correlation $\psi^{tra_1[A]\,tra_2[B]}$ between $tra_1[A]$ and $tra_2[B]$ can be easily deduced from the spatial correlation ψ^{AB}. Note that when $tra_1 = tra_2 = tra$, Eq. (11) becomes

$$\psi^{tra[A]\,tra[B]}(\vec{v}) = \psi^{AB}(\vec{v}). \tag{12}$$

Now, consider an invertible linear transformation *lin*. As shown in Appendix C:

$$\psi^{lin[A]\,lin[B]}(\vec{v}) = |\det(lin)|\;\psi^{AB}(lin^{-1}\cdot\vec{v})\,. \tag{13}$$

Finally, consider an invertible affine transformation $aff = tra \circ lin$. We have

$$\psi^{aff[A]\,aff[B]}(\vec{v}) = \psi^{tra[lin[A]]\,tra[lin[B]]}(\vec{v})\,. \tag{14}$$

Equations (12, 14) give

$$\psi^{aff[A]\,aff[B]}(\vec{v}) = \psi^{lin[A]\,lin[B]}(\vec{v}) \tag{15}$$

and Eqs. (13, 15) give

$$\psi^{aff[A]\,aff[B]}(\vec{v}) = |\det(lin)|\; \psi^{AB}(lin^{-1} \cdot \vec{v}) \;. \tag{16}$$

4.2. Properties of $\hat{\mathrm{F}}_r^{AB}$

The behavior of the histogram of forces F_r^{AB} towards the basic affine transformations shown in Fig. 2 was examined in previous work: proofs of **Properties 1-4** below are in [7] (Chapter II, Appendix A) and proof of **Property 5** is in [17] (Appendix C). The behavior of F_r^{AB} towards *any* invertible affine transformation was not, however, determined. This is achieved here, through **Property 6**. Its proof is given in Appendix D. Since $\hat{\mathrm{F}}_r^{AB} = \mathrm{F}_r^{AB}$ (Section 3), all these properties hold, of course, for both F_r^{AB} and $\hat{\mathrm{F}}_r^{AB}$. For any $\theta \in [0,2\pi)$:

Property 1: $$\mathrm{F}_r^{tra[A]\;tra[B]}(\theta) = \mathrm{F}_r^{AB}(\theta) \tag{17}$$

Property 2: $$\mathrm{F}_r^{ref[A]\,ref[B]}(\theta) = \mathrm{F}_r^{AB}(\theta') \tag{18}$$

where θ' belongs to $[0,2\pi)$ and is congruent to $2\alpha-\theta$ modulo 2π.

Property 3: $$\mathrm{F}_r^{rot[A]\,rot[B]}(\theta) = \mathrm{F}_r^{AB}(\theta') \tag{19}$$

where θ' belongs to $[0,2\pi)$ and is congruent to $\theta-\rho$ modulo 2π.

Property 4: $$\mathrm{F}_r^{dil[A]\;dil[B]}(\theta) = \lambda^{3-r}\,\mathrm{F}_r^{AB}(\theta) \tag{20}$$

Property 5: $$\mathrm{F}_r^{str[A]\;str[B]}(\theta) \;=\; k^{2-r}[1+(k^2-1)\cos^2\theta]^{(r-1)/2}\;\mathrm{F}_r^{AB}(\theta') \tag{21}$$

where θ' belongs to $[0,2\pi)$ and is congruent to $\arctan(k^{-1}\tan\theta)$ if $\cos\theta$ is positive; to θ if $\cos\theta$ is zero; to $\arctan(k^{-1}\tan\theta)+\pi$ otherwise.

Property 6: $$\hat{\mathrm{F}}_r^{aff[A]\,aff[B]}(\theta) = |\det(lin)|\;|\,lin^{-1} \cdot \vec{\theta}\,|^{r-1}\;\hat{\mathrm{F}}_r^{AB}(\theta') \tag{22}$$

where θ' is $\angle(lin^{-1} \cdot \vec{\theta})$ and $aff = tra \circ lin$ is any invertible affine transformation.

Note that **Property 6** implies all other properties. As an example, Appendix E shows how **Property 5** can be derived from **Property 6**. The proof is far more elegant than the one in [17].

5. Case of raster data

Here, A and B correspond to regions in a digital image of size $N=m\times n$. In other words: m and n are positive integers; A and B are raster objects; for any point p such that $A(p)\neq 0$ or $B(p)\neq 0$, the x- and y- coordinates of $\lfloor p \rfloor$ belong to $0..m-1$ and $0..n-1$ respectively. As shown in Section 5.2 (and illustrated later in Section 6), the force histogram $\hat{\mathrm{F}}_r^{AB}$ can then be approximated in an efficient and accurate way. First, Section 5.1 examines how to implement ψ^{AB}. Note that some of the elements in Section 5.1 can be found in a preliminary work [16], which introduced the idea of the spatial correlation. In [16], however, ψ^{AB} was defined directly in the discrete space and could not, therefore, be properly linked to the histogram of forces. As a result, the histograms actually considered (and approximated) in [16] are similar—but not identical—to force histograms.

5.1. *Implementing* ψ^{AB}

Consider Eq. (8). Practically, only a finite number of $\psi^{AB}(s,t)$ values can be calculated. It is easy to show that if s and t are integers, we have (since A and B are raster objects):

$$\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} A(x,y)\,B(x+s,y+t)\,\mathrm{d}x\,\mathrm{d}y = \sum_{x=0}^{m-1}\sum_{y=0}^{n-1} A(x,y)\,B(x+s,y+t)\,. \tag{23}$$

This double sum is 0 if $s\notin -m+1..m-1$ or $t\notin -n+1..n-1$ (Fig. 5). In other words,

$$\psi^{AB}(s,t) = \sum_{x=0}^{m-1}\sum_{y=0}^{n-1} A(x,y)\,B(x+s,y+t) \tag{24}$$

and (about) $4N$ pairs (s,t) of integers are of interest. When A and B are crisp, $\psi^{AB}(s,t)$ simply counts the number of pairs (p,q) where p and q are points with integer coordinates such that $A(p)=1$, $B(q)=1$, and $\overrightarrow{pq}=(s,t)$.

Fig. 5. Spatial correlation ψ^{AB} in the case of raster data. *(a)* Two (crisp) objects A and B in an $m\times n$ image. *(b)* The nonzero ψ^{AB} values are all in a $(2m-1)\times(2n-1)$ domain. The brighter the area, the higher the value.

Equation (24) calculates each $\psi^{AB}(s,t)$ in $\mathcal{O}(N)$ time, i.e., it calculates ψ^{AB} in $\mathcal{O}(4N^2)$ time. There is, however, a much faster way. Let A' be the reflection of A about the origin and let B' be the translation of B by the vector $(-m+1,-n+1)$. For any real numbers x and y, we have $A'(x,y) = A(-x,-y)$ and $B'(x,y) = B(x+m-1,y+n-1)$. See Fig. 6. Consider the matrix $[\psi^{AB}]$ with $2m-1$ columns and $2n-1$ rows defined as follows: for any i in $1..2m-1$ and any j in $1..2n-1$, the element $[\psi^{AB}]_{ij}$ at the i-th column (counting from left to right) and j-th row (from bottom to top) is $\psi^{AB}(i-m,j-n)$. The matrices $[A']$ and $[B']$ are defined in the same way. Now, let us make the change of variables $x'=-x$ and $y'=-y$ in Eq. (24). We get:

$$\psi^{AB}(s,t)=\sum\nolimits_{x'=-m+1}^{0}\sum\nolimits_{y'=-n+1}^{0}A'(x',y')\,B(s-x',t-y')\,. \tag{25}$$

Equation (25) corresponds to a 2-D discrete convolution. Using matrix notation:

$$[\psi^{AB}]=[A']\otimes[B']\,, \tag{26}$$

where $\otimes$ denotes the convolution operator. As we all know, the Fourier transform has the ability to convert a convolution into an ordinary product, and vice versa. According to the convolution theorem, Eq. (26) can be transformed into

$$[\psi^{AB}] = W_{2n-1}^{-1}\cdot\mathbf{[}(W_{2n-1}\cdot[A']\cdot W_{2m-1})\times(W_{2n-1}\cdot[B']\cdot W_{2m-1})\mathbf{]}\cdot W_{2m-1}^{-1}, \tag{27}$$

where W_k and W_k^{-1} are the Fourier and inverse Fourier transform matrices of order k, and where × is the array product (given two matrices M_1 and M_2 of the same size, $[M_1 \times M_2]_{ij} = [M_1]_{ij}\,[M_2]_{ij}$ for any i and j). According to Eq. (27), the computation of $[\psi^{AB}]$ requires three 2-D discrete Fourier transforms. ψ^{AB} can therefore be calculated in $\mathcal{O}(N\log N)$ time—since this is the complexity of Fast Fourier Transform algorithms.

