Chapter 5
Wald's Sequential Analysis for Time-constrained Vision Problems

Jiří Matas and Jan Šochman

5.1 Introduction

In many decision problems in computer vision, both classification errors and time to decision characterise the quality of an algorithmic solution. This is especially true for applications of vision to robotics where real-time response is typically required.

Time-constrained classification, detection and matching problems can be often formalised in the framework of sequential decision-making. We show how to derive quasi-optimal time-constrained solutions for three different vision problems by applying Wald's sequential analysis. In particular, we adapt and generalise Wald's sequential probability ratio test (SPRT) and apply it to the three vision problems: (i) face detection, (ii) real-time detection of distinguished regions (interest points) and (iii) establishing correspondences by the RANSAC algorithm with application e.g. in SLAM, 3D reconstruction and object recognition.

In the face detection problem, we are interested in learning the fastest detector satisfying constraints on false positive and false negative rates. We solve the problem by WaldBoost [15], a combination of Wald's sequential probability ratio test and AdaBoost learning [2]. The solution can be viewed as a principled way to build a close-to-optimal "cascade of classifiers" [22]. Naturally, the approach is applicable to other classes of objects.

In the interest point detection emulation, we show how a fast (real-time) implementation of the Hessian-Laplace detector [9] is obtained by WaldBoost [16]. The emulated detector provides a training set of positive and negative examples of interest points. WaldBoost finds an approximation to the detector output in terms of a linear combination of efficiently computable filter responses. The trained detector output differs from the "teacher" detector only at a small, controllable fraction of locations and yet is significantly faster.

Center for Machine Perception, Dept. of Cybernetics, Faculty of Elec. Eng. Czech Technical University in Prague, Karlovo nám. 13, 121 35 Prague, Czech Rep. e-mail: \{matas,sochmj1\}@cmp.felk.cvut.cz

RANSAC (**RAN**dom **SA**mple **C**onsensus) is a robust estimator that has been used in many computer vision algorithms e.g. for short and wide baseline stereo matching and structure and motion estimation. In the time-optimal RANSAC, we derive the fastest randomised strategy for hypothesis verification satisfying a constraint on the probability that the returned solution is correct. The optimal strategy is found again with the help of Wald's SPRT test.

The rest of the paper is structured as follows. First, we formally define the time-constrained detection problem and present the relevant parts of Wald's theory in Section 5.2. Next, the WaldBoost algorithm for sequential decision making is presented in Section 5.3. A face detector trained by the WaldBoost procedure is presented in Section 5.4. A similar methodology is applied in Section 5.5 to the problem of fast approximation of a repeatable interest point detector. In Section 5.6, Wald's SPRT test is combined with RANSAC and a very fast method for robust estimation of geometric relations and model parameters in general is obtained.

5.2 The Two-class Sequential Decision-making Problem

Let x be an object belonging to one of two classes $\{-1,+1\}$, and let an ordering on the set of measurements $\{x_1,\ldots,x_m\}$ on x be given. A sequential decision strategy is a set of decision functions $S=\{S_1,\ldots,S_m\}$, where $S_i:\{x_1,\ldots,x_i\}\rightarrow\{-1,+1,\sharp\}$. The strategy S takes the measurements one at a time and at time i makes a decision based on S_i. The '$\sharp$' sign stands for a "continue" (do not decide yet) decision[1]. If a decision is '$\sharp$', x_{i+1} is measured and S_{i+1} is evaluated. Otherwise, the output of S is the class returned by S_i.

In other words, a sequential strategy takes one measurement at a time. After the i-th measurement, it either terminates by classifying the object to one of the classes $+1$ or -1, or continues by taking the next measurement.

In two-class classification problems, errors of two kinds can be made by strategy S. Let us denote by α_S the probability of error of the first kind (x belongs to $+1$ but is classified as -1) and by β_S the probability of error of the second kind (x belongs to -1 but is classified as $+1$).

A sequential strategy S is characterised by its error rates α_S and β_S and its average evaluation time

$$\bar{T}_S = E(T_S(x)) \,, \tag{5.1}$$

where the expectation E is over $p(x)$ and $T_S(x)$ is the expected evaluation time (or time-to-decision) for strategy

$$T_S(x) = \arg\min_i (S_i(x) \neq \sharp) \,. \tag{5.2}$$

[1] In pattern recognition, this is called "the rejection option"

An optimal strategy for the sequential decision making problem is then defined as

$$S^* = \arg\min_S \bar{T}_S \tag{5.3}$$

$$\text{s.t.} \quad \beta_S \leq \beta \ , \tag{5.4}$$

$$\alpha_S \leq \alpha \tag{5.5}$$

for specified α and β.

The sequential decision-making theory was developed by Wald [23], who proved that the solution of the optimisation problem (5.3) is the *Sequential Probability Ratio Test (SPRT)*.

5.2.1 Sequential Probability Ratio Test

Let x be an object characterised by its hidden state (class) $y \in \{-1,+1\}$. This hidden state is not observable and has to be determined based on successive measurements $x_1, x_2, \ldots$. Let the joint conditional density $p(x_1,\ldots,x_m|y=c)$ of the measurements $x_1,\ldots,x_m$ be known for $c \in \{-1,+1\}$ and for all m.

SPRT is a sequential strategy S^*, which is defined as:

$$S^*_m = \begin{cases} +1, & R_m \leq B \\ -1, & R_m \geq A \\ \sharp, & B < R_m < A \end{cases} \tag{5.6}$$

where R_m is the likelihood ratio

$$R_m = \frac{p(x_1,\ldots,x_m|y=-1)}{p(x_1,\ldots,x_m|y=+1)} \ . \tag{5.7}$$

The constants A and B are set according to the required error of the first kind α and error of the second kind β. Optimal A and B are difficult to compute in practice, but tight bounds are easily derived.

