A note on contextual performance prediction for image analysis algorithms

B. Chalmond * and C. Graffigne +

January 2000

* CMLA
Bernard.Chalmond@cmla.ens-cachan.fr
http://www.cmla.ens-cachan.fr/~chalmond/

+ PRISME, Paris V University
Christine.Graffigne@math-info.univ-paris5.fr

Abstract. This paper explores a generic approach to predict the output accuracy of an algorithm without running it, by a careful examination of the local context. Such a performance prediction will allow to qualify the appropriateness of an algorithm to treat images with given properties (contrast, resolution, noise, richness in details, contours or textures, etc.) resulting either from experimental acquisition conditions or from a specific type of scene. We have to answer the following question: a context c being given at any site, what will be the performance? In our experiments, c is described by three contextual variables: Gabor components, entropy and signal/noise ratio. Two different approaches have been explored. In the first one the prediction function is determined from training using a logistic regression model. In the second one which is more dedicated to the performance of a chain of algorithms, the performance is determined from an analytical computation. These techniques are illustrated on aerial infra-red images for two types of algorithm: edge detection and edge linking.
1 Introduction

For many image processing applications where different possible procedures may be concurrently used to solve a given task, it may be interesting to evaluate the ability of these procedures to perform the task on a given image. In this paper, the ability is learned using a learning set of images. Based on this learning, the evaluation is then done on every given image without running the procedure. This is what we call "performance prediction" of the algorithm. Performance prediction will allow to qualify the appropriateness of an algorithm to treat images with given properties (contrast, resolution, noise, richness in details, contours or textures, etc.) resulting either from experimental acquisition conditions or from a specific type of scene. In the next lines we call "context" these factors of difference whatever their origin (either depending on the acquisition or on the scene to be processed).

Performance prediction as defined here is a part of the domain of performance characterization which received a great attention in the computer vision community [12, 20]. Several ways exist to assess the adequacy of an algorithm to a given task. Our approach is different from the previous approaches [5, 7, 11, 12, 13, 19]. We are not interested in optimizing the task-oriented algorithms, but only to evaluate their capability to provide adequate results on some specific images. It is a common experience that, depending on the context, a given algorithm will perform well and lead to dependable results on a region of an image while in another region, still because of the context, the same algorithm will perform poorly and provide unreliable results. Our objective is to extract from a limited sample of images well representing the context a measure of the performances of the method in order, for instance, to choose among a set of methods, or, when several sensors (or several parameters of these sensors) are at the user’s disposal, to choose among them.

This note is organized as follows: in Section 2 we describe the proposed approach and provide the definitions and the methodology of performance prediction. In Section 3 we present the learning stage based on a logistic model and a ML estimator. In Section 3 and 4, our approach is detailed for two types of algorithm and the results are illustrated on aerial infra-red images: section 3 is devoted to edge detection, a low level single operation the result of which is a binary information, and section 4 is about edge linking (called here line tracking), a pile of several successive image processing tasks.

Beside this presentation, other algorithm performances are currently studied using the learning technique of this note, and especially the motion estimation algorithm in collaboration with ENST-Paris. The whole study will be presented in a forthcoming paper devoted to the case of the performance prediction of a single algorithm.

2 Framework

2.1 Problem Statement

Let us stay for simplicity with the example of the edge (element) detection algorithm. We will denote \( Y^a \) the binary variable of detected edges for a given algorithm \( a \), \( Y^a_s = 1 \) if there is an edge detected at position \( s \) and 0 otherwise. We will also use the "ground truth" \( Y^g \) as many authors do although it is not absolutely clear that such a concept really exists. The ground truth represents the exact state of nature as long as edges are concerned. Our objective is clearly to obtain a detection as close as possible to \( Y^g \). In the framework of the statistical hypothesis testing \( (H_0: \text{no edge at } s, \text{versus } H_1: \text{edge at } s) \) or in another words \( H_0: Y^g_s = 0, \)

1. Submitted to IEEE Trans. on Image Processing
versus \( H_1: Y_s^a = 1 \), the two distributions of probability \( \{ P(Y_s^a = r | Y_s^a = 1), r = 0, 1 \} \) and \( \{ P(Y_s^a = r | Y_s^a = 0), r = 0, 1 \} \) are classically used to measure the quality of the edge detection algorithm. We will consider here the dual distributions \( \{ P(Y_s^a = r | Y_s^a = 1), r = 0, 1 \} \) and \( \{ P(Y_s^a = r | Y_s^a = 0), r = 0, 1 \} \) which are related, from a Bayesian point of view, to the posterior distributions on the ground thrue if no particular prior information is given on them except the uniform one.

**Non contextual measurement**

When the observations are independently identically distributed (i.i.d.), Information Theory gives a theoretical expression of the two probabilities of error \( \alpha = P(Y_s^a = 1 | Y_s^a = 0) \) and \( \beta = P(Y_s^a = 0 | Y_s^a = 1) \), (see [8], Chap. 12). These theoretical results have been used in [14, 20] for performance evaluation. This approach does not take into account the context variability since the observations are assumed to be identically distributed. Let us gives some elements of this classical approach.

Let \( \bar{D} = (D_1, ..., D_n) \) be an i.i.d. sample of a random variable \( D \) with an unknown distribution \( \{ \pi(\ell), \ell \in J \} \). We want to test between two hypotheses:

\[
H_0 : \pi = \pi_0, \quad \text{versus} \quad H_1 : \pi = \pi_1,
\]

\( \pi_1 \) and \( \pi_2 \) given. The optimal decision rule [15] is based on the log-likelihood ratio \( L(\bar{D}) = \log \prod \frac{\pi_1(d)}{\pi_0(d)} \). Then, the random decision is: "Accept \( H_1 \) if \( L(\bar{D}) > \tau \) and \( H_0 \) otherwise".

For instance, if \( \pi_0 \) is the Gaussian law \( \mathcal{G}(\mu_0, \sigma^2) \) and \( \pi_1 \) is the Gaussian law \( \mathcal{G}(\mu_1, \sigma^2) \) with \( \mu_1 > \mu_0 \), this decision rule is equivalent to: "Accept \( H_1 \) if \( \bar{D} > t \) and \( H_0 \) otherwise". In our context, if we assume that the response \( D_s^a \) of the edge detector follows such a Gaussian law, we get the standard edge detector:

\[
Y_s^a = 1 \text{ if } D_s^a > t^a, \quad \text{ and } Y_s^a = 0 \text{ else},
\]

but in this case the decision is taken with a sample of size \( n = 1 \! \)!