Fig. 6. Computation of ψ^{AB}. *(a)* A' is the reflection of A about the origin. B' is the translation of B by the vector $(-m+1,-n+1)$. *(b)* The matrix $[A']$ defined from A'. *(c)* The matrix $[B']$ defined from B'. *(d)* The matrix $[\psi^{AB}]$, which defines ψ^{AB}, can be deduced from $[A']$ and $[B']$.

The behavior of ψ^{AB} towards translations leads to further optimization. Consider a vector $\vec{v}$ and a translation *tra*. If $\vec{v}$ and the translation vector have integer coordinates then

$$\psi^{tra[A]B}(\vec{v}) = \psi^{AB}(\vec{v} + tra)\,. \tag{28}$$

Equation (28) comes from Eq. (11): replace tra_1 with *tra* and tra_2 with the translation whose vector is the null vector. Let us permute the two members of the equality, and let us make the change of variable $\vec{u} = \vec{v} + tra$. We get:

$$\psi^{AB}(\vec{u}) = \psi^{tra[A]B}(\vec{u} - tra). \tag{29}$$

Now, let us choose *tra* so that the minimum bounding rectangle of *tra*[A] and B is as small as possible. This rectangle defines a host image of size $N' = m' \times n'$. See Fig. 7. Since $\psi^{tra[A]B}$ can be computed in $\mathcal{O}(N'\log N')$ time using the FFT, and since ψ^{AB} can be derived from $\psi^{tra[A]B}$ in $\mathcal{O}(N)$ time according to Eq. (29), ψ^{AB} can be computed in $\mathcal{O}(N'\log N' + N)$ time—which can be noticeably less than $\mathcal{O}(N\log N)$ if N' is small compared to N.

Fig. 7. Optimization of the algorithm for the computation of ψ^{AB}. *(a)* A and B are objects in an $m \times n$ image. $A' = tra[A]$ and B are objects in a smaller $m' \times n'$ image. *(b)* ψ^{AB} can be deduced from $\psi^{A'B}$.

5.2. Implementing $\hat{\mathrm{F}}_r^{AB}$

Here again, only a finite number of $\hat{\mathrm{F}}_r^{AB}(\theta)$ values can be calculated. Only a finite number K of directions θ are therefore considered. A reasonable option, as mentioned in Section 2.3, is to pick a set of evenly distributed reference directions such as $\{2\pi i/K\}_{i \in 0..K-1}$. Assume $\theta \in [0,\pi/4) \cup [7\pi/4, 2\pi)$. Let $s = k\cos\theta$, i.e., $k = s/\cos\theta$. We then have $1/k^r = (\cos\theta)^r/s^r$ and $\mathrm{d}k = \mathrm{d}s/\cos\theta$. Equation (10) becomes:

$$\hat{\mathrm{F}}_r^{AB}(\theta) = (\cos\theta)^{r-1} \int_0^{+\infty} [\psi^{AB}(s, s\tan\theta) / s^r] \mathrm{d}s \,. \tag{30}$$

$\hat{\mathrm{F}}_r^{AB}(\theta)$ is approximated as follows:

$$\hat{\mathrm{F}}_r^{AB}(\theta) \approx (\cos\theta)^{r-1} \sum_{s=0}^{m-1} \psi^{AB}(s, \lfloor \mathrm{s} \tan\theta \rfloor) \, f_r(s) \,, \tag{31}$$

where $f_r(s)$ is chosen based upon the interpretation of $1/s^r$ in the context of Eq. (30). Consider two points p and q in $\mathcal{P}$ such that $\overrightarrow{pq}$ is the vector of coordinates $(s, s\tan\theta)$. Let p' and q' be the projections of p and q on the X-axis. In Eq. (30), the value s represents the distance between p' and q', and $1/s^r$ measures the magnitude of the elementary force exerted by q' on p'. This force tends to pull p' in direction 0. See Fig. 8(a). In Eq. (31), however, p and q are such that $\overrightarrow{pq}$ is the vector of coordinates $(s, \lfloor s\tan\theta \rfloor)$. The point p is actually the representative of a whole set of points, i.e., the pixel it belongs to. The same applies to q. These pixels project on the X-axis as unit line segments, I and J. See Fig. 8(b). It is therefore natural to define $f_r(s)$ as the sum of the elementary forces exerted in direction 0 by all points of J on those of I, i.e.,

$$f_r(s) = \int_0^1 \int_0^1 \varphi_r(x - y + s)\,\mathrm{d}x\,\mathrm{d}y, \tag{32}$$

where φ_r is defined as in Eq. (3). This double integral can be calculated analytically. Its algebraic expression depends on s and r, as shown in Table 1.

Fig. 8. About the function f_r. Case $\theta\in[0,\pi/4)\cup[7\pi/4,2\pi)$. In *(a)*, s is a nonnegative real number and $\overrightarrow{pq}$ is the vector of coordinates $(s, s\tan\theta)$. The value $1/s^r$ is the magnitude of the force exerted by q' on p'. In *(b)*, s is a nonnegative integer and $\overrightarrow{pq}$ is the vector of coordinates $(s, \lfloor s\tan\theta\rfloor)$. The value $f_r(s)$ is the magnitude of the force exerted by J on I.

Table 1. Algebraic expressions for the double integral in Eq. (32), where ln denotes the Neperian logarithm.

Equation (31) only holds when $\theta\in[0,\pi/4)\cup[7\pi/4,2\pi)$. Other values of θ yield other equations. When $\theta\in[3\pi/4,5\pi/4)$,

$$\hat{\mathrm{F}}_r^{AB}(\theta) \approx (-\cos\theta)^{r-1} \sum_{s=0}^{m-1} \psi^{AB}(-s, -\lfloor \mathrm{s}\tan\theta\rfloor)\, f_r(s)\,; \tag{33}$$

when $\theta\in[\pi/4,3\pi/4)$,

$$\hat{\mathrm{F}}_r^{AB}(\theta) \approx (\sin\theta)^{r-1} \sum_{s=0}^{m-1} \psi^{AB}(\lfloor \mathrm{s}\cot\theta\rfloor, s)\, f_r(s)\,; \tag{34}$$

and when $\theta\in[5\pi/4,7\pi/4)$,

$$\hat{\mathrm{F}}_r^{AB}(\theta) \approx (-\sin\theta)^{r-1} \sum_{s=0}^{m-1} \psi^{AB}(-\lfloor \mathrm{s}\cot\theta\rfloor, -s)\, f_r(s)\,. \tag{35}$$

Equations (31, 33-35) represent a new way of computing force histograms in the case of 2-D raster data. The histograms are computed in $\mathcal{O}(N\log N + K\sqrt{N})$ time: the equations above require $\mathcal{O}(K\sqrt{N})$ by themselves, while the spatial correlation ψ^{AB} requires $\mathcal{O}(N\log N)$ (see Section 5.1).

Let us take a closer look at K. Assume the image is of size $N=n\times n$. Consider two raster lines as depicted in Fig. 9(ab). Starting from the boundary pixel p_1 (in the leftmost column), they run in slightly different directions θ_1 and θ_2, with $\theta_1 \in [0,\pi/4]$ and $\theta_2 \in [0,\pi/4]$. The two lines correspond to different sets of pixels, but both join p_1 and p_2 (in the rightmost column). We hypothesize that the difference between the computed values for $\hat{\mathrm{F}}_r^{AB}(\theta_1)$ and $\hat{\mathrm{F}}_r^{AB}(\theta_2)$ is negligible, below significance considering the image resolution. In other words, no more than n directions are worth considering in $[0,\pi/4]$ (see Fig. 9(c)), and no more than $8n-8$ directions in $[0,2\pi)$. This shows that there is no interest in choosing for K a value greater than $8\sqrt{N}-8$.