Theorem 5.1 (Wald). *A is upper bounded by $(1-\beta)/\alpha$ and B is lower bounded by $\beta/(1-\alpha)$.*

Proof. For each sample $\{x_1,\ldots,x_m\}$, for which SPRT returns the class -1 we get from (5.6)

$$p(x_1,\ldots,x_m|y=-1) \geq A \cdot p(x_1,\ldots,x_m|y=+1) \ . \tag{5.8}$$

Since this holds for all samples classified to the class -1

$$P\{S^*=-1|y=-1\} \geq A \cdot P\{S^*=-1|y=+1\} \ . \tag{5.9}$$

The term on the left is the probability of correct classification of an object from the class -1 and is therefore $1-\beta$. The term on the right is the probability of incorrect classification of an object to the class $+1$, and is equal to α. After this substitution and rearranging, we get the upper bound on A. Repeating this derivation with the samples classified by SPRT to the class $+1$ the lower bound on B is derived. □

In practical applications, Wald suggests to set the thresholds A and B to their upper and lower bound respectively

$$A' = \frac{1-\beta}{\alpha}, \qquad B' = \frac{\beta}{1-\alpha}\ . \tag{5.10}$$

The effect of this approximation on the test error rates was summarised by Wald in the following theorem.

Theorem 5.2 (Wald). *When A' and B' defined in (5.10) are used instead of the optimal A and B, the real error probabilities of the test change to α' and β' for which*

$$\alpha' + \beta' \leq \alpha + \beta\ . \tag{5.11}$$

Proof. From Theorem 5.1 it follows that

$$\frac{\alpha'}{1-\beta'} \leq \frac{1}{A'} = \frac{\alpha}{1-\beta}\ , \text{and} \tag{5.12}$$

$$\frac{\beta'}{1-\alpha'} \leq \frac{1}{B'} = \frac{\beta}{1-\alpha}\ . \tag{5.13}$$

Multiplying the first inequality by $(1-\beta')(1-\beta)$ and the second by $(1-\alpha')(1-\alpha)$ and summing both inequalities, the result follows. □

This result shows that at most one of the probabilities α and β can be increased and the other has to be decreased by the approximation.

Theorem 5.3 (Wald). *SPRT (with optimal A and B) is an optimal sequential test in a sense of the optimisation problem (5.3).*

Proof. The proof is complex. We refer interested reader to [23]. □

Wald analysed SPRT behaviour when the upper bound A' and B' is used instead of the optimal A and B. He showed that the effect on the speed of evaluation is negligible.

However, Wald did not consider the problem of optimal ordering of measurements, since in all of his applications the measurements were i.i.d and the order did not matter. Secondly, Wald was not concerned with the problem of estimating (5.7) from a training set, since in the i.i.d case

$$p(x_1,\ldots,x_m|y=c) = \prod_{i=1}^{m} p(x_i|y=c) \tag{5.14}$$

and thus R_m can be computed incrementally from a one dimensional probability density function.

5.3 WaldBoost

For dependent measurements, which is the case in many computer vision tasks, SPRT can still be used if the likelihood ratio can be estimated. However, that usually encompasses many-dimensional density estimation, which becomes infeasible even for a moderate number of measurements.

In [15], it was suggested to use the AdaBoost algorithm for measurement selection and ordering and we review the relevant results in this section. The section is structured as follows. First, the AdaBoost learning algorithm is reviewed in Section 5.3.1. In Section 5.3.2, an approximation for the likelihood ratio estimation is proposed for such (statistically dependent) measurements. The WaldBoost algorithm combining SPRT and AdaBoost is described in Section 5.3.3.

5.3.1 AdaBoost

The AdaBoost algorithm [13, 2][2] is a greedy learning algorithm. Given a labelled training set $\mathscr{T} = \{(x_1, y_1), \ldots, (x_l, y_l)\}$, where $y_i \in \{-1, +1\}$, and a class of weak classifiers $\mathscr{H}$, the AdaBoost produces a classifier of the form

$$H_T(x) = \sum_{t=1}^{T} h^{(t)}(x) \,, \tag{5.15}$$

where $h^{(t)} \in \mathscr{H}$ and usually $T \ll |\mathscr{H}|$. Weak classifiers can be of an arbitrary complexity but are often chosen to be very simple. The final classifier then boosts their performance by combining them into a strong classifier H_T.

The outputs of selected weak classifiers will be taken as measurements used in SPRT.

In AdaBoost training, an upper bound on the training error is minimised instead of the error itself. The upper bound has an exponential form

$$J(H_T) = \sum_i e^{-y_i H_T(x_i)} = \sum_i e^{-y_i \sum_{t=1}^{T} h^{(t)}(x_i)} \,. \tag{5.16}$$

Training of the strong classifier runs in a loop. One weak classifier is selected and added to the sum at each loop cycle. A selected weak classifier is the one which minimises the exponential loss function (5.16)

[2] The real valued version is used.

$$h_{T+1} = \arg\min_h J(H_T + h)\,, \tag{5.17}$$

It has been shown [13, 3] that the weak classifier minimising (5.17) is

$$h_{T+1} = \frac{1}{2}\log\frac{P(y=+1|x,w^{(T)}(x,y))}{P(y=-1|x,w^{(T)}(x,y))}\,, \tag{5.18}$$

where $w^{(T)}(x,y) = e^{-yH_T(x)}$ is a weight of a sample (x,y) at cycle T.

As shown in [3], choosing a weak classifier according to (5.18) in each cycle of the AdaBoost learning converges asymptotically to

$$\lim_{T\to\infty} H_T(x) = \tilde{H}(x) = \frac{1}{2}\log\frac{P(y=+1|x)}{P(y=-1|x)}\,. \tag{5.19}$$

This result will be used in the following section.