For the hypothesis testing (1), the Sanov’s theorem expresses the two probabilities of error \( \alpha = P(L(\bar{D}) > \tau \ | \ H_0) \) and \( \beta = P(L(\bar{D}) < \tau \ | \ H_1) \). To do that, we assume that \( J \) is a finite discrete set and then we rewrite the log-likelihood ratio as: \( L(d) = \sum_{\ell} \log \frac{\pi_1(d)}{\pi_0(d)} = \sum_{\ell} n \hat{\pi}(\ell) \frac{\pi_1(d)}{\pi_0(d)} \), \( \hat{\pi}(\ell) \) being the frequency of the level \( \ell \) inside the sample. The theorem gives:

\[
\alpha = 2^{-nK(\pi_0^* \| \pi_0)}, \quad \beta = 2^{-nK(\pi_1^* \| \pi_1)},
\]

where \( \pi_0^* \) is the closest element of \( R = \{ \hat{\pi} : L(\bar{d}) > \tau \} \) in term of the Kullback-Leibler divergence \( K(\pi_0^* \| \pi_0) = \sum_d \pi_0^*(d) \log(\pi_0^*(d)/\pi_0(d)) \). Similarly, \( \pi_0^* \) is the closest element of \( R^c = \{ \hat{\pi} : L(\bar{d}) \leq \tau \} \).

For instance, let us consider the binary random variable \( Y^a \) with \( J = \{0, 1\} \) representing the response of an edge detector. We assume that on every edge elements, the distribution \( \pi_1 \) of \( Y^a \) is a Bernouilli law: \( \pi_1(1) = P(Y^a = 1 \ | \ H_1) = p_1 \), and in the background the distribution \( \pi_0 \) of \( Y^a \) is another Bernouilli law: \( \pi_0(0) = P(Y^a = 1 \ | \ H_0) = p_0 \), with \( p_1 > p_0 \). Let \( (Y_1^a, ..., Y_n^a) \) be an i.i.d. sample drawn either from edges or from the background. For testing (1), the decision rule is again: "Accept \( H_1 \) if \( Y^a > t \) and \( H_0 \) otherwise" and the probabilities of error are given by (3). For performance evaluation, these probabilities have a main drawback: the identical distributed nature of the sample means that this evaluation is valid only for a type of context. It means that we have to compute these quantities for every context of interest in order to have a contextual
based performance evaluation. This is the topic of our paper to summarize in a unique function this contextual performance evaluation.

**Contextual measurement**

Now, both \( P(Y_s^g = r|Y_s^a = 1) \) and \( P(Y_s^g = r|Y_s^a = 0) \) depend on \( s \). They also depend on the complete context of the detection, and therefore they depend on the complete image. But, in practice, they usually only depend on the context around \( s \): if exactly the same grey level configuration in a neighborhood around \( s \) is observed at another site \( t \), then the quantities \( P(Y_t^g = 1|Y_t^a = 1) \) and \( P(Y_t^g = 0|Y_t^a = 0) \) should equal \( P(Y_s^g = 1|Y_s^a = 1) \) and \( P(Y_s^g = 0|Y_s^a = 0) \) respectively. It is also our experience that the performance of edge detection algorithms depends on the surrounding of a site: for example, if it is noisy, it is probable that a mistake be made in the detection, while if the surrounding is clean the probability of obtaining a correct answer is much higher. This experience suggests to introduce the context in the modelling. Let \( c \) be a vector of observed contextual information in the neighborhood of a site \( s \) in the original image. We explicitly introduce the dependency in the context \( c \) by denoting:

\[
P_c(Y_s^g = 0|Y_s^a = 0) = \Psi_0(c) \\
P_c(Y_s^g = 1|Y_s^a = 1) = \Psi_1(c)
\]

*these two quantities being dependent on \( s \) only through the context \( c \) around \( s \).*

They measure the reliability that can be assigned to the decisions \( Y_s^a = 1 \) and \( Y_s^a = 0 \) obtained from the considered edge detection. \( \Psi_0 \) and \( \Psi_1 \) can be seen as functions defined on the set \( \mathcal{C} \) of all the possible contexts: \( c \in \mathcal{C} \rightarrow \Psi_r(c) \) with \( r = 0, 1 \). If these functions are known, it is not necessary to dispose of the result \( y \) on a given image to define \( P_c(Y_s^g = r|Y_s^a = r) \) on this image. Of course once the result of the edge detection \( y \) is available, the preceding quantities can be used directly to quantify the performance (or reliability) of \( y \). This is the more classical performance evaluation process. In this case, we use the quantity \( P_c(Y_s^g = y^g|Y_s^a = y^a) \) and obtain a reliability map associated to \( y \).

In the case where \( k \) algorithms (denoted by \( (a_1, a_2, \cdots, a_k) \)) are to be linked, the previous prediction stage has to be applied \( k \) times and the results concatenated. For instance, the performance of the chain of algorithms \( \mathcal{A} = (a_1, a_2) \) is predicted by first predicting the performance of the algorithm \( a_1 \) and then using this performance to deduce the performance of the chain. In order to prove the feasibility of this approach, a chain \( \mathcal{A} \) of two algorithms will be considered later in this paper: edge detection \( a_1 \) and line tracking \( a_2 \).

**2.2 Definitions**

Let \( I = \{I_s, \ s \in S\} \) be an image which belongs to \( E^S \) where \( E \) is the set of values taken by \( I_s \), typically \( E = \{0, \cdots, 255\} \), and assume that a set of local characteristics \( x = \{x_s, \ s \in S\} \), extracted from \( I \), is available. Such characteristics are described in section 2.3. The vector \( x_s \) describes the contextual information around \( s \). \( x_s \) can be seen as the occurrence of a random variable taking its values \( c \) in a set \( \mathcal{C} \) which does not depend on \( s \).