In conclusion, the algorithm presented in this section actually runs in $\mathcal{O}(N\log N + \sqrt{N}\sqrt{N})$, i.e., in $\mathcal{O}(N\log N)$. Computational times are basically independent of r (the type of force histograms), K (the number of directions in which forces are computed), and M (the number of possible membership degrees). All objects—whether they are convex or concave, of simple or complex shapes, defined as crisp or fuzzy sets—are handled in an equal manner. The experiments in Section 6 validate this analysis.

Fig. 9. Directions θ in an $n\times n$ image. The gray pixels represent *(a)* the raster line in direction θ_1 originating from p_1, and *(b)* the raster line in direction θ_2 originating from p_1. Since both lines join p_1 to p_2, there is no reason to distinguish θ_2 from θ_1. Therefore, *(c)*, no more than n directions are worth considering in the range $[0,\pi/4]$.

6. Comparative experimental study

In this section, $\hat{\mathrm{F}}_r^{AB}$ denotes a force histogram computed using the new algorithm (Section 5); F_r^{AB} denotes a histogram computed using the traditional algorithm (briefly described in Section 2.3); $\bar{\mathrm{F}}^{AB}$ and $\tilde{\mathrm{F}}^{AB}$ are constant force histograms computed in $\mathcal{O}(KN)$ and $\mathcal{O}(N\log N)$ time respectively (see Section 2.3 as well). For a given K, the $\theta_i=2\pi i/K$ values with i in $0..K-1$ are the directions in which forces are considered.

The four algorithms are dedicated to the handling of 2-D data in raster form. All are approximation algorithms. For example, the traditional algorithm is based on the approximations expressed by Eqs. (5, 6), while the new algorithm is based on the approximations expressed by Eqs. (31, 33-35). We can expect, therefore, to have $\hat{\mathrm{F}}_r^{AB} \approx \mathrm{F}_r^{AB}$ instead of $\hat{\mathrm{F}}_r^{AB} = \mathrm{F}_r^{AB}$. We can also expect $\widetilde{\mathrm{F}}^{AB} \approx \mathrm{F}_0^{AB}$. The exception is $\overline{\mathrm{F}}^{AB} = \mathrm{F}_0^{AB}$, since the two algorithms only differ in that one uses recursion and the other does not (Section 2.3). In the case of vector data, however, there exists an *exact* algorithm: its time complexity (for crisp objects) is $\mathcal{O}(K\eta\log\eta)$, where η is the total number of object vertices [7][10][15]. It is therefore possible to evaluate the accuracy of each histogram $\hat{\mathrm{F}}_r^{AB}$, F_r^{AB}, $\overline{\mathrm{F}}^{AB}$ and $\widetilde{\mathrm{F}}^{AB}$. This is achieved in Section 6.2. Processing times are compared in Section 6.3. The test data are presented first, in Section 6.1. Note that the memory complexity of the four approximation algorithms is $\mathcal{O}(N)$. The algorithms were implemented in C and run on a machine equipped with Intel Pentium D CPU 3.0 GHz, 1GB memory, and Windows Vista.

6.1. Test data

The test data include 4 pairs of crisp objects. Each pair consists of two identical objects: the quadratic Koch island fractal at iteration 0 (Fig. 10(a)), 1 (Fig. 10(b)) and 3 (Fig. 10(cd)). Any line that runs through *A* and *B* in Fig. 10(a) intersects them in one segment each. As explained in Section 2.3, this is the best-case scenario for F_r^{AB} in terms of processing time. Even more so that many directions θ_i satisfy $\mathrm{F}_r^{AB}(\theta_i)$=0; these directions can be predetermined in order to speed up computation. On the other hand, a line that runs through *A* and *B* in Fig. 10(c) intersects them in a large number of segments. This is due to the fractal structures of the objects (and would be less apparent with, say, the more famous Koch snowflake). Processing times for F_r^{AB} can be expected to be much higher. The worst-case scenario is represented by Fig. 10(d): since *A* and *B* overlap, lines that run through *A* and *B* can be found in every direction θ_i (i.e., $F_r^{AB}(\theta_i)$ is never 0). In the end, the variety offered by the four pairs of objects allows us to fairly compare the computational efficiency of the different algorithms.

Each pair of objects in Fig. 10 was also used to generate several pairs of fuzzy objects. For a given *M*, the pixels in *A* were assigned different membership degrees, randomly selected from $\{i/M\}_{i\in 1..M}$. The object *B* was fuzzified in the same way.

Fig. 10. Test data. The crisp objects used in the experiments are quadratic Koch islands generated after *(a)* 0, *(b)* 1, and *(c)(d)* 3 iterations. In *(d)*, the two objects overlap (black area).

6.2. Results on accuracy

The handling of fuzzy objects basically comes down to the handling of their α-cuts, which are crisp objects [7][15]. The accuracy of the force histogram approximations does not depend, therefore, on whether the objects are crisp or fuzzy. This was confirmed experimentally (results not shown here). In this section, we use the (crisp) squares of Fig. 10(a) to illustrate the fact that the new algorithm is more accurate than its competitors. The two squares, *A* and *B*, can be easily encoded in vector form. As mentioned in the introductory paragraphs of Section 6, this makes it possible to calculate the exact force histogram—which can then be compared with $\hat{F}_r^{AB}$, F_r^{AB}, $\bar{F}^{AB}$ and $\tilde{F}^{AB}$. The dissimilarity between one of these four histograms, *h*, and the exact histogram, h_e, is measured by the difference ratio

$$DR(h)=\left(\frac{\sum_{i=0}^{K-1}\left|h(\theta_i)-h_e(\theta_i)\right|}{\sum_{i=0}^{K-1}\max\{h(\theta_i),h_e(\theta_i)\}}\right)\times 100\% \tag{36}$$

$DR(h)$=0% iff h=h_e, and $DR(h)$=100% iff h and h_e are orthogonal in the vector space $\Re^K$. The lower $DR(h)$, the higher the accuracy of the approximation h. Note that DR is defined based on the grid norm (1-norm). Other norms, like the Euclidean norm (2-norm), could be used. Besides, there are other bin-by-bin or cross-bin measurements suitable for the task as well [21].

Figure 11 summarizes the most interesting results of our experiments. $\bar{\mathrm{F}}^{AB}$ does not appear in this figure for the simple reason that $\bar{\mathrm{F}}^{AB}=\mathrm{F}_0^{AB}$ (as explained in the introductory paragraphs of Section 6) and hence $DR(\bar{\mathrm{F}}^{AB})$=$DR(\mathrm{F}_0^{AB})$. The parameter K does not appear either because it has no noticeable impact on the measurement of accuracy; all the diagrams look the same whether K is a three- or a four-digit number. The accuracy of all histograms increases with N. It drops, however, as r increases past 2. The most accurate approximation is $\hat{\mathrm{F}}_r^{AB}$: we have $DR(\hat{\mathrm{F}}_r^{AB})<0.1\%$ when $r\leq 2$ and $N\geq 256^2$. The accuracy of F_r^{AB} is not as high, but comparable: in our experiments, we have consistently found that $DR(\hat{\mathrm{F}}_r^{AB})<DR(\mathrm{F}_r^{AB})<3\times DR(\hat{\mathrm{F}}_r^{AB})$. The least accurate approximation is $\tilde{\mathrm{F}}^{AB}$, since $18\times DR(\hat{\mathrm{F}}_r^{AB})<DR(\tilde{\mathrm{F}}^{AB})<128\times DR(\hat{\mathrm{F}}_r^{AB})$.

Fig. 11. Histogram accuracy. *A* and *B* are the objects from Fig. 10(a). In *(c)*, N=256^2.

6.3. Results on efficiency

Experiments (not shown here) confirm that r has no impact on the processing time of $\hat{\mathrm{F}}_r^{AB}$. This is not exactly the case for F_r^{AB}, as F_1^{AB} and F_2^{AB} are processed noticeably faster (Fig. 12). The main reason is that the computation of each $\mathrm{F}_r^{AB}(\theta_i)$ translates into the assessment of a number of algebraic expressions (Section 2.3) which are computationally simpler when r=1 or r=2. These two values naturally appear when determining the expressions through integral calculus (as they appear in Table 1). See [7] for details. In the following, only constant and gravitational force histograms are considered: r=0 represents the worst-case scenario for F_r^{AB}, while r=2 represents the best-case scenario.