5.3.2 Likelihood Ratio Estimation with AdaBoost

The likelihood ratio (5.7) computed on the outputs of weak classifiers found by AdaBoost has the form

$$R_t(x) = \frac{p(h^{(1)}(x),...,h^{(t)}(x)|y=-1)}{p(h^{(1)}(x),...,h^{(t)}(x)|y=+1)}\,, \tag{5.20}$$

where the outputs of the weak classifiers cannot be treated as statistically independent.

Since the computation of $R_t(x)$ involves a high dimensional density estimation, it is approximated so that this task simplifies to a one dimensional likelihood ratio estimation. The t-dimensional space is projected into a one dimensional space by the strong classifier function H_t (see equation (5.15)). Hence, all points $(h^{(1)},...,h^{(t)})$ are projected to a value given by the sum of their individual coordinates. Using this projection, the ratio (5.20) is estimated by

$$\hat{R}_t(x) = \frac{p(H_t(x)|y=-1)}{p(H_t(x)|y=+1)}\,. \tag{5.21}$$

Justification of this approximation can be seen from equation (5.19) which can be reformulated using Bayes formula to the form

$$\tilde{H}(x) = -\frac{1}{2}\log R(x) + \frac{1}{2}\log\frac{P(+1)}{P(-1)}\,. \tag{5.22}$$

Thus, in an asymptotic case, the strong classifier is related directly to the likelihood ratio. In particular, it maps all points with the same likelihood ratio to the same value. Consequently, it makes sense to estimate the likelihood ratio for every value

Algorithm 1 WaldBoost Learning

Input: $(x_1, y_1), ..., (x_l, y_l)$; $x_i \in \mathcal{X}, y_i \in \{-1, 1\}$,
desired final false negative rate α and false
positive rate β.

Initialise weights $w_1(x_i, y_i) = 1/l$
Set $A = (1-\beta)/\alpha$ and $B = \beta/(1-\alpha)$
For $t = 1, ..., T$

1. Choose h_t according to equation (5.18),
2. Estimate the likelihood ratio R_t according to Eq. (5.21)
3. Find thresholds $\theta_A^{(t)}$ and $\theta_B^{(t)}$
4. Throw away samples from training set for which $H_t \geq \theta_B^{(t)}$ or $H_t \leq \theta_A^{(t)}$
5. Sample new data into the training set

end
Output: Strong classifier H_T and thresholds $\theta_A^{(t)}$ and $\theta_B^{(t)}$.

of $\tilde{H}(x)$ and the estimate (5.21) is then exactly equal to $R(x)$. For a non-asymptotic case we take an assumption that the same relation holds between $H_t(x)$ and $\hat{R}_t(x)$ as well.

Several methods can be used to estimate $\hat{R}_t(x)$, like logistic regression for direct ratio estimation or the class densities can be estimated instead and the ratio can be calculated based on these density estimates. The method used in our implementation is described in Section 5.3.6.

Having the likelihood ratio estimate $\hat{R}_t$, the SPRT can be applied directly. Assuming monotonicity of the likelihood ratio, only two thresholds are needed on H_t values. These two thresholds $\theta_A^{(t)}$ and $\theta_B^{(t)}$, each one corresponding to one of the conditions in (5.6), are determined uniquely by the bounds A and B.

5.3.3 The WaldBoost Algorithm

The analysis given above allows us to define the WaldBoost algorithm. The WaldBoost learning phase is summarised in Algorithm 1 and described in Section 5.3.4. A WaldBoost classifier evaluation is explained in next Section 5.3.5 and summarised in Algorithm 2. Finally, a discussion of the algorithm details is given in Section 5.3.6.

5.3.4 Learning

WaldBoost requires, in addition to the usual AdaBoost initialisation by a labelled training set, two additional parameters specifying desired final false negative rate

Algorithm 2 WaldBoost Classification

Given: H_T, $\theta_A^{(t)}$, $\theta_B^{(t)}$, γ.
Input: a classified object x.
For $t = 1, \ldots, T$ *(SPRT execution)*
  If $H_t(x) \geq \theta_B^{(t)}$, classify x to the class $+1$ and terminate
  If $H_t(x) \leq \theta_A^{(t)}$, classify x to the class -1 and terminate
end
If $H_T(x) > \gamma$, classify x as $+1$. Classify x as -1 otherwise.

α and false positive rate β of the output classifier. These rates are used to compute the two thresholds A and B according to equation (5.10). The training runs in a loop, where the first step is a standard AdaBoost search for the best weak classifier (Step 1), as described in Section 5.3.1. Then, the likelihood ratio is estimated (Step 2) and the thresholds $\theta_A^{(t)}$ and $\theta_B^{(t)}$ are found (Step 3), as described in Section 5.3.2. Based on the thresholds, the training set is pruned (Step 4). Finally, a new training set is created by a random sampling over the samples, which have not been decided yet (Step 5). The steps 4 and 5 are discussed in more detail below.

Pruning of the training set (Step 4) is necessary to keep the final false negative and false positive rate under the specified values α and β. SPRT requires the likelihood ratio R_m to be estimated only over the samples which have passed undecided through all pruning steps up to the current learning cycle. The samples already classified as positive or negative class samples are removed from the training set.

For the **new data collection** (Step 5), a random sampling is performed over those data samples, which have not been assigned to any class yet. The number of newly sampled samples depends on the previous pruning step.

These two steps are similar to the bootstrapping technique [17] except that the samples are not collected only but thrown away in Step 4 as well. Another close approach is the cascade building procedure [22] with the substantial difference that the pruning and new data collection in the WaldBoost learning are run after every weak classifier is trained.

5.3.5 Classification

The structure of the WaldBoost classifier is summarised in Algorithm 2. The classification executes the SPRT test on the trained strong classifier H_T with thresholds $\theta_A^{(t)}$ and $\theta_B^{(t)}$. If H_t exceeds the respective threshold, a decision is made. Otherwise, next weak classifier is taken. If a decision is not made within T cycles, the input is classified by thresholding H_T on a value γ specified by the user.