**Case of a single algorithm**

For sake of simplicity, this paper deals with algorithms \( a \) whose output is binary: \( Y_s^a = r \in \{0, 1\} \). It is not a theoretical limitation as it will be precise in Section 3.3.2: continuous output
Definition 1 Given a context \( x = \{x_s, s \in S\} \), the performance of the algorithm \( a \) for this context is given by the two maps \( \mathcal{P}_0 \) and \( \mathcal{P}_1 \) where:

\[
\mathcal{P}_0(s) = \Phi_0(x_s) = P_{x_s}[Y^a_s = 0 \mid Y^a_s = 0] \quad \text{and} \quad \mathcal{P}_1(s) = \Phi_1(x_s) = P_{x_s}[Y^a_s = 1 \mid Y^a_s = 1].
\]

\( \Phi_0 \) and \( \Phi_1 \) are defined in (4). \( \mathcal{P}_0 \) and \( \mathcal{P}_1 \) can be considered as a set of two images (see for example figure 5). Note that the subscripts \( r = 0, 1 \) refer to the conditions \( Y^a_s = 0 \) and \( Y^a_s = 1 \) respectively. It has to be underlined that these maps \( \mathcal{P}_0 \) and \( \mathcal{P}_1 \) do not depend on the realization \( y^a \) of \( Y^a \) and that, therefore, they can be computed independently from the algorithm \( a \) itself as soon as \( \Phi_0(c) \) and \( \Phi_1(c) \) are given. However, if \( y^a \) is available, the reliability of \( y^a \) is directly deduced from the performance according to the following definition.

Definition 2 Given \( I \), its context \( x = \{x_s, s \in S\} \) and the output \( y^a \), the reliability of \( y^a \) is given by the map \( \mathcal{F} \) where:

\[
\mathcal{F}(s) = P_{x_s}[Y^a_s = y^a_s \mid Y^a_s = y^a_s].
\]

\( \mathcal{F} \) can be considered as an image. Notice that \( \mathcal{F} \) is known as soon as \( \mathcal{P}_0, \mathcal{P}_1 \) and \( y^a \) are known since:

\[
\mathcal{F}(s) = \begin{cases} 
\mathcal{P}_0(s) & \text{if } y^a_s = 0 \\
\mathcal{P}_1(s) & \text{if } y^a_s = 1.
\end{cases}
\]

Case of an algorithm in a chain of algorithms

This situation is described here on the example of a chain \( \mathcal{A} \) made of two algorithms:

- algorithm \( a_1 \): edge detection algorithm,
- algorithm \( a_2 \): line tracking algorithm.

We denote a chain of connected edge elements as a line, \( Y^a_{s^2}, s \in S \), is the binary random output of \( a_2 \): \( Y^a_{s^2} = 1 \) if \( s \) belongs to a detected line and \( Y^a_{s^2} = 0 \) otherwise. It is clear that \( \mathcal{A} \) cannot perform well if \( a_1 \) does not perform well, since \( a_2 \) relies on the outputs of \( a_1 \). In fact, in this kind of algorithm, the output of \( a_2 \) is a merely deterministic function of \( y^a \) and the performances of \( a_2 \) only depend on the performances of \( a_1 \). A more general situation is when the performances of \( a_2 \) also depend on the context. Let \( \{x^a_{s^1}, s \in S\} \) and \( \{x^a_{s^2}, s \in S\} \) denote the contexts associated with \( a_1 \) and \( a_2 \). We pile up the performance maps:

\[
x^a_{s^1} \longrightarrow (\mathcal{P}^a_{0_{s^1}}, \mathcal{P}^a_{1_{s^1}}) \quad \text{then} \quad \tilde{x} = (x^a_{s^2}, \mathcal{P}^a_{0_{s^2}}, \mathcal{P}^a_{1_{s^2}}) \longrightarrow (\mathcal{P}_{0_{s^2}}, \mathcal{P}_{1_{s^2}}).
\]

As will be seen later (§4), we should indeed write \( \tilde{x}_s = (x^a_{s^2}, (\mathcal{P}^a_{0_{s^1}}(\ell))_{\ell \in N_s}, (\mathcal{P}^a_{1_{s^1}}(\ell))_{\ell \in N_s}) \), where \( N_s \) is a neighborhood of \( s \). Now, \( \tilde{x} \) taking the place of \( x \) in Definition 1, we get the following new definition.

Definition 3 Given a context \( x^a \) and the first algorithm performance maps \( \mathcal{P}_0^a \) and \( \mathcal{P}_1^a \), the performance of the algorithm \( a_2 \) is given by the two maps \( \mathcal{P}_0^{a_2} \) and \( \mathcal{P}_1^{a_2} \) where:

\[
\mathcal{P}_0^{a_2}(s) = P_{x^a_s}[Y^{a_2}_s = 0 \mid Y^{a_2}_s = 0] \quad \text{and} \quad \mathcal{P}_1^{a_2}(s) = P_{x^a_s}[Y^{a_2}_s = 1 \mid Y^{a_2}_s = 1].
\]
Once again, the maps $P_0^{a_2}$ and $P_1^{a_2}$ do not depend on the realization $y^{a_2}$ of $Y^{a_2}$ and therefore they can be computed before the algorithm $a_2$ is run. Once $y^{a_2}$ is available, we define the reliability $F^{a_2}$ of $y^{a_2}$. It is directly deduced from the performance as before:

$$F^{a_2}(s) = \begin{cases} P_0^{a_2}(s) & \text{if } y^{a_2} = 0 \\ P_1^{a_2}(s) & \text{if } y^{a_2} = 1. \end{cases}$$

### 2.3 Image characteristics and context

The choice of contextual variables is a typical problem of image modelling [6]. One solution of this problem consists of selecting, within a broad set of features, those which most contribute to explain the response $Y$ according to a specific criterion. [17] provides a comprehensive summary of this procedure for building a regression model. In this paper, we have adopted a "logistic regression model" (see §2.4) for which this procedure can be adapted. However we do not detail anymore the variable selection and we take the same variables in our two examples (contour detection, motion estimation), although the variable selection has to be dependant of the algorithm under study. Finally, three contextual variables have been selected providing at every site $s$ a vector $x_s$ with six components.