Fig. 12. Processing time (T) of F_r^{AB} with respect to the type of forces (r).
The objects A and B are from Fig. 10(c), N=256^2, K=360, and the increment for r is 0.5.

K has almost no influence on the computational efficiencies of $\hat{\mathrm{F}}_r^{AB}$ and $\tilde{\mathrm{F}}^{AB}$, and $\hat{\mathrm{F}}_0^{AB}$ computes much faster than $\tilde{\mathrm{F}}^{AB}$ (Figs. 13(b)-14(b)). On the other hand, the processing times of F_r^{AB} and $\bar{\mathrm{F}}^{AB}$ depend linearly on K. When K is set to $8\sqrt{N}-8$ (the maximum number of directions that are worth considering, as shown in Section 5.2), $\hat{\mathrm{F}}_r^{AB}$ outperforms F_r^{AB} and $\bar{\mathrm{F}}^{AB}$ (Fig. 13). When K is small enough, however, F_r^{AB} and $\bar{\mathrm{F}}^{AB}$ compute faster than $\hat{\mathrm{F}}_r^{AB}$ (Fig. 14).

The shape of the objects A and B has no influence on the computational efficiencies of $\hat{\mathrm{F}}_r^{AB}$, $\tilde{\mathrm{F}}^{AB}$ and $\bar{\mathrm{F}}^{AB}$, but it does have (as mentioned in Section 2.3) a significant influence on F_r^{AB} (see the gray areas in Figs. 13(ac)-14(ac)). The phenomenon, however, is less pronounced for F_2^{AB} than for F_0^{AB}, for the same reason that F_2^{AB} computes noticeably faster than other F_r^{AB} histograms (as explained above).

Similarly, the relative position of the objects has no influence on the computational efficiencies of $\hat{\mathrm{F}}_r^{AB}$ and $\tilde{\mathrm{F}}^{AB}$, but it does have an influence on F_r^{AB}. In particular, F_0^{AB} displays a huge efficiency decrement when A and B overlap (upper boundary of the gray areas in Figs. 13(a)-14(a)). The efficiency of $\bar{\mathrm{F}}^{AB}$ decreases as well (Figs. 13(b)-14(b)), for the same reason.

Fig. 13. Processing times (T) of $\hat{\mathrm{F}}_r^{AB}$, F_r^{AB}, $\bar{\mathrm{F}}^{AB}$ and $\tilde{\mathrm{F}}^{AB}$ with respect to the image size (N), when K=8$\sqrt{N}$–8. In each diagram, the objects A and B are from Fig. 10(a-d). The processing times of F_r^{AB} and $\bar{\mathrm{F}}^{AB}$ depend on the shape and/or the relative position of the objects—hence the gray areas. The lower boundary of each gray area corresponds to the best-case scenario, i.e., A and B are from Fig.10(a). The upper boundary corresponds to the worst-case scenario, i.e., A and B are from Fig. 10(d).

Fig. 14. Processing times (T) of $\hat{\mathrm{F}}_r^{AB}$, F_r^{AB}, $\bar{\mathrm{F}}^{AB}$ and $\tilde{\mathrm{F}}^{AB}$ with respect to the image size (N), when K=360.

Experiments confirm that the number M of membership degrees has no influence on the computational efficiencies of $\hat{\mathrm{F}}_r^{AB}$ (Fig. 15), $\bar{\mathrm{F}}^{AB}$ and $\tilde{\mathrm{F}}^{AB}$ (results not shown here). On the other hand (Fig. 15 and Section 2.2), the complexity of F_r^{AB} depends on M either quadratically (when using the double sum scheme) or linearly (when using the single sum scheme). Consequently, $\hat{\mathrm{F}}_r^{AB}$ outperforms F_r^{AB} even for small values of M.

Table 2 summarizes the dependence of the processing times on the various parameters.

Fig. 15. Processing times (T) of $\hat{\mathrm{F}}_r^{AB}$ and F_r^{AB} with respect to the number of possible membership degrees (M). In each diagram, the objects A and B are from Fig. 10(c), N=256^2, and K=360. The symbols Σ and ΣΣ indicate the simple and double sum schemes (see Section 2.2).

Table 2. The processing time of a histogram like F_r^{AB}, $\bar{\mathrm{F}}^{AB}$, $\tilde{\mathrm{F}}^{AB}$ or $\hat{\mathrm{F}}_r^{AB}$ may (YES) or may not (NO) depend significantly on N, K, M, r, and on the *shape* and relative *position* of the objects A and B. Dependence can be, e.g., linear (K) or quadratic (M^2).

To conclude this section, let us highlight an important feature of the new algorithm. Consider Fig. 14(a). The curve for $\hat{\mathrm{F}}_0^{AB}$ would look exactly the same if K was, say, 1440 instead of 360 (on the other hand, the curves for F_0^{AB} would go 4 times higher). This is because most of the processing time is actually spent on ψ^{AB}. As a result, the following tasks are performed in practically the same amount of time: computing $\hat{F}_r^{AB}$ with K=360 values; computing $\hat{F}_r^{AB}$ with K=1440; computing $\hat{F}_{r_1}^{AB}$, $\hat{F}_{r_2}^{AB}$, $\hat{F}_{r_3}^{AB}$ and $\hat{F}_{r_4}^{AB}$ with 360 values each; computing $\hat{F}_r^{tra_1[A]\,tra_2[B]}$, $\hat{F}_r^{tra_3[A]\,tra_4[B]}$, $\hat{F}_r^{tra_5[A]\,tra_6[B]}$ and $\hat{F}_r^{tra_7[A]\,tra_8[B]}$ with 360 values each (since $\psi^{tra_i[A]\,tra_j[B]}$ can be easily derived from ψ^{AB} using Eq. (11)). In other words, the new algorithm is particularly suitable not only when forces must be considered in a large number of directions, but also when different force histograms must be calculated for the same two objects (as in [9]), or when the force histogram of two moving objects must be calculated at different times.

7. Conclusion

The traditional algorithm for force histogram computation in the case of 2-D raster data runs in $\mathcal{O}(KM^2N\sqrt{N})$ time, where K is the number of directions in which forces are considered, M is the number of possible membership degrees, and N is the number of pixels in the image. An alternative algorithm runs in $\mathcal{O}(KMN\sqrt{N})$ time. Moreover, two algorithms dedicated to constant force histogram computation can be found in the literature: one is in $\mathcal{O}(KN)$, and the other in $\mathcal{O}(N\log N)$. In this paper, we have described an algorithm that runs in $\mathcal{O}(N\log N)$, regardless of the type of force. A comparative study has shown the following. For the computation of a histogram of constant forces when only a relatively small number K of histogram values are needed: the dedicated algorithm in $\mathcal{O}(KN)$ is the most efficient if the objects are fuzzy (M>1), or crisp (M=1) with complex shapes; the traditional algorithm has the best performance if the objects are crisp with simple shapes. For the computation of a histogram of non-constant forces when only a relatively small number of histogram values are needed: the traditional algorithm is the most efficient if the objects are crisp with simple shapes. In all other cases, the new algorithm outperforms its competitors. It is also particularly suitable when different force histograms must be calculated for the same two objects, or when the force histogram of two moving objects must be calculated at different times. The algorithm derives from an equivalent definition of the force histogram that involves a mathematical correlation over the 2-D vector space. This correlation provides raw information about the relative position between the objects and can be efficiently computed using the Fast Fourier Transform. We have shown, moreover, that the equivalent definition is better adapted to the solving of theoretical issues. We have used it here to determine the behavior of the force histogram towards any invertible affine transformation. In the future, it should help us solve other issues, such as the inverse problem [6], object localization [22] and histogram composition [23]. We will also show that it can be extended to 3-D raster data, like the standard definition [24], and we will examine the case of vector data.

Acknowledgements

The authors want to express their gratitude for support from the Natural Science and Engineering Research Council of Canada (NSERC), grant 262117. They also thank the anonymous reviewers for their constructive comments.

References

[1] H.J. Miller, E.A. Wentz, Representation and spatial analysis in geographic information systems, Annals of the Association of American Geographers. 93(3) (2003) 574-594.

[2] J.D.R. Millan, Robot navigation, in: M.A. Arbib (Ed.), Handbook of Brain Theory and Neural Networks, Second Ed., Cambridge, MA: MIT Press, 2002.