5.3.6 Algorithm Details

Two parts of WaldBoost have not been fully specified. First, the exact likelihood ratio $R_t(x)$ is not know. Only its approximation $\hat{R}_t$ is used. Although this estimate is approaching the correct value with onward training, wrong and irreversible decisions can be made easily in early evaluation cycles. Hence, an inaccurate likelihood ratio estimation can affect performance of the whole classifier.

To reduce this effect, we estimate the likelihood ratio in the following way. The densities $p(H_t(x)|y=+1)$ and $p(H_t(x)|y=-1)$ are estimated not from the training set directly, but from an independent validation set to get an unbiased estimate. Moreover, the estimation uses the Parzen windows technique with the kernel width set according to the *oversmoothing rule* for the Normal kernel [14]

$$h_{OS} = 1.144\sigma n^{-1/5}, \tag{5.23}$$

where σ is the sample standard deviation and n the number of samples. The h_{OS} is an upper bound on an optimal kernel width and thus, the density estimate is smoother than necessary for an optimal density estimation. Due to this conservative strategy, the evaluation time can be prolonged but the danger of wrong and irreversible decisions is reduced.

Second important aspect of the WaldBoost learning is the stopping criterion. For practical reasons, only limited number of weak classifiers is found, which implies truncation of the sequential test during strong classifier evaluation. Wald [23] studies the effect of truncation of the sequential test procedure, however, his derivations hold only for cases where independent identically distributed measurements are taken. For that case, he suggests to threshold the final likelihood ratio at zero and analyses the effect of such method on the false negative and false positive rates of the test.

In our implementation, the final threshold is left unspecified. It can be used to regulate a false positive and a false negative rate in the application. It is also used in a ROC curve generation in the experiment section.

Generally, the more training cycles are allowed, the more precise is the likelihood ratio estimation and the better is the separation of the classes, but the slower is the classifier evaluation. For an analysis of the effect of truncation on WaldBoost performance see Section 5.4.1.

5.4 WaldBoost Applied to Face Detection

The WaldBoost algorithm is applicable to any time-constrained classification task. In this section, we show how to apply WaldBoost to face detection. The face detection problem has two specific features: (i) highly unbalanced class sizes and complexities, and (ii) particular requirements on error of the first and the second kind.

The object class size (the face class in our case) is usually relatively small and compact compared to the non-object class. The object class samples are difficult to collect and too much pruning can reduce the size of the object training set irreversibly. The non-object class, on the other hand, consists of all images except the images of an object itself. Such a huge and complex class cannot be represented by a small training set sufficiently. So, the goal of the learning is to explore the largest possible subspace of the non-object class while keeping most of the object samples during the learning process.

The second specific of the object detection is that error of the first kind (missed object) is considered as more serious than error of the second kind (falsely detected object). An ideal way of training a classifier would be to require a zero false negative rate and the smallest possible false positive rate.

Having the above specifics in mind, WaldBoost can be initialised in the following way. Let the required false positive rate β is set to zero and the required false negative rate α to some small constant (note the inverse initialisation compared to the above reasoning). In this setting, equations (5.10) reduce to

$$A = \frac{1-0}{\alpha} = \frac{1}{\alpha}\,, \qquad B = \frac{0}{1-\alpha} = 0 \tag{5.24}$$

and the SPRT strategy (5.6) becomes

$$S_m^* = \begin{cases} +1, & R_m \leq 0 \\ -1, & R_m \geq 1/\alpha \\ \sharp, & 0 < R_m < 1/\alpha \end{cases} \tag{5.25}$$

Since R_m is always positive, the algorithm will never classify a sample to the object class. The only allowed decision is the classification to the non-object class. Hence, the learning process will never prune the object part of the training set while pruning the non-object part. Such initialisation thus leads to an exploration of the non-object class (by pruning and new sample collection) while working with a small and unchanging object training set. Moreover, the detection rate of the final classifier is assured to be $1-\alpha$ while the false positive rate is progressively reduced by each training cycle.

5.4.1 Experiments

The proposed WaldBoost algorithm was tested on the frontal face detection problem. The classifier was trained on 6350 face images divided into a training and a validation set. In each training cycle, the non-face part of the training and the validation set included 5000 non-face samples sampled randomly from a pool of subwindows from more than 3000 non-face images. The weak classifier set $\mathscr{H}$ used in training is the same as in [22] but WaldBoost is not feature-specific and any other weak classifiers can be used. Unlike [22], the weak classifiers are real valued (de-

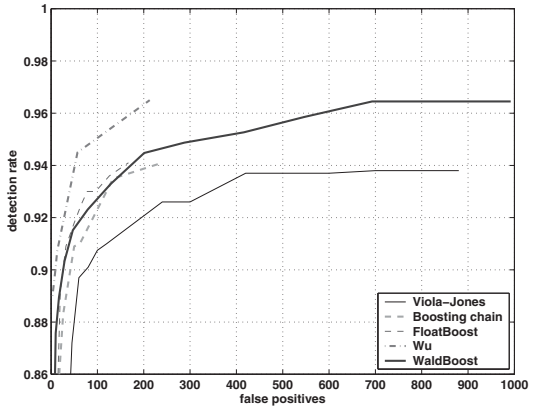


Fig. 5.1 ROC curve comparison of the WaldBoost algorithm with the state-of-the-art methods.

fined by equation (5.18)) and implemented as in [6]. The allowed false negative rate α was set to $5 \cdot 10^{-4}$. The training was run with $T = 600$, i.e. till the strong classifier consisted of 600 weak classifiers.