- **Gabor components.** For every $s$, a Gabor filtering along 4 directions $\alpha$ is made [3, 9]:
  \[ \{\alpha_k, k = 1, \ldots, 4\} = \{0, \pi/2, \pi/4, -\pi/4\}. \]
  \( \{x_s(k), k = 1, \ldots, 4\} \) is given by the convolution \( (I \ast G(k))_s \) where $G(k)$ is the Gabor filter:

  \[ G_s(k) = \cos(2\pi\omega(\cos(\alpha_k)s_1 + \sin(\alpha_k)s_2)) \exp\left(\frac{-1}{2\sigma^2}( s_1^2 + s_2^2 )\right), \quad k = 1, \ldots, 4, \]

  with \( s = (s_1, s_2) \) and \( \omega = 1/(2\sigma), \sigma = 2, 3, \ldots \) is the scale factor and $\omega$ is the wavelet frequency.

- **Entropy.** For every $s$, a local entropy $x_s(5)$ is computed in a square window $F_s$ of odd side length taking its value in \( \{7, 9, \ldots\} \). In order to obtain a reliable estimate of the entropy, the image is first quantified on 16 levels only and we compute:

  \[ x_s(5) = -\sum_{i=1}^{16} f_s^i \log(f_s^i), \]

  where $f_s^i$ denotes the occurrence of grey level $i$ in the window $F_s$. This entropy is usually large where contours and contrasted texture exist.

- **Signal/Noise ratio.** For every $s$, the image \( \{I_t, t \in F_s\} \) is locally approximated by a bicubic spline \( \{\tilde{I}_t, t \in F_s\} \). $F_s$ is a square window centered at $s$ with an odd side length taking its value in \( \{7, 9, \ldots\} \). The spline is a piecewise polynomial surface made of 4 pieces of identical sizes [6]. The last component of $x_s$ is defined as:

  \[ x_s(6) = \frac{\max_{t \in F_s} \tilde{I}_t - \min_{t \in F_s} \tilde{I}_t}{\text{s.d.}\{I_t - \tilde{I}_t, t \in F_s\}}, \]

  where s.d. is the empirical standard deviation. In the presence of a noisy smooth edge in $F_s$, this characteristic is the ratio between the amplitude of the non noisy smooth edge and the s.d. of the noise.

  Finally, a context at site $s$ is defined by a vector \( (x_s(1), \ldots, x_s(6))^T \in \mathcal{C} \subset \mathbb{R}^6 \).
2.4 Methodology

The objective is the estimation of the performance maps (5) and (7). We have to answer the question: a context $c$ being given at any site $s$, what will be the foreseeable performances $\{q_r(c), r = 0, 1\}$ of the algorithm? One of the main contributions of this paper is to propose a model for such a function in the case of the considered algorithms. For that we explored two different approaches.

- Learning approach. In the case (5), the prediction function is determined from training. It will be the case for contour detection and motion estimation (§3.3). We start from a set of learning images $I^L$ on which the outputs $Y^g$ and $Y^a$ as well as the contexts $x_s$ are computed. Let $C^L \subseteq C$ denote the set of contexts observed on $I^L$. This set provides examples which allow to estimate, at fixed context $c$ in $C^L$, the probabilities $P[Y^g = 0 | Y^a = 0]$ and $P[Y^g = 1 | Y^a = 1]$ using the occurrences of events $[Y^g = 0 | Y^a = 0]$ and $[Y^g = 1 | Y^a = 1]$ respectively, on all the sites such that $x_s = c$. Let $\hat{q}_0(c)$ and $\hat{q}_1(c)$ denote these frequencies. The prediction function is taken using the well-known logistic regression model:

$$q_r(c) = \frac{\exp[\theta_0 + \theta^T c]}{1 + \exp[\theta_0 + \theta^T c]}, \quad r = 0, 1,$$

where $(\theta_0, \theta^T)$ is the unknown vector of the model parameters. This model which performs a logistic regression is used in medicine for a long time, for example to predict the remission in cancer patients [1]. In the same spirit, this model is now widely used for pattern recognition in the context of the single-layer network which acts on the linear sum $(\theta_0 + \theta^T c)$ to give a discriminant function for two class problems [4]. The estimation of the model parameters is chosen so that $q_r(c)$ provides the best approximate of $\hat{q}_r(c)$ on the set $C^L$. It will be seen that this model presents good approximation properties for data $\hat{q}_r(c)$ on $C^L$ (see Fig. 4), as well as good prediction properties on $C$ (see Fig. 7).

- Analytical approach. The second approach is more specially devoted to the cases as (7). This approach can typically be used for all the algorithms inside a chain of algorithms except for the first one and in the case discussed before of a deterministic dependency between algorithm $a_{n-1}$ and algorithm $a_n$. In this case, it is possible to obtain an explicit computation of the performance. It is clearly very much dependent on the type of algorithm considered; an example will be given in Section 4.

3 Performance prediction by training

We now pay attention to the way of estimating the function $q_1$. Suppose that, on the training images $I^L$, two sub-sets $R_{1,1}$ and $R_{0,1}$ of the set of sites $S$ are available:

$$R_{1,1} \subset \{s \in S : Y^g_s = 1, Y^a_s = 1\}$$
$$R_{0,1} \subset \{s \in S : Y^g_s = 0, Y^a_s = 1\}.$$  \hspace{1cm} (9)

We also note: $R_1 = R_{0,1} \cup R_{1,1}$. These elements contribute to the estimation of $q_1$.