[3] A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta, R. Jain, Content-based image retrieval at the end of the early years, IEEE Transactions on Pattern Analysis and Machine Intelligence. 22(12) (2000) 1349-1380.

[4] G. Chronis, M. Skubic, Robot navigation using qualitative landmark states from sketched route maps, Proceedings of IEEE International Conference on Robotics and Automation. 2 (2004) 1530-1535.

[5] R. Krishnapuram, J.M. Keller, Y. Ma, Quantitative analysis of properties and spatial relations of fuzzy image regions, IEEE Transactions on Fuzzy Systems. 1(3) (1993) 222-233.

[6] K. Miyajima, A. Ralescu, Spatial organization in 2-D segmented images: representation and recognition of primitive spatial relations, Fuzzy Sets and Systems. 65(2-3) (1994) 225-236.

[7] P. Matsakis, Relations spatiales structurelles et interprétation d'images, PhD Thesis, Institut de Recherche en Informatique de Toulouse, France, 1998.

[8] P. Matsakis, D. Nikitenko, Combined extraction of directional and topological relationship information from 2-D concave objects, in: M. Cobb, F. Petry, V. Robinson (Eds.), Fuzzy Modeling with Spatial Information for Geographic Problems, Springer-Verlag Publications, 2005, pp. 15-40.

[9] P. Matsakis, J. Keller, L. Wendling, J. Marjamaa, O. Sjahputera, Linguistic description of relative positions in images, IEEE Transactions on Systems, Man and Cybernetics. 31(4) (2001) 573-588.

[10] M. Skubic, P. Matsakis, G. Chronis, J. Keller, Generating multi-level linguistic spatial descriptions from range sensor readings using the histogram of forces, Autonomous Robots. 14(1) (2003) 51-69.

[11] L. Wendling, S. Tabbone, P. Matsakis, Fast and robust recognition of orbit and sinus drawings using histograms of forces, Pattern Recognition Letters. 23(14) (2002) 1687-1693.

[12] S. Tabbone, L. Wendling, Color and grey level object retrieval using a 3D representation of force histogram, Image and Vision Computing. 21(6) (2003) 483-495.

[13] P. Matsakis, Understanding the spatial organization of image regions by means of force histograms: a guided tour, in: P. Matsakis, L. Sztandera (Eds.), Applying Soft Computing in Defining Spatial Relations, Springer-Verlag Publications, 2002, pp. 99-122.

[14] J. Marjamaa, O. Sjahputera, J. Keller, P. Matsakis, Fuzzy scene matching in LADAR imagery, IEEE International Conference on Fuzzy Systems. 2 (2001) 692-695.

[15] P. Matsakis, L. Wendling, A new way to represent the relative position of areal objects, IEEE Transactions on Pattern Analysis and Machine Intelligence. 21(7) (1999) 634-643.

[16] J. Ni, P. Matsakis, Force histograms computed in O(NlogN), Proceedings of the 19th International Conference on Pattern Recognition. (2008) 1-4.

[17] P. Matsakis, J. Keller, O. Sjahputera, J. Marjamaa, The use of force histograms for affine-invariant relative position description, IEEE Transactions on Pattern Analysis and Machine Intelligence. 26(1) (2004) 1-18.

[18] D. Dubois, M.C. Jaulent, A general approach to parameter evaluation in fuzzy digital pictures, Pattern Recognition Letters. 6(4) (1987) 251-259.

[19] J.E. Bresenham, Algorithm for computer control of a digital plotter, IBM Systems Journal. 4(1) (1965) 25-30.

[20] Y. Wang, F. Makedon, R.L. Drysdale, Fast algorithms to compute the force histogram, unpublished (2004).

[21] J. Puzicha, J.M. Buhmann, Y. Rubner, C. Tomasi, Empirical evaluation of dissimilarity measures for color and texture, Proceedings of the 7th IEEE International Conference on Computer Vision. 2 (1999) 1165-1172.

[22] P. Matsakis, J. Ni, X. Wang, Object localization based on directional information: case of 2-D raster data, Proceedings of the 18th International Conference on Pattern Recognition. 2 (2006) 142-146.

[23] J. Malki, L. Mascarilla, E.H. Zahzah, P. Boursier, Directional relations composition by orientation histogram fusion, Proceedings of the 15th International Conference on Pattern Recognition. 3 (2000) 758-761.

[24] J. Ni, P. Matsakis, L. Wawrzyniak, Quantitative representation of the relative position between 3D Objects, Proceedings of the 4th IASTED International Conference on Visualization, Imaging, and Image Processing. (2004) 452-289.

Appendix A. Proof of $\hat{\mathbf{F}}_r^{AB} = \mathbf{F}_r^{AB}$

Equations (3, 9) give:

$$\hat{\mathrm{F}}_r^{AB}(\theta) = \int_{-\infty}^{+\infty} [\; \psi^{AB}(k\vec{\theta})\; \varphi_r(k)\;]\; \mathrm{d}k$$

Combine it with Eq. (7):

$$\hat{\mathrm{F}}_r^{AB}(\theta) = \int_{-\infty}^{+\infty} \Big[\int_{\vec{u}\,\in\,\vec{\mathcal{P}}} \; A(\omega + \vec{u}\,)\; B(\omega + \vec{u} + k\vec{\theta}\,)\; \mathrm{d}\vec{u}\; \Big]\; \varphi_r(k)\; \mathrm{d}k$$

$(\vec{\theta}, \vec{\theta}^{\perp})$ is an orthonormal basis of $\vec{\mathcal{P}}$. Therefore, for each vector $\vec{u}$, there exist values $s \in \Re$ and $t \in \Re$ such that $\vec{u} = t\vec{\theta}^{\perp} + s\vec{\theta}$. The above equation becomes:

$$\hat{\mathrm{F}}_r^{AB}(\theta)=\int_{-\infty}^{+\infty}\Big(\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} A\,(\omega+t\vec{\theta}^{\,\perp}+s\vec{\theta}\,)\ \ B\,(\omega+t\vec{\theta}^{\,\perp}+(s+k)\vec{\theta}\,)\ \ \varphi_r(k)\ \ \mathrm{d}k\ \ \mathrm{d}s\Big)\,\mathrm{d}t$$

The points $p=\omega+t\vec{\theta}^{\perp}+s\vec{\theta}$ and $q=\omega+t\vec{\theta}^{\perp}+(s+k)\vec{\theta}$ belong to $\Lambda_\theta(t)$.

Since $\overrightarrow{pq}=k\vec{\theta}$ we have $\overrightarrow{pq}\bullet\vec{\theta}=k$. Therefore—and according to Eqs. (2, 1):

$$\hat{\mathrm{F}}_r^{AB}(\theta)=\int_{-\infty}^{+\infty}\Big[\int_{p\in\Lambda_\theta(t)}\int_{q\in\Lambda_\theta(t)}A(p)\,B(q)\,\varphi_r(\,\overrightarrow{pq}\bullet\vec{\theta}\,)\,\mathrm{d}p\ \mathrm{d}q\Big]\,\mathrm{d}t$$

$$=\int_{-\infty}^{+\infty}\mathrm{F}_r(A,B,t,\theta)\,\mathrm{d}t\ =\ \mathrm{F}_r^{AB}(\theta)$$

Appendix B. Proof of Eq. (11)

From Eq. (7), we have:

$$\psi^{tra_1[A]\,tra_2[B]}(\,\vec{v}\,)=\int_{\vec{u}\,\in\vec{\mathcal{P}}} tra_1[A](\omega+\vec{u}\,)\ tra_2[B](\omega+\vec{u}+\vec{v}\,)\,\mathrm{d}\vec{u}$$

Since $tra_1[A](p)=A(p+(-tra_1))$ and $tra_2[B](p)=B(p+(-tra_2))$:

$$\psi^{tra_1[A]\,tra_2[B]}(\,\vec{v}\,)=\int_{\vec{u}\,\in\vec{\mathcal{P}}} A(\omega+\vec{u}-tra_1)\ B(\omega+\vec{u}+\vec{v}-tra_2)\,\mathrm{d}\vec{u}$$

$$=\int_{\vec{u}\,\in\vec{\mathcal{P}}} A(\omega+\vec{u}-tra_1)\ B(\omega+\vec{u}-tra_1+\vec{v}+tra_1-tra_2)\,\mathrm{d}\vec{u}$$