The WaldBoost classifier was tested on the MIT+CMU dataset [12] consisting of 130 images containing 507 labelled faces. A direct comparison with the methods reported in literature is difficult since they use different subsets of this dataset with the most difficult faces removed (about 5% in [6, 25]!). Nevertheless, we tested the WaldBoost classifier on both full and reduced test sets with similar results, so we report the results on the full dataset and plot them in one graph with the other methods (see Figure 5.1). However, the results of the other methods are not necessarily mutually comparable.

The speed and the error rates of a WaldBoost classifier are influenced by the classifier length. To examine this effect, four classifiers of different lengths (300, 400, 500 and 600 weak classifiers) were compared. The average evaluation time $\bar{T}_S$ (for definition see (5.1)) for these four classifiers is reported in Table 5.1. As expected, the average evaluation time decreases when less weak classifiers are used. However, shortening of the classifier affects the detection rates as well. The ROC curves for the four classifiers are depicted in Figure 5.2. Detection rates are comparable for the classifiers consisting of 400, 500 and 600 weak classifiers but the detection rate drops significantly when only 300 weak classifiers are used. Thus, using the classifier consisting of 400 weak classifiers only may be preferred for its faster evaluation. However, further reducing the classifier length leads to a substantial detection results degradation.

For a comparison of the WaldBoost classifier length with the other methods see Table 5.2. From the compared methods, the WaldBoost classifier needs the least number of weak classifiers, or in other words it produces the most compact classifier.

The bottom row of Table 5.2 shows the average evaluation times to decision $\bar{T}_S$ (sometimes referred to as the average number of weak classifiers evaluated) for the

#wc	600	500	400	300
$\bar{T}_S$	13.92	12.46	10.84	9.57

Table 5.1 Speed for different length WaldBoost classifiers.

Method	WB	VJ[22]	Li[6]	Xiao[25]	Wu[24]
#wc	400	4297	2546	700	756
$\bar{T}_S$	10.84	8	(18.9)	18.1	N/A

Table 5.2 The number of weak classifiers used and a speed comparison with the state-of-the-art methods. The parentheses around $\bar{T}_S$ of Li's method indicate that this result was not reported by the authors but in [25].

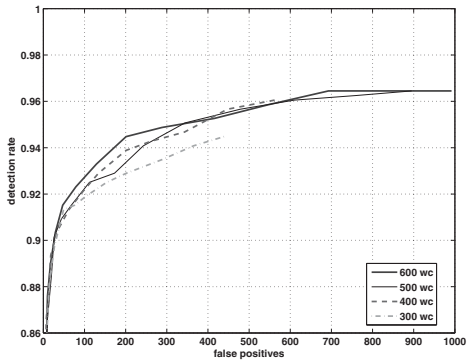


Fig. 5.2 The effect of reducing the number of weak classifiers in WaldBoost classifier on the detection rate.

compared methods. The WaldBoost learning results in the fastest classifier among the compared methods except for the Viola-Jones method which, despite its high speed, gains significantly worse detection results.

To conclude the experiments, the WaldBoost algorithm applied to the face detection problem proved its near optimality in the number of measurements needed for a reliable classification. The detection rates reached by the proposed algorithm are comparable to the state-of-the-art methods. The only method outperforming the proposed algorithm in the quality of detection is the "nesting-structured cascade" approach by Wu [24]. This can be caused by different features used, different subset of the MIT+CMU dataset used or any other implementation details.

5.5 WaldBoost Trained Fast Interest Region Detection

Learning a sequential classifier implementing a face detector as described in Section 5.3 can be viewed as a process of a fast minimum error approximation of the face detector. If suitable computation elements are available, many other binary functions can be approximated in the same way.

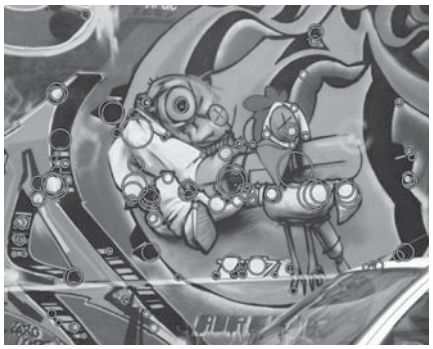


Fig. 5.3 The strongest responses of Mikolajczyk Hessian-Laplace region detector (threshold 3500).

Fig. 5.4 The strongest responses of WaldBoost Hessian-Laplace region detector (same number of responses as in Figure 5.3). $\bar{T}_S = 2.07$.

This section shows how to train a sequential detector approximating the behaviour of an interest region detector – the non-affine Hessian-Laplace detector of Mikolajczyk [9]. The detector has been shown to be widely useful in applications like wide baseline matching or image retrieval.

5.5.1 Hessian-Laplace WaldBoost Classifier

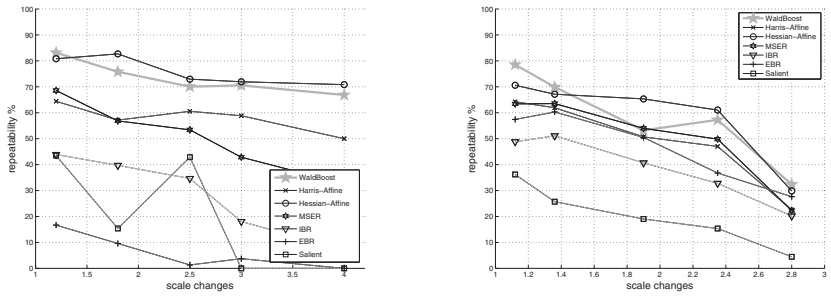
In the task of Hessian-Laplace region detector[3] approximation the positive examples correspond to the regions found by the detector and all other regions in the image are used as negative examples. The approximation error is determined by the agreement of the detectors outputs.