In a similar way, $R_{0,0}$, $R_{1,0}$ et $R_0$ are defined in order to model $q_0$. Remark that, at this point, we do not need a complete processing of the training images, but only a training sufficiently exhaustive to explain the output $r$ of the algorithm as a function of the context $c$. In Fig.1(b) this is illustrated for the case of edge detection where only a small number of edge elements is kept.
3.1 Logistic regression Model

**General case.** Let assume that \( x_\ell \) takes a small number of values \( \{c^{(j)}\} \) only. This provides a partition \( \{S^{(j)}\} \) of \( R_1 \) in sub-sets \( S^{(j)} = \{ s : x_\ell = c^{(j)} \} \). In this case, the estimate of \( \hat{\eta}_1(c^{(j)}) \) is nothing but the frequency:

\[
\hat{\eta}_1(c^{(j)}) = \frac{\sum_{s \in S^{(j)}} Y_s^{(j)}}{|S^{(j)}|} = \frac{K_j}{n_j},
\]

and \( K_j \) may be considered as a random variable distributed according to a binomial distribution \( B(n_j, \hat{\eta}_1(c^{(j)})) \). In this expression, \(|S^{(j)}|\) denotes the cardinality of the set \( S^{(j)} \). The logistic model relates \( \{\hat{\eta}_1(c^{(j)})\} \) with \( \{c^{(j)}\} \) by:

\[
\hat{\eta}_1(c^{(j)}) = \frac{\exp[\theta_0 + \theta^T c^{(j)}]}{1 + \exp[\theta_0 + \theta^T c^{(j)}]} = \frac{\exp\Theta^T \hat{c}^{(j)}}{1 + \exp \Theta^T \hat{c}^{(j)}}
\]

where \( \Theta^T = (\theta_0, \theta^T) \) and \( \hat{c}^{(j)} = (1, c^{(j)T}) \).

The aim is to approximate the \( \{\hat{\eta}_1(c^{(j)})\} \) by the \( \{\eta_1(c^{(j)})\} \) according to expression (11). This approximation is possible through an estimation of parameters \( \Theta \) obtained by maximizing the likelihood. For fixed \( K_j \), the likelihood associated to the Binomial distribution is:

\[
L(\Theta) = \prod_j C_{n_j}^{K_j} [\eta_1(c^{(j)})]^{K_j} [1 - \eta_1(c^{(j)})]^{n_j - K_j}.
\]

**Special case.** In a real situation, \(|C|\) being large, there may be as many contexts as sites. Every part of the partition is reduced to a singleton: \( n_j = 1 \) for all \( j \) and \( \eta_1(c^{(j)}) \equiv \eta_1(x_\ell) \). The expression (10) reduces to \( \hat{\eta}_1(x_\ell) = Y_s^{(j)} \) and the binomial distribution to the Bernouilli distribution with parameters \( \eta_1(x_\ell) \). The likelihood is written as:

\[
L(\Theta) = \prod_{s \in R_1} \eta_1(x_\ell)^{Y_s}(1 - \eta_1(x_\ell))^{1-Y_s}
\]

where \( \eta_1(x_\ell) = \frac{\exp \Theta^T \hat{x}_s}{1 + \exp \Theta^T \hat{x}_s} \).

3.2 Implementation

The parameters \( \Theta \) are estimated by maximizing the likelihood (12) as described below. We shall denote this estimate as \( \hat{\Theta} \). The log likelihood of (12) is:

\[
\log L(\Theta) = \sum_{s \in R_1} Y_s^{\theta} \Theta^T \hat{x}_s - \sum_{s \in R_1} \log(1 + \exp \Theta^T \hat{x}_s).
\]

The numerical solution of \( \hat{\Theta} = \arg \max \log L(\Theta) \) is obtained using the Newton-Raphson algorithm. It can be shown that any iteration \( t \) of this algorithm consists in solving, according to the weighted least square criterion the linear system:

\[
f^{(t)}(\Theta) = X^T \Theta + W,
\]

where \( X \) is the matrix with current line \( x_s^T \) and:

\[
f^{(t)} = \left[ \log \left( \frac{\hat{\eta}_1^{(t)}(\hat{x}_s)}{1 - \hat{\eta}_1^{(t)}(\hat{x}_s)} \right) + \frac{Y_s^{(j)} - \hat{\eta}_1^{(t)}(\hat{x}_s)}{\hat{\eta}_1^{(t)}(\hat{x}_s)(1 - \hat{\eta}_1^{(t)}(\hat{x}_s))} \right] _{s \in R_1}
\]

with \( \hat{\eta}_1^{(t)}(\hat{x}_s) = \frac{\exp x_s^T \hat{\Theta}^{(t)}}{1 + \exp x_s^T \hat{\Theta}^{(t)}} \).
Here $W$ is a residual random vector, the variance matrix of which equals:

$$\Gamma^{(t)} = \text{Diag}\left[\frac{1}{\Psi_1^{(t)}(\bar{x}_s)(1-\Psi_1^{(t)}(\bar{x}_s))}\right].$$

These formulas are explained in [1] and [21]. The weighted least square estimate is: $\hat{\Theta}^{(t)} = [X^T(\Gamma^{(t)})^{-1}X]^{-1}X^T(\Gamma^{(t)})^{-1}f^{(t)}$.

Note that the logistic model plays a double role. From one side, it provides a discrimination axis between $Y^o_s = 1$ and $Y^o_s = 0$ values, with respect to the context which is reduced to the axis $\hat{\delta} = \hat{\Theta}^T\bar{x}_s$ and compresses the six contextual variables contained in $x_s$. From the other side it allows to determine a 1D logistic function strictly growing on $[0,1]$, a function which expresses the performance as a function of context summed up by $\delta \in \mathbb{R}$. The estimations of these two entities, i.e. discrimination axis and 1D logistic function, are made simultaneously through $\Theta$. The quality of these estimates appeared to be far better than when estimating first the axis, then the logistic function. This is especially the case when the axis is obtained from Principal Component Analysis [16] (which, on our base $\mathcal{I}_L$ represents 99% of the inertia of the $x_s$).

Fig.2 is a typical graph of the estimate of the maximum likelihood estimation of $\Psi_1$ obtained in the case of edge detection ($\S 3.3$). It has been drawn versus the logistic axis $\hat{\delta} = \hat{\Theta}^T\bar{x}_s$, $\bar{x}_s \in \mathbb{R}^7$.

**Interpretation.** The previous model having been estimated, it may be refined. The main interest of this refinement will be to clarify the multidimensional logistic model (13). Indeed, as shown on Fig.2, due to the absence of frequencies between 0 and 1, we can hardly see how the curve is fitted to the data, and the discriminant effect of the axis is difficult to appreciate. To alleviate this drawback, we propose to come back to the general case ($\S 3.1$) where the logistic model fits frequencies between 0 and 1 (see Fig.4). To compute these frequencies (10), we have to build a partition $\{S^{(j)}, j = 1, ..., J\}$ of $R_1$. For this building, we use the previous axis $\hat{\delta}$ as follows.