Let $\vec{u}\,'=\vec{u}-tra_1$. We have $\mathrm{d}\vec{u}\,'=\mathrm{d}\vec{u}$ and—according to Eq. (7):

$$\psi^{tra_1[A]\,tra_2[B]}(\,\vec{v}\,)=\int_{\vec{u}\,'\,\in\vec{\mathcal{P}}} A(\omega+\vec{u}\,')\ B(\omega+\vec{u}\,'+(\,\vec{v}+tra_1-tra_2))\,\mathrm{d}\vec{u}\,'\ =\ \psi^{AB}(\,\vec{v}+tra_1-tra_2)$$

Appendix C. Proof of Eq. (13)

From Eq. (7): $\psi^{lin[A]\,lin[B]}(\,\vec{v}\,)=\int_{\vec{u}\,\in\vec{\mathcal{P}}} lin[A](\omega+\vec{u}\,)\ lin[B](\omega+\vec{u}+\vec{v}\,)\,\mathrm{d}\vec{u}$

However: $lin[A](\omega+\vec{u}\,)=A(lin^{-1}(\omega+\vec{u}\,))=A(\omega+lin^{-1}\cdot\vec{u}\,)$

Similarly: $lin[B](\omega+\vec{u}+\vec{v}\,)=B(\omega+lin^{-1}\cdot(\,\vec{u}+\vec{v}\,))=B(\omega+lin^{-1}\cdot\vec{u}+lin^{-1}\cdot\vec{v}\,)$

Therefore: $\psi^{lin[A]\,lin[B]}(\,\vec{v}\,)=\int_{\vec{u}\,\in\vec{\mathcal{P}}} A(\omega+lin^{-1}\cdot\vec{u}\,)\ B(\omega+lin^{-1}\cdot\vec{u}+lin^{-1}\cdot\vec{v}\,)\,\mathrm{d}\vec{u}$

Let $\vec{u}\,'=lin^{-1}\cdot\vec{u}$. We then have $\vec{u}=lin\cdot\vec{u}\,'$ and, according to linear algebra,

$\mathrm{d}\vec{u} = \mathrm{d}(lin \cdot \vec{u}\,') = |\det(lin)|\; \mathrm{d}\vec{u}\,'$. The above equation becomes:

$$\psi^{\,lin[A]\;lin[B]}(\,\vec{v}\,) = |\det(lin)| \int_{\vec{u}\,' \in \vec{\mathcal{P}}} A(\omega + \vec{u}\,')\, B(\omega + \vec{u}\,' + lin^{-1} \cdot \vec{v}\,)\; \mathrm{d}\vec{u}\,'$$

which, according to Eq. (7), can be rewritten:

$$\psi^{\,lin[A]\;lin[B]}(\,\vec{v}\,) = |\det(lin)|\; \psi^{AB}(lin^{-1} \cdot \vec{v}\,)$$

Appendix D. Proof of Eq. (22)

According to Eq. (9): $\hat{\mathrm{F}}_r^{aff[A]\,aff[B]}(\theta) = \int_0^{+\infty} [\, \psi^{aff[A]\,aff[B]}(k\vec{\theta}\,) / k^r \,]\; \mathrm{d}k$

Combine it with Eq. (15): $\hat{\mathrm{F}}_r^{aff[A]\,aff[B]}(\theta) = \int_0^{+\infty} [\, \psi^{lin[A]\,lin[B]}(k\vec{\theta}\,) / k^r \,]\; \mathrm{d}k$

and now with Eq. (13): $\hat{\mathrm{F}}_r^{aff[A]\,aff[B]}(\theta) \;=\; |\det(lin)| \int_0^{+\infty} [\, \psi^{AB}(k\,(lin^{-1} \cdot \vec{\theta}\,)) / k^r \,]\; \mathrm{d}k$

Let $\theta' = \angle(lin^{-1} \cdot \vec{\theta}\,)$, i.e., $\vec{\theta}\,' = (lin^{-1} \cdot \vec{\theta}\,) \,/\, |\, lin^{-1} \cdot \vec{\theta}\,|$. The above equation can be rewritten as:

$$\hat{\mathrm{F}}_r^{aff[A]\,aff[B]}(\theta) \;=\; |\det(lin)| \int_0^{+\infty} [\, \psi^{AB}(k\,|\,lin^{-1} \cdot \vec{\theta}\,|\; \vec{\theta}\,') / k^r \,]\; \mathrm{d}k$$

Now, let $t = k\,|\,lin^{-1} \cdot \vec{\theta}\,|$. We have $k = |\,lin^{-1} \cdot \vec{\theta}\,|^{-1}\, t$ and $\mathrm{d}k = |\,lin^{-1} \cdot \vec{\theta}\,|^{-1}\, \mathrm{d}t$. Therefore:

$$\hat{\mathrm{F}}_r^{aff[A]\,aff[B]}(\theta) \;=\; |\det(lin)|\;\; |lin^{-1} \cdot \vec{\theta}\,|^{\,r-1} \int_0^{+\infty} [\, \psi^{AB}(t\vec{\theta}\,') / t^r \,]\; \mathrm{d}t$$

which can be rewritten (according to Eq. (9)):

$$\hat{\mathrm{F}}_r^{aff[A]\,aff[B]}(\theta) \;=\; |\det(lin)|\;\; |lin^{-1} \cdot \vec{\theta}\,|^{\,r-1}\;\; \hat{\mathrm{F}}_r^{AB}(\theta')$$

Appendix E. Proof of Property 5

In this case, $aff = lin = str = \begin{bmatrix} 1 & 0 \\ 0 & k \end{bmatrix}$, $lin^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & 1/k \end{bmatrix}$ and $\det(lin) = k$. Moreover, we have $\vec{\theta} = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}$, $lin^{-1} \cdot \vec{\theta} \;=\; \begin{bmatrix} \cos\theta \\ \dfrac{\sin\theta}{k} \end{bmatrix}$ and $|lin^{-1} \cdot \vec{\theta}\,| = k^{-1}\,[1 + (k^2 - 1)\,\cos^2\theta]^{1/2}$.

Property 6 then yields:

$$\mathrm{F}_r^{str[A]\;str[B]}(\theta) \;=\; k^{2-r}\,[1 + (k^2 - 1)\cos^2\theta\,]^{(r-1)/2}\;\; \mathrm{F}_r^{AB}(\theta')$$

with $\theta' = \angle(lin^{-1} \cdot \vec{\theta}\,) = \angle\begin{bmatrix} \cos\theta \\ \dfrac{\sin\theta}{k} \end{bmatrix}$, i.e., θ' belongs to $[0,2\pi)$ and is congruent to arctan(k^{-1}tanθ) if cosθ is positive; to θ if cosθ is zero; to arctan(k^{-1}tanθ)+π otherwise.

Table 1

$f_r(s)$	$s=0$	$s=1$	$s>1$
$r<1$	$1/[(1-r)(2-r)]$	$(2^{2-r}-2)/[(1-r)(2-r)]$	$[(s+1)^{2-r}-2s^{2-r}+(s-1)^{2-r}]/[(1-r)(2-r)]$
$r=1$	$+\infty$	$2\ln(2)$	$(s+1)\ln(s+1)-2s\ln(s)+(s-1)\ln(s-1)$
$1<r<2$	$+\infty$	$(2^{2-r}-2)/[(1-r)(2-r)]$	$[(s+1)^{2-r}-2s^{2-r}+(s-1)^{2-r}]/[(1-r)(2-r)]$
$r=2$	$+\infty$	$+\infty$	$\ln\{s^2/[(s-1)(s+1)]\}$
$r>2$	$+\infty$	$+\infty$	$[(s+1)^{2-r}-2s^{2-r}+(s-1)^{2-r}]/[(1-r)(2-r)]$

Table 2

	N	*K*	*M*	*r*	*shape*	*position*
F	YES (N to $N\sqrt{N}$) [1]	YES (K)	YES (M or M^2) [2]	NO [3]	YES [4]	YES [5]
$\overline{\text{F}}$	YES (N)	YES (K)	NO	N/A	NO	YES [5]
$\widetilde{\text{F}}$	YES ($N \log N$)	NO	NO	N/A	NO	NO
$\hat{\text{F}}$	YES ($N \log N$)	NO	NO	NO	NO	NO

[1] N for convex objects, $N\sqrt{N}$ for fractal-like objects

[2] M with the single sum scheme, M^2 with the double sum scheme

[3] F_1^{AB} and F_2^{AB} histograms, however, are processed faster than other F_r^{AB} histograms