An example of output of a non-affine Hessian-Laplace detector is shown in the Figure 5.3. Only the strongest responses corresponding to threshold 3500 are shown. Training a WaldBoost classifier approximating the behaviour of the Hessian-Laplace detector entails several important differences in the training settings compared to the face detection.

First, positive examples are collected as an output of *rotationally invariant* Hessian-Laplace detector. To mimic this quality, the training set has to include rotated versions of positive examples. Nevertheless, the Haar-like filters are inherently axis parallel and thus the final rotation invariance will be weakened.

An important property of the interest regions detection task is that the *positive examples are very easy to collect*. Running the original detector on any image gives a new set of positive examples. The situation is similar to the problem of very huge negative examples set in the face detection problem. To process as many positive

[3] Available from `http://www.robots.ox.ac.uk/~vgg/research/affine/`



(a) (b)

Fig. 5.5 Repeatability score of the WaldBoost detector compared to the state-of-the-art methods for **(a)** Bark sequence **(b)** Boat sequence. Overlap 40%, norm. size = 30 pixels.

examples during training as possible, the positive examples set can be bootstrapped as well (i.e. β is set to a non-zero value).

Another difference is that there are *no images without positive samples* for negative examples collection by random sampling. Negative examples are not taken from an indifference region in the vicinity of a positive sample.

Finally, *the positive and negative classes are highly overlapping*. The interest region detectors have usually one parameter regulating the amount of returned positive samples. However, the more regions are returned, the less reliable the regions are and thus changing this parameter, the difficulty of the training set is controlled.

Another consequence of positive and negative class overlapping is that the WaldBoost classifier will give responses on many positions and scales. One way of removing the less trusty detections is to threshold the final classifier response. However, a better option is to set α to a higher value and let the training to concentrate on the most representative part of the positive class. This leads to much faster classifier, since the less trusty detections are decided earlier and do not need full classifier evaluation.

5.5.2 Experiments

The Hessian-Laplace detector has been approximated by a sequential WaldBoost classifier consisting of 20 weak classifiers (see [16] for more details). The strongest responses of the WaldBoost classifier are shown in Figure 5.4. Note that the detections are not exactly the same as the strongest responses of the Hessian-Laplace detector (Figure 5.3). The WaldBoost training does not retain the quality ordering of the detections of the original detector. Nevertheless, similar and sometimes the same structures are detected.

To evaluate the detector, the same tests as in [10] have been run to test the repeatability rate. The result on Bark and Boat sequences is shown in Figure 5.5. The

sequences contain scale and rotation changes. The repeatability of the WaldBoost detector is similar to the *affine* version of the Hessian-Laplace detector. The important property of the trained detector is its speed of 2 weak classifiers evaluated per window on average. The WaldBoost detector thus gains the speed of the original manually tuned algorithm with 1.3s per 850×680 image.

5.6 Robust Estimation of Model Parameters - RANSAC with Optimal Sequential Verification

RANSAC (**RAN**dom **SA**mple **C**onsensus) is a widely used robust estimator that has become a de facto standard in the field of computer vision. RANSAC has been applied to many vision problems: short baseline stereo [20, 19], wide baseline stereo matching, motion segmentation [20], mosaicing, detection of geometric primitives , robust eigenimage matching, structure and motion estimation [11, 18], object recognition and elsewhere.

In this section, we show how RANSAC speed can be improved by application of Wald's theory. We first briefly review a model verification strategy for RANSAC based on Wald's SPRT test. The resulting method [8] finds, like RANSAC, a solution that is optimal with user-specified probability. The solution is found in time that is (i) close to the shortest possible and (ii) superior to any deterministic verification strategy.

The RANSAC algorithm proceeds as follows. Repeatedly, subsets of the input data (e.g. a set of tentative correspondences) are randomly selected and model parameters fitting the sample are computed. In a second step, the quality of the parameters is evaluated on the input data. Different cost functions have been proposed [21], the standard being the number of inliers, i.e. the number of data points consistent with the model. The process is terminated when the probability of finding a better model becomes lower than a user-specified probability η_0. The $1-\eta_0$ confidence in the solution holds for all levels of contamination of the input data, i.e. for any number of outliers within the input data.

The speed of standard RANSAC depends on two factors: the number of random samples and the number N of the input data points. In all common settings where RANSAC is applied, almost all models whose quality is verified are incorrect with arbitrary parameters originating from contaminated samples. Such models are consistent with only a small number of the data points.

A provably fastest model verification strategy is designed for the (theoretical) situation when the contamination of data by outliers is known. In this case, the algorithm is the fastest possible (on average) of all randomised RANSAC algorithms guaranteeing a given confidence in the solution. The derivation of the optimality property is based on Wald's theory of sequential decision making, in particular a modified sequential probability ratio test (SPRT). In application, the requirement of a priori knowledge of the fraction of outliers is unrealistic and the quantity must be estimated online.

The speed of RANSAC depends on two factors. First, the percentage of outliers determines the number of random samples needed to guarantee the a given confidence in the optimality of the solution. Second, the time needed to assess the quality of a hypothesised model parameters is proportional to the number N of input data points. The total running time t of RANSAC can be expressed as

$$t = k(t_M + \overline{m}_S\, t_V)\ , \tag{5.26}$$

where k is the number of samples drawn, t_M is time needed to instantiate a model hypotheses given a sample, $\overline{m}_S$ is an average number of models per sample and t_V is average time needed to evaluate the quality of the sample. We choose the time needed to verify a single correspondence as the unit of time for t_M, t_V and t. Note that in standard RANSAC $t_V = N$.