At first a partition of the axis $\hat{\delta}$ is made (Fig.3). Let $\delta^{(j)}$ denote the centers of the intervals which build up the partition. They induce a partition $\{\mathcal{C}_s^{(j)}\}$ of the local contexts:

$$\mathcal{C}_s^{(j)} = \{x_s: \hat{\Theta}^T\bar{x}_s \in \text{Int}(\delta^{(j)}), s \in R_1\},$$

where Int$(\delta^{(j)})$ represents the interval of center $\delta^{(j)}$. For each $\mathcal{C}_s^{(j)}$, let $c^{(j)}$ be the $x_s$ value for which $\hat{\Theta}^T\bar{x}_s$ is the nearest value of $\delta^{(j)}$. The number of intervals should be chosen large so that each part $\mathcal{C}_s^{(j)}$ only contains sites of similar context, that is, $x_s \approx c^{(j)}$, $\forall x_s \in \mathcal{C}_s^{(j)}$. Note that sites of a same part are not necessary connected. Finally, $S^{(j)}$ is the set of sites associated to $\mathcal{C}_s^{(j)}$: $S^{(j)} = \{s : x_s \in \mathcal{C}_s^{(j)}\}$, the common context value being $c^{(j)}$. Based on this partition, the frequencies (10) can be computed. In Fig.3, the two graphs display theses frequencies, or more exactly the populations $K_j$ of correct detections and $n_j - K_j$ of false detections computed on the selected partition.

Finally, we approximate the frequencies $K_j/n_j$ by a 1D logistic model of type (11):

$$\Psi_1(c^{(j)}) \equiv \Psi_1(\delta^{(j)}) = \frac{\exp(\zeta_0 + \zeta_1\delta^{(j)})}{1 + \exp(\zeta_0 + \zeta_1\delta^{(j)})}, \quad j = 1, ..., J.$$
Other estimations. The same procedure can be applied for the prediction of $\Psi_0$ from the regions:

$$
R_{0,0} \subset \{ s \in S : Y^s_0 = 0, Y^a_0 = 0 \}
$$

$$
R_{1,0} \subset \{ s \in S : Y^s_0 = 1, Y^a_0 = 0 \}.
$$

This procedure also allows to predict the probability of false positives $\alpha = P[ Y^a_0 = 1 \mid Y^s_0 = 0 ]$ and the probability of false negatives $\beta = P[ Y^s_0 = 0 \mid Y^a_0 = 1 ]$, often used as quality criteria (see Section 2.1). In order to do so, the regions $\{ R_{0,0}, R_{1,0} \}$ and $\{ R_{1,1}, R_{0,1} \}$ must be used.

3.3 Experimental results

To estimate $\Psi_1(c)$, it is necessary to have the sets $R_{1,1}$ and $R_{0,1}$ as defined in (9). Therefore, we have to extract true edges from some regions in $T^L$. A first method to perform this extraction relies on the subjective evaluation of edges by people [13]. The danger in this case is that edges and context be not consistent; for instance, at sites $s$ and $s'$ contextually very close, $(x_s \approx x_{s'})$, opposite decisions $Y^s_y = 1$ and $Y^{y}_{s'} = 0$ may be associated. Another method would be to use a higher level algorithm as for example a segmentation algorithm. But in this case the edges map is rather different from the one obtained by edge detection and here again un-consistency may be introduced. We decided to use edges produced by an edge detector similar to the one that will be used later for performance prediction, but adding a post-processing which guarantees edges that are usually unaccessible with the edge detector only. This is done for instance using a regularization procedure as described in [10]. Remember that we do not need a complete extraction of edges, but we want a set of samples exhaustive enough to estimate the functions $\Psi_r(c)$. In Annex the edge detection on $T^L$ before the post-processing is described.

Based on these definitions, and after computing the contexts $\{ x_s, s \in S \}$ on every image $l \in T^L$, the functions $\Psi_{0}(c)$ and $\Psi_{1}(c)$ defined in (13) are fitted according to the maximum likelihood principle. Fig.5 presents the associated performance maps $\{ P_r, r = 0, 1 \}$ in the case of an algorithm based on the Sobel edge detector for an image $I \notin I^L$. Recall that the two prediction maps have been computed without knowing the output $y^a$ of the algorithm. Let us emphasized that $P_0$ is not the negative image of $P_1$. On this example, the sites such that $P_1(s)$ is close to 1 are much more concentrated than the sites such that $P_0(s)$ is close to 0, these last ones following every discontinuity in the image. The question now would be how to combine $P_0$ and $P_1$ to build an accurate decision rule concerning the performance of the given algorithm.

4 Analytical approach for line tracking

We will present in this section a very simple example of the treatment of the second algorithm in a chain of two algorithms. Here, the performance maps will be computed directly through coarse approximations of the probabilities and there is no training. This approach, which is different from the one presented earlier, has been chosen both because it was feasible according to the problem of line tracking and in order to illustrate approximations as opposed to training. Note however that the approximations proposed here are ad-hoc.

The line tracking algorithm $a_2$ belongs to a chain of algorithms in which $a_1$ is an edge detector. In our experiments, we used the edge detector presented in section 3.3 and the corresponding performance maps $\{ P_r^{a_1}, r = 0, 1 \}$. The creation of a line by $a_2$ (i.e. a chain of connected edge elements) is done considering only the output of the edge detection: edges are chained using the 4 nearest neighbors and chains shorter than 5 are deleted. There are no edges created, for
example in order to concatenate two chains. This is not necessarily the case for sophisticated algorithms [2, 10] but we chose to use for a start a rather simple approach in order to illustrate the case of a chain of algorithms introduced in section 2.2.

The computation of the performance maps $P_{0}^{a_2}$ and $P_{1}^{a_2}$ of $a_2$ is based only on the performance maps of the $a_1$ algorithm and we do not use any context $x^{a_2}$, therefore the input (8) is $x = (P_{0}^{a_1}, P_{1}^{a_1})$. The output of $a_2$ is a binary image of lines:

$$Y_s^{a_2} = 1 \text{ if } s \text{ belongs to a line}$$

$$Y_s^{a_2} = 0 \text{ otherwise},$$

and the goal is to compute the performances defined in (7):

$$P_{0}^{a_2}(s) = P_{x_s}^{a_2}[Y_s^{a_2} = 0 \mid Y_s^{a_2} = 0]$$

$$P_{1}^{a_2}(s) = P_{x_s}^{a_2}[Y_s^{a_2} = 1 \mid Y_s^{a_2} = 1].$$

The existence of a true line is obviously related to the existence of true edge elements. We consider that the event $Y_s^{a_2} = 1$ is realized when there is a "small chain" containing $s$, a small chain being a set $\ell$ of 5 true edge elements such that:

- $\ell$ is connected according to the 8 connectivity;
- $\ell$ lies in a window $N_s$ of size $5 \times 5$ pixels centered on $s$;
- each site of $\ell$ has at most 2 neighbors in $\ell$.