[4] see [1] above

[5] processing times are higher when there are forces in all directions (e.g., overlapping objects)

Figure 1

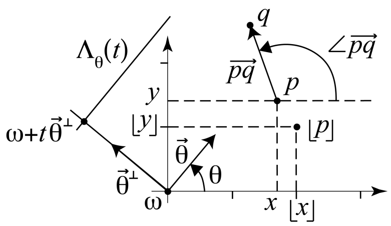
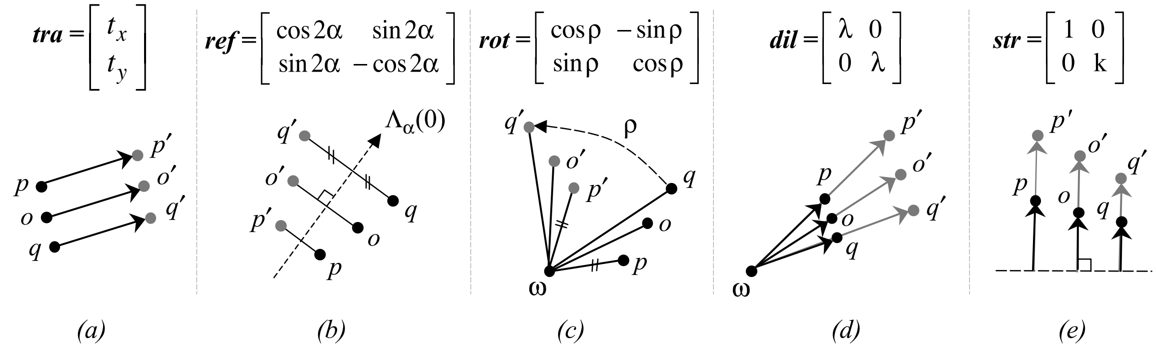
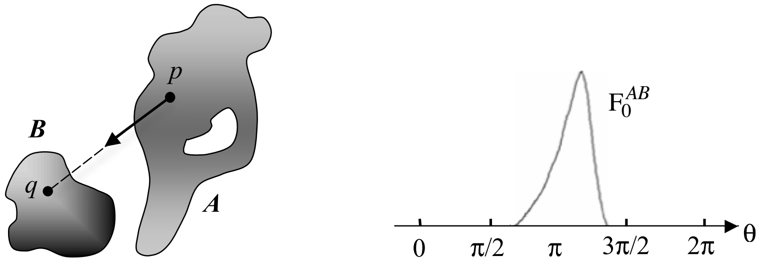

q
$\Lambda_\theta(t)$
$\angle\overrightarrow{pq}$
$\overrightarrow{pq}$
p
y
$\lfloor y \rfloor$
$\lfloor p \rfloor$
$\omega+t\vec{\theta}^\perp$
$\vec{\theta}$
θ
$\vec{\theta}^\perp$
ω
x
$\lfloor x \rfloor$

Figure 2


$\boldsymbol{tra} = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$
p
o
q
p'
o'
q'
$\boldsymbol{ref} = \begin{bmatrix} \cos 2\alpha & \sin 2\alpha \\ \sin 2\alpha & -\cos 2\alpha \end{bmatrix}$
q'
$\Lambda_\alpha(0)$
o'
p'
q
o
p
$\boldsymbol{rot} = \begin{bmatrix} \cos\rho & -\sin\rho \\ \sin\rho & \cos\rho \end{bmatrix}$
q'
ρ
o'
p'
q
o
p
ω
$\boldsymbol{dil} = \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}$
p'
p
o'
o
q'
q
ω
$\boldsymbol{str} = \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{k} \end{bmatrix}$
p'
o'
q'
p
o
q

(a) *(b)* *(c)* *(d)* *(e)*

Figure 3



(a) *(b)*

Figure 4

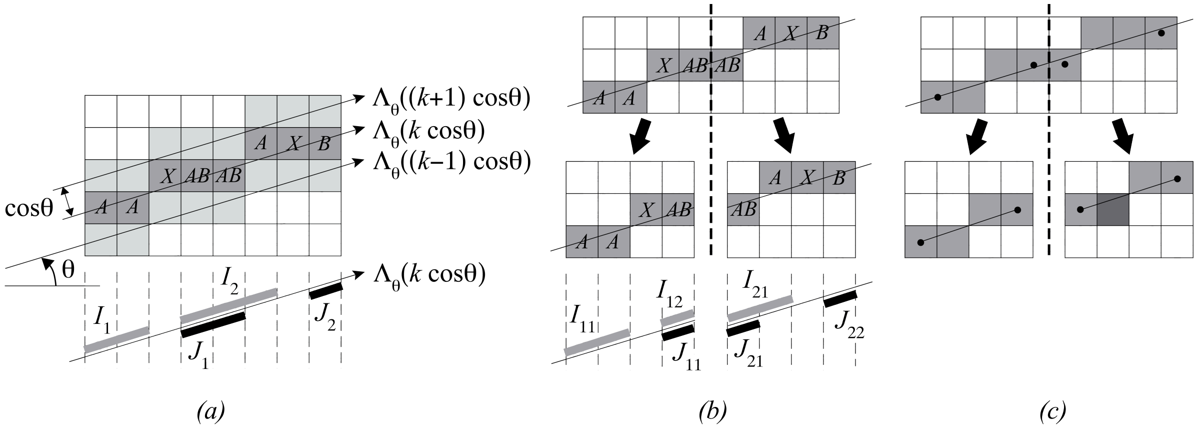
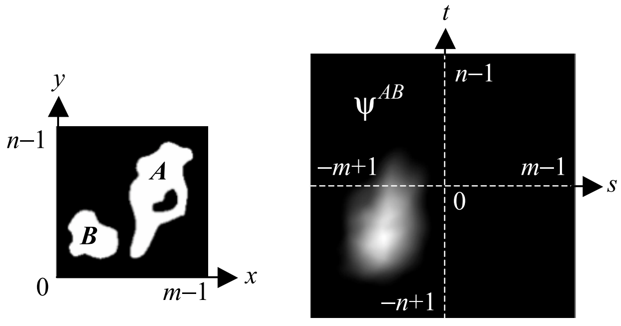
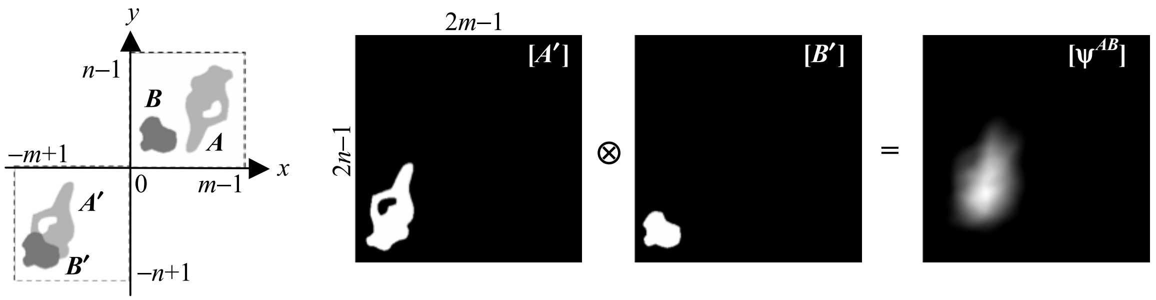

Λθ((k+1) cosθ)
Λθ(k cosθ)
Λθ((k−1) cosθ)
cosθ
θ
A
A
X
AB
AB
A
X
B
I1
I2
J1
J2
Λθ(k cosθ)
A
X
B
X
AB
AB
A
A
X
AB
AB
A
X
B
A
A
I11
I12
J11
I21
J21
J22
(a)
(b)
(c)

Figure 5



(a) *(b)*

Figure 6



(a) (b) (c) (d)

