# On Testing Image Processing Applications With Statistical Methods

Johannes Mayer

Abteilung Angewandte Informationsverarbeitung
Universität Ulm
D–89069 Ulm
mayer@mathematik.uni-ulm.de

**Abstract:** Testing image processing applications is very ressource consuming. Many complex images have to be generated as test inputs and the expected resulting outputs have to be determined to complete the test cases. The present paper deals with this challenge in testing implementations of image operations, namely dilation. It applies random testing using models from stochastic geometry for random input generation. The Statistical Oracle, a modification of the well-known Heuristic resp. Parametric Oracle, is used to compare the results. Therefore, reliability predictions are also possible.

## 1 Introduction

What is the major problem in testing applications for image processing? Non-trivial images, i. e. the test inputs, are not easy to produce. Furthermore, it takes time to figure out the expected result. Therefore, this is a very ressource consuming task. And there is still an important question after the tests have been performed: What do these tests say about the reliability of the application? Coverage criteria are not suitable to answer this question.

How to overcome these problems? Random testing, i. e. testing with randomly generated inputs, can easily be applied to generate a large number test cases, i. e. complex images. According to [DN84, Fr98, Fr99, HT90], random testing is effective. Random testing increases the chance of finding bugs [Ho01]. Whereas the random generation of test inputs is simple, the corresponding expected results are usually not obvious. This is the well-known test oracle problem. A test oracle is responsible for the decision, whether a test case passes or not. If no expected results are provided, which can be compared to the actual results, more complex oracles are needed.

The present paper shows how random testing can be used to test applications from image processing. Therefore, it presents a model from stochastic geometry and a general solution of the test oracle problem in the case that explicit formulae for the mean, the distribution, and so on, of characteristics computable from the test results are available. A Statistical Oracle using statistical methods is described, being a special case of the Heuristic Oracle [Ho99] resp. Parametric Oracle [Bi00]. The Statistical Oracle is based on statistical tests.

Therefore, it is also possible to make preditions on the reliability. However, as with the Heuristic Oracle resp. the Parametric Oracle, the decision of a Statistical Oracle is not always correct. This error probability can be specified explicitely.

The following section contains a review of some related work on random testing. Thereafter, the Statistical Oracle is described in general. Then, the necessary statistical methods are presented to implement the components of the Statistical Oracle. Finally, the application of this oracle to the testing of the implementation an image processing operator, namely dilation, is described, followed by a conclusion and perspectives.

## 2 Related Work

Random testing, i. e. testing with randomly generated inputs, is a well-known and efficient method [Ag78, DN84, Fr98, Fr99, HT90, Ha94, Sc79]. It requires test oracles to ensure the adequate evaluation of test results.

Most oracles, such as Solved Example Oracle, Simulation Oracle, Gold Standard Oracle, Reversing Oracle, Generated Implementation Oracle, and Different But Equivalent Oracle (all described in [Bi00]) check the actual output for correctness. For the above oracles, the expected results are specified by hand, generated through a simplified, prototype, or gold standard implementation, verified using simple computations, produced by an implementation generated from a formal specification, or generated from equivalent executions of the IUT, respectively.

The oracles so far exactly compared the expected outputs with the actual. However, this is not always feasible. Therefore, oracles exist that only verify some properties of the actual output of the test. The Built-in Test Oracle [Bi00] requires the IUT to have some self-checking mechanisms integrated, such as assertions verifying constraints. Similarly, for complex software, checking of the actual outputs can be done exactly for simple inputs or approximately—concerning constraints, accuracy, or inconsistencies—as described in [We82]. Finally, the Heuristic Oracle [Ho99] resp. the Parametric Oracle [Bi00] extracts some parameters from the actual outputs and compares them with the expected values. Hoffman [Ho99] as well as Binder [Bi00] mention the use of statistical parameters such as the mean and the variance. However, they do not detail how the comparison is to be performed—which is essential in this case. The present papers contains an application of the Statistical Oracle [MG04]—a special case of the Parametric Oracle resp. the Heuristic Oracle—giving also necessary implementation details especially for the comparison.

## 3 Statistical Oracle

The Statistical Oracle [MG04] verifies some statistical characteristics of the actual test results—and can therefore be applied in random testing, provided that the mean, variance, distribution, and so on, of characteristics computable from the test results are known. It

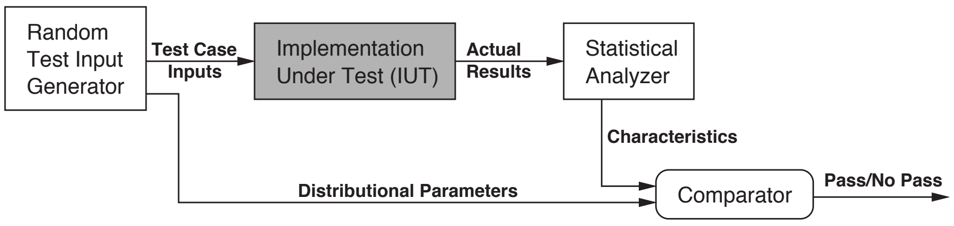

Figure 1: Statistical Oracle for random testing

makes sense in case the inputs are complex and the exact output cannot be verfied with moderate effort. It is a special case of the Heuristic Oracle [Ho99] resp. the Parametric Oracle [Bi00]. Here statistical characteristics are employed and compared using statistical methods, esp. statistical tests.

Figure 1 shows the principal structure of the Statistical Oracle which can only be used in random testing. It consists of the Statistical Analyzer and the Comparator.

- The Statistical Analyzer computes various characteristics from the test output that may be modeled as random variables. These characteristics are then sent to the Comparator.
- The Comparator computes the empirical sample mean and the empirical sample variance of its inputs.
- The Comparator receives the distributional parameters from the Random Test Input Generator. Therefore, the Random Test Input Generator must be prepared to deliver the distributional parameters to the Comparator. Furthermore, expected values and properties of the characteristics are computed by the Comparator (based on the distributional parameters of the random test input).

Using statistical tests, the reliability can be specified. The decision of a Statistical Oracle is not always correct—in contrast to usual oracles. However, the probability of a false pass can be adjusted as will be shown.

An important consequence of the Statistical Oracle is that is cannot decide whether a single test case passes or not. It can only make this decision for a couple of test cases. If a failure occurs, no single test case can be identified that detected the bug—it is the couple of test cases as a whole.

The Statistical Oracle does not check the actual output but only some characteristics of it. Therefore, as explained in [Bi00, Ho99], it does not suffice to perform all tests with a Statistical resp. Parametric resp. Heuristic Oracle, since these tests only focus on the observed characteristics. Therefore, other test cases and oracles are also necessary.

## 4 On the Implementation of the Statistical Analyzer and the Comparator