Let $L_s$ be the set of all small chains containing $s, \ell \in L_s$ is written $\ell = \{\ell_1, \ell_2, \ell_3, \ell_4, \ell_5\}$. Assuming the conditional independence of the $Y_{\ell_i}^{a_1}$ given the $Y_{\ell_i}^{a_1}$, the two performances are approximated by:

$$P_{0}^{a_2}(s) \approx 1 - \max_{\ell \in L_s} \prod_{i=1}^{5} P_{x_{\ell_i}}^{a_1}[Y_{\ell_i}^{a_1} = 1 \mid Y_{\ell_i}^{a_1} = 0] = 1 - \max_{\ell \in L_s} \prod_{i=1}^{5} (1 - P_{0}^{a_1}(\ell_i)) \quad (15)$$

$$P_{1}^{a_2}(s) \approx \max_{\ell \in L_s} \prod_{i=1}^{5} P_{x_{\ell_i}}^{a_1}[Y_{\ell_i}^{a_1} = 1 \mid Y_{\ell_i}^{a_1} = 1] = \max_{\ell \in L_s} \prod_{i=1}^{5} P_{1}^{a_1}(\ell_i). \quad (16)$$

The main lines of the derivation of (16) are now given, (15) being obtained in a similar way. $P_{1}^{a_2}(s)$ is the probability that a true line exists given that a line has been found by $a_2$. In terms of "small chains", we are looking for the probability that there exists a true small chain centered at $s$ given that there is an observed one. Many possible true small chains may exist (as much as the cardinality of $L_s$), and therefore the probability that the true small chain exists is difficult to compute. However, we will first verify that the main term in this probability is:

$$\max_{\ell \in L_s} P[Y_{\ell_i}^{a_1} = 1, i = 1, \ldots, 5 \mid Y_s^{a_2} = 1]. \quad (17)$$

Let $\tilde{\ell} = \{\tilde{\ell}_1, \tilde{\ell}_2, \ldots, \tilde{\ell}_5\}$ denote the argument of (17) then

$$P_{x_s}^{a_2}[Y_s^{a_2} = 1 \mid Y_s^{a_2} = 1] = P[Y_{\ell_i}^{a_1} = 1, i = 1, \ldots, 5 \mid Y_{\ell_i}^{a_2} = 1]$$

$$+ P[\exists \ell \in L_s, \ell \neq \tilde{\ell}, Y_{\ell_i}^{a_1} = 1, i = 1, \ldots, 5] \text{ and } \exists i \in \{1, \ldots, 5\}, Y_{\ell_i}^{a_1} = 0 \mid Y_s^{a_2} = 1].$$

The second term in this sum which we will denote by $B$ is small compared to the first one (denoted by $A$):

- If $A$ is large, then there is a prominent orientation and it is not likely that there should be two of them. Due to the way $P[Y_{\ell_i}^{a_1} = 1 \mid Y_{\ell_i}^{a_1} = 1]$ is built, the occurrence of a true
line is in practice associated to a rather thick band of high values in \( P_{a} \). Therefore, it is very unlikely that there should be another \( \ell \) in \( L \) with an orientation close to the one of \( \hat{\ell} \). In case \( \ell \) has an orientation really different from the orientation of \( \hat{\ell} \), the associated probability is small unless two main orientations of lines centered at \( s \) exist. In this case, the probability \( P_{a}^{\mu}(s) \) is underestimated but is still high.

- If \( A \) is small, then there is no orientation clearly identified. The probability is certainly again underestimated but it is still consistent with the reliability that one would attribute in such occurrences.

Then, since the localization of detected edges is usually not very precise, we assume that if \( Y_{s}^{a2} = 1 \) then all the \( Y_{i}^{a1}, t \) in a square neighborhood of size 5 around \( s \), may potentially be equal to 1. Therefore, we approximate \( P[Y_{s}^{a2} = 1, i = 1, \ldots, 5 \mid Y_{s}^{a2} = 1] \) by \( P[Y_{i}^{a2} = 1, i = 1, \ldots, 5 \mid Y_{s}^{a2} = 1, i = 1, \ldots, 5] \).

Finally, the conditional independence leads to (16). These approximations (15) and (16) are certainly rather coarse but they allow for a much faster computation. Fig.6 illustrates the results obtained on an infrared image. The performance map \( P_{a}^{\mu} \) is given in Fig.6(e). It should be interpreted this way: - black pixels: the probability that there is no real line knowing that there is no detected line is small. The black pixels point out regions where an output of “no line” from \( a_{2} \) is not reliable; - white pixels: the probability that there is no real line knowing that there is no detected line is high. The white pixels point out regions where an output of “no line” from \( a_{2} \) is reliable. The performance map \( P_{a}^{\mu} \) given in Fig.6(f) should be interpreted in a dual way. The reliability image \( F_{a}^{\mu} \) in Fig.6(g) is easier to understand: white means that the given output of algorithm \( a_{2} \) (Fig.6(h)) is reliable, while black means it is not reliable. Most of the dark regions correspond to relatively high probability of presence of a true line while there is none detected. Some of the isolated dark lines point out detected lines that are not reliable.

5 Conclusion

We explored a generic approach to predict the output of an algorithm without running it, by a careful examination of the local context. Our motivation is the need to characterize the behavior of different algorithms under different experimental conditions. For instance we want to give an answer to the question: what will be the performance of the edge detector in case of hilly landscape or in case of poor weather? We have presented an efficient methodology to learn from some training images the statistical dependency between some well-chosen image features and the future output of the algorithm. In most of the techniques of algorithm evaluation, the difficult problem is the definition of the ground truth. It is not so severe for the proposed solution since we do not need a complete knowledge of the state of nature on all the training images, but only a dense enough sampling set to reliably estimate the relationship between the local context and the algorithm output.