Figure 7

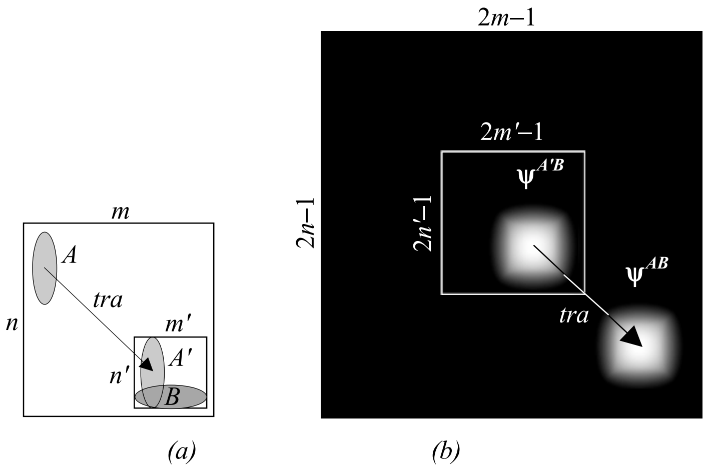
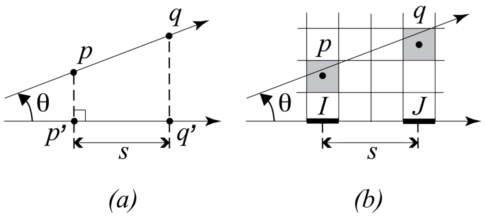


Figure 8



(a)

(b)

Figure 9

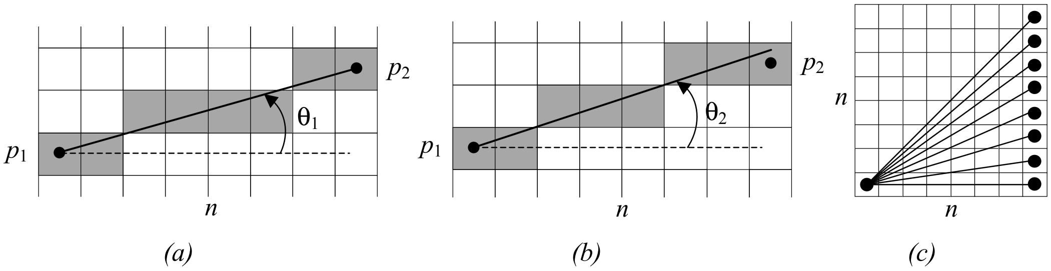
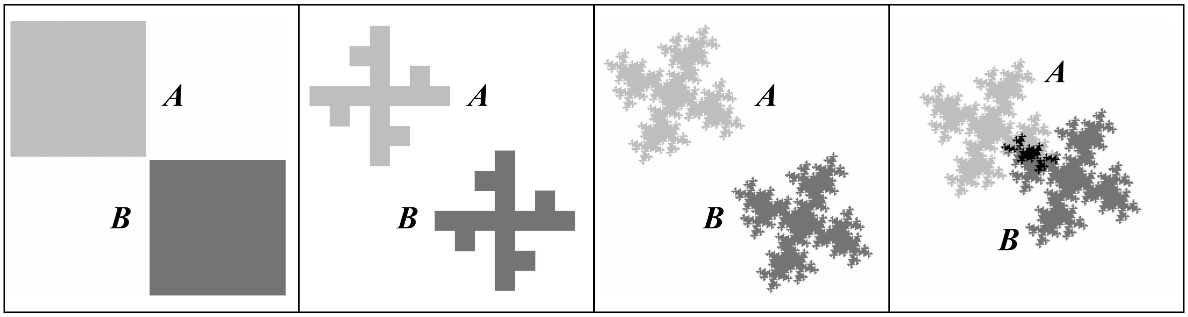
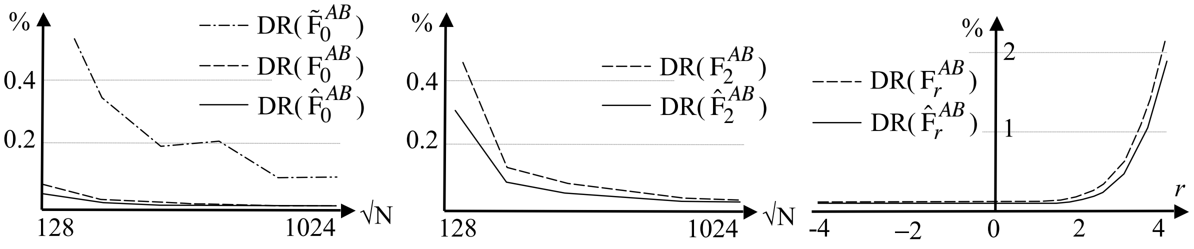


Figure 10



(a) *(b)* *(c)* *(d)*

Figure 11



(a) *(b)* *(c)*

Figure 12

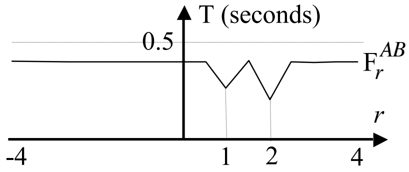


Figure 13

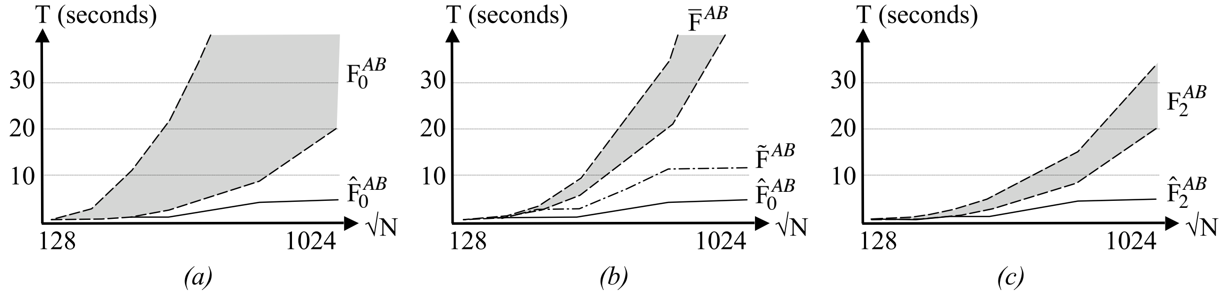
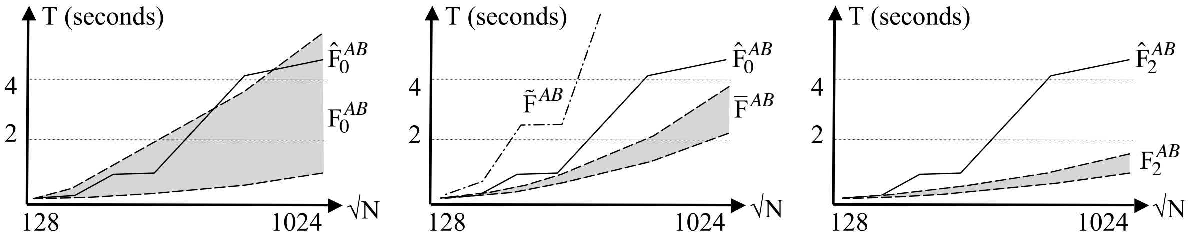
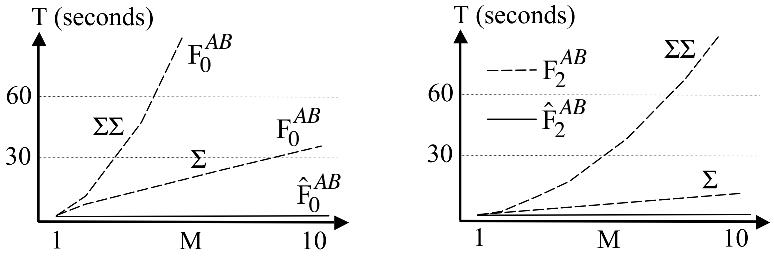

T (seconds)
30
20
10
F_0^{AB}
$\hat{F}_0^{AB}$
128
1024
√N
(a)
T (seconds)
$\overline{F}^{AB}$
30
20
10
$\tilde{F}^{AB}$
$\hat{F}_0^{AB}$
128
1024
√N
(b)
T (seconds)
30
20
10
F_2^{AB}
$\hat{F}_2^{AB}$
128
1024
√N
(c)

Figure 14



(a) *(b)* *(c)*

Figure 15


T (seconds)
F_0^{AB}
60
$\Sigma\Sigma$
F_0^{AB}
30
Σ
$\hat{F}_0^{AB}$
1
M
10
T (seconds)
F_2^{AB}
$\Sigma\Sigma$
60
$\hat{F}_2^{AB}$
30
Σ
1
M
10

(a)

(b)