The core idea of the Randomised (hypothesis evaluation) RANSAC, i.e. RANSAC with sequential hypothesis testing, is that most evaluated model hypotheses are influenced by outliers. To reject such erroneous models, it is sufficient to perform a statistical test on only a small number of data points. The test can be formulated as follows. The hypothesis generation step proposes a model. It is either 'good', i.e. it is uncontaminated with outliers and leads to the optimal solution (the solution with maximal support), or it is 'bad' (or contaminated), i.e. at least one of the data points in the sample is an outlier The property 'good' is a hidden state that is not directly observable but is statistically linked to observable events. The observable events are "data point (correspondence) is/is-not consistent with the model".

The statistical test has two effects on RANSAC behaviour: it (i) reduces the number of verified correspondences (and thus time complexity of the verification step) and (ii) introduces the possibility of rejecting (overlooking) a good sample. The probability α of rejecting a good sample is the significance of the test and it increases the number of samples drawn before the $1-\eta_0$ confidence is ensured. The correct model parameters are recovered if an uncontaminated sample is drawn and passes the test. This happens with probability

$$P = P_g(1-\alpha)\ .$$

The problem is to find a test that balances the number of correspondences needed for model verification and the increase in the number of samples induced by false rejections so that the total running time t Eq. (5.26) is minimised. Since the average time to draw an uncontaminated model that passes the test is $\bar{k} = 1/(P_g(1-\alpha))$, we have

$$t = \frac{1}{P_g(1-\alpha)}(t_M + \overline{m}_S\, t_V)\ . \tag{5.27}$$

5.6.1 The Optimal Sequential Test

In sequential testing, as applied e.g. in industrial inspection, the problem is to decide whether a model (or the batch of products) is 'good' or 'bad' in the shortest possible time (i.e. making the smallest number of observations) and yet satisfying the predefined bounds on the probabilities of the two possible errors – accepting a 'bad' model as 'good' and vice versa. Wald's SPRT test is a solution of this *constrained optimisation* problem. The user supplies the acceptable probabilities of the errors of the first and the second kind and the resulting optimal test is a trade-off between time to decision (or cost of observations) and the errors committed.

However, when evaluating RANSAC, the situation is different. First of all, a 'good' model is always evaluated for all data points (correspondences) since the number of inliers is one of the outputs of the algorithms. So the only error that can be committed is an early rejection of a 'good' model (error of the first kind). But this only means that more samples have to be drawn to achieve the required confidence $1-\eta_0$ of finding the optimal solution. So unlike in the classical setting, we are solving a *global optimisation* problem, minimising a single real number – the time to decision, since the consequence of an error is also a loss of time.

The model evaluation step of the optimal R-RANSAC proceeds as Wald's sequential probability ratio test (SPRT) with the probability α of rejecting a 'good' sample set to achieve maximum speed of the whole RANSAC process.

In the model evaluation step, our objective is to decide between the hypothesis H_g that model is 'good' and the alternative hypothesis H_b that the model is 'bad'. A 'good' model is computed from an all-inlier sample. The Wald's SPRT is based on the likelihood ratio [23]

$$\lambda_j = \prod_{r=1}^{j} \frac{p(x_r|H_b)}{p(x_r|H_g)} = \lambda_{j-1} \cdot \frac{p(x_j|H_b)}{p(x_j|H_g)} \,, \tag{5.28}$$

a ratio of two conditional probabilities of the observation x_r under the assumptions of H_g and H_b respectively. Not that here, unlike in the case of face and interest point detection, observations are independent since we are sampling at random and the product rule applies. In RANSAC, x_r is equal to 1 if the r-th data point is consistent with a model with parameters θ and 0 otherwise. For example, a correspondence is consistent with (i.e. supporting) an epipolar geometry represented by a fundamental matrix F if its Sampson's error is smaller than some predefined threshold [4]. The probability $p(1|H_g)$ that any randomly chosen data point is consistent with a 'good' model is approximated by the fraction of inliers ε among the data points[4]. The probability of a data point being consistent with a 'bad' model is modelled as a probability of a random event with Bernoulli distribution with parameter δ: $p(1|H_b) = \delta$. The process of estimation of δ and ε is discussed in Section 5.6.2.

[4] The probability ε would be exact if the data points were selected with replacement. Since the objective of the verification is to count the size of the support of the model, the correspondences are drawn without replacement. However, the approximation is close.

Algorithm 3 The Adapted Sequential Probability Ratio Test (Adapted SPRT).

Output: model accepted/rejected, number of tested data points j, a fraction of data points consistent with the model

Set $j = 1$
1. Check whether j-th data point is consistent with the model
2. Compute the likelihood ratio λ_j Eq. (5.28)
3. If $\lambda_j > A$, decide the model is 'bad' (model "rejected"), else increment j
4. If $j > N$, where N is the number of correspondences, decide model "accepted" else go to Step 1.

Algorithm 4 The Structure of R-RANSAC with SPRT.

Initialise ε_0, δ_0, calculate A_0 and set $i = 0$.

Repeat until the probability η of finding a model with support larger than $\hat{\varepsilon}$ falls under a user defined value η_0:

1. Hypothesis generation
- Select a random sample of minimum size m from the set of data points.
- Estimate model parameters θ fitting the sample.

2. Verification

1mm=3pt Execute the SPRT (Alg. 3) and update the estimates if

a. Model rejected: re-estimate δ. If the estimate $\hat{\delta}$ differs from δ_i by more than 5% design (i+1)-th test ($\varepsilon_{i+1} = \varepsilon_i$, $\delta_{i+1} = \hat{\delta}$, $i = i+1$)

b. Model accepted and the largest support so far: design (i+1)-th test ($\varepsilon_{i+1} = \hat{\varepsilon}$, $\delta_{i+1} = \hat{\delta}$, $i = i+1$). Store the current model parameters θ.

After each observation the standard Wald's SPRT makes one of three decisions: accept a 'good' model, reject a 'bad' model, or continue testing. Since in RANSAC the total number of inliers is needed to decide on termination, nothing is gained by an early decision in favour of a 'good' model. Therefore the option of an early acceptance of the model has been removed in the Adapted SPRT (Alg. 3). The full SPRT is described e.g. in Wald [23] and, in a more accessible form, in Lee [5].