Three problems have not been addressed in this paper. First, we never discussed the choice of the training set \( I^{L} \). Our experiments have been conducted on a small set of infra-red aerial images. A more systematic test is currently being performed. This is clearly a problem of generalization. It seems that the logistic model is quite robust. On Fig. 7, the prediction maps of a visible range image present a good quality although they have been learned only from infra-red images. Second, the choice of optimized indicators of the context has never been addressed, we took from our previous experience some indicators which appeared to work rather well (cf. Section 2.3). Better indicators probably exist and a methodology to determine adequate indicators
has to be found. The third point concerns the way to exploit the performance maps. The prediction performance brings an essential contribution to the high level system design. For instance, it permits to optimize the hardware / software trade-off when designing an optronic equipment: for a given scene, the hardware characteristics like MTF (Modulation Transfer Function) have an impact on the image properties. In that case it is possible to compare, for a given scene, the outputs obtained from two different hardwares, via the performance quantization provided by $\xi_0$ and $\xi_1$.

Annex: Edge detection ground-truth

The objective is to determine a reliable set of contours $Y_s^\beta$. For any image $I \in I^L$, the detection algorithm calculates at every pixel $s$ a local variable $D_s^a$ which reflects the gradient of the grey level in the image. As in (2), the detection is based on the following decision rule:

$$Y_s^a = 1 \text{ if } D_s^a > t^a, \quad \text{and} \quad Y_s^a = 0 \text{ else}.$$  

$t^a$ is a decision threshold which is tuned according to an edge strength $\delta^a$. Let assume that $D_s^a$ is related to the context $x_s$ according to the model:

$$D_s^a = \mu_s(x_s) + W_s \equiv \delta_s + W_s,$$

where $W_s$ is zero-mean white noise and $\mu_s$ is the mean value of the random variable $D_s^a$. For an edge strength $\delta^a$ ($H_1$ hypothesis) and a non-contour strength $\delta^0$ ($H_0$ hypothesis), the power $(1 - \beta_s)$ of the detector and its risk $\alpha_s$ are classically defined as (see [8, 15]):

$$1 - \beta_s(x_s) = P[D_s^a > t^a \mid \mu_s(x_s) = \delta^a]$$

$$\alpha_s(x_s) = P[D_s^a > t^a \mid \mu_s(x_s) = \delta^0].$$

**Good decision region.** It is the difference $\delta^a - \delta^0$ which makes the detector power. Therefore, when this difference increases, the power grows to 1. In this case we may use a high threshold $t^0 > t^a$ to determine true contour sites: $D_s^a > t^0 \Rightarrow Y_s^\beta = 1$. Let us denote $R_{1,1}$ these sites which build up a good decision region since $D_s^a > t^0 \Rightarrow D_s^a > t^a \Rightarrow Y_s^a = 1$. The power is close to 1. On the contrary, when $\delta^a - \delta^0 \approx 0$, then $(1 - \beta_s) \approx \alpha_s$. In order to define the sites without real contours, the detector is used with a low threshold $t^0$ so that $t^0 < t^a$: $D_s^a < t^0 \Rightarrow Y_s^\beta = 0$. The set of such sites constitute a region of good decision since $D_s^a < t^0 \Rightarrow D_s^a < t^a \Rightarrow Y_s^a = 0$. On the images, $R_{1,1}$ is mainly attached to the clear edges and $R_{0,0}$ is made of a set of sites spread over the background.

**False detection regions.** False detections mostly appear for amplitudes close to $\delta^a$. Therefore, an interval $[t^a - h, t^a + h]$ selected in a way that $t^0 < t^a - h$ and $t^a + h < t^0$ with $h$ low, provides many false detections. Let us denote $R_{1,0}$ and $R_{0,1}$ the false detection region $R_s = \{s : t^a - h < D_s^a < t^a + h\}$. We do not know how to detect these regions. But the estimation problem at hand can be seen as a discrimination problem between good and false decisions with respect to the context. We propose to discriminate the detector behavior by comparing $R_s$ with $R_{1,1}$ and $R_{0,0}$. Therefore we propose to use:

$$R_{1,0} = R_{0,1} = \{s : t^a - h < D_s^a < t^a + h\}.$$  

As a matter of example, for Sobel edge detector, it has been chosen empirically: $t^0 = 1.3t^a$, $t^0 = 0.5t^a$, $t^0 - h = 0.9t^a$ and $t^a + h = 1.1t^a$. $R_s$ denotes inaccurate edges. They mostly are isolated edges or edges at the extremities of contours.
Acknowledgments. The authors wish to express their thanks to Robert Azencott and Michel Prenat for suggesting the subject and for stimulating discussions.

Références


Fig. 1: (a) An example of a training infra-red image. (b) $R_{1,1}$ (yellow points) and $R_{0,1}$ (white points).
Fig. 2: Multidimensional logistic model with binary variables. (The probabilities $\hat{q}_1(\hat{\delta})$ have been ranked along $\hat{\delta} = \hat{\Theta}^T \hat{c}$. The values $r = 0, 1$ of the observed $Y_r^g$ are plotted with |).

Fig. 3: Context classes along the logistic axis: Population of correct detection (above) and false detection (below) for each context class.

Fig. 4: Unidimensional logistic model and frequencies (plotted with +). (The $\hat{q}_1(\hat{\delta})$ values have been ranked along the logistic axis).
**Fig. 5:** Edge detection algorithm. (a) An aerial infrared image. (b) Map $\mathcal{P}_1$. (c) Output $y^\circ$. (d) Map $\mathcal{P}_0$. 
Fig. 6: Line tracking algorithm. (a) An aerial infrared image. (b) Output $y_{a1}$ of the edge detection algorithm. (c) Map $P_{0a1}$. (d) Map $P_{1a1}$. (e) Map $P_{0a2}$. (f) Map $P_{1a2}$. (g) Map $F_{a2}$. (h) Output $y_{a2}$ of the line tracking algorithm.
Fig. 7: Prediction performance of an edge detection algorithm. (a) A visible light image. (b) Map $\mathcal{P}_1$. 