The Optimal Value of the Decision Threshold The decision threshold A is the only parameter of the Adapted SPRT. In [8], Chum and Matas show show how to set it to achieve optimal performance. The total expected time of RANSAC is expressed as a function of A: The average time to the solution expressed as a function of A is

$$t(A) = \frac{1}{P_g(1-1/A)} \left(t_M + \overline{m}_S \frac{\log A}{\mathbb{E}\left(\log \frac{p(x|H_b)}{p(x|H_g)}\right)} \right) . \tag{5.29}$$

The minimum of $t(A)$ is found iteratively by process with fast convergence.

The R-RANSAC with SPRT algorithm is outlined in Alg. 4. To fully specify details of the algorithm, two issues have to be addressed. First, the estimation of parameters δ and ε; second, the termination criterion guaranteeing $1-\eta_0$ confidence in the solution has to be derived.

Algorithm 4 proceeds like standard RANSAC [1, 4], only instead of checking all data points in the model verification step, the data points are evaluated sequentially and hypotheses with low support are rejected early. After a hypothesis is rejected, δ is re-estimated (Alg. 4, step 2a). Accepted hypotheses are candidates for the RANSAC outcome (see below). The overhead of the evaluation of the likelihood ratio λ_j Eq. (5.28) is negligible compared to the evaluation of the model versus data point error function.

5.6.2 Estimation of δ and ε

The optimal test derived in Section 5.6.1 requires the knowledge of two parameters, ε and δ. These probabilities are different for different data sets and we assume they are unknown. The proposed algorithm uses values of ε and δ that are estimated during the sampling process and the test is adjusted to reflect the current estimates.

If the probabilities ε and δ are available a-priori, e.g. in some standard setting where the algorithm is run repeatedly, they can be used in the initialisation of the algorithm.

Estimation of δ. Since almost all tested models are 'bad'[5], the probability δ can be estimated as the average fraction of consistent data points in rejected models. When current estimate δ differs from the estimate used to design the SPRT (by more than 5%, for example), new (i+1)-th test is designed. The initial estimate δ_0 is obtained by geometric considerations, i.e. as a fraction of the area that supports a hypothesised model (a strip around an epipolar line in case of epipolar geometry) to the area of possible appearance of outlier data (the area of the search window). Alternatively, a few models can be evaluated without applying SPRT in order to obtain an initial estimate of δ.

Estimation of ε. In general, it is not possible to obtain an unbiased estimate of ε, since this would require the knowledge of the solution to the optimisation problem we are solving. The tightest lower bound on ε is provided by the size of the largest support so far. It was shown in [7] that a sample with the largest support so far appears $\log k$ times, where k is the number of samples drawn. When such a sample (with support of size I_{i+1}) appears, new test is designed for $\varepsilon_{i+1} = I_{i+1}/N$. Throughout the course of the algorithm, a series of different tests with

$$\varepsilon_0 < \cdots < \varepsilon_i < \cdots < \varepsilon$$

are performed. The initial value of ε_0 can be derived from the maximum time the user is willing to wait for the algorithm to terminate.

The properties of R-RANSAC with SPRT were tested on a wide range of standard data and a two to tenfold speed up of the algorithm was observed [8]. Tests

[5] RANSAC verifies, on average, $-\log(\eta_0)$ 'good' models, e.g. for the typical $\eta_0 = 0.05$ a 'good' model is hypothesised three times prior to termination of the algorithm.

included epipolar geometry estimation in both wide and narrow baseline settings and homography estimation.

5.7 Conclusions

A framework exploiting Wald's sequential analysis for designing time-efficient two-class detection and matching algorithms was presented. Besides Wald's Sequential Probability Ratio Test, we relied on WaldBoost, a method that allows learning sequential classifiers in the case of non-i.i.d. features.

The WaldBoost algorithm was applied to the problems of face and interest point detection. Error rates of the face detector proposed algorithm were comparable to the state-of-the-art methods. In the interest point application, the output of the Hessian-Laplace detector [9] was approximated by a sequential WaldBoost classifier consisting of 20 weak classifiers. The detector was evaluated according to the standard testing protocol on reference images [10] and its repeatability was similar to the *affine* version of the Hessian-Laplace detector. The WaldBoost detector gains the speed of the original manually tuned Hessian-Laplace algorithm — only about 2 weak classifiers are evaluated per window on average, which means that about eight additions are needed on average to decide a window corresponds to an interest point. Finally, we have presented a sequential strategy based on Wald's SPRT for evaluation of model quality in RANSAC. The resulting RANSAC with SPRT is significantly faster (2 to 10 times) than its deterministic counterpart.

Acknowledgements The authors were supported by EC projects FP6-IST-004176 COSPAL (JM), FP6-IST-027113 eTRIMS (JŠ) and by the Grant Agency of the Czech Republic project 201/06/1821.

References

1. Fischler, M., Bolles, R.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. CACM **24**(6), 381–395 (1981)
2. Freund, Y., Schapire, R.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences **55**(1), 119–139 (1997)
3. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. Tech. rep., Department of Statistics, Sequoia Hall, Stanford Univerity (1998)
4. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision, 2nd edn. Cambridge University, Cambridge (2003)
5. Lee, P.M.: Sequential probability ratio test. University of York. www.york.ac.uk/depts/maths/teaching/pml/ais/sprt.ps
6. Li, S., Zhu, L., Zhang, Z., Blake, A., Zhang, H., Shum, H.: Statistical learning of multi-view face detection. In: ECCV, p. IV: 67 ff. (2002)
7. Matas, J., Chum, O.: Randomized RANSAC with $T_{d,d}$ test. Image and Vision Computing **22**(10), 837–842 (2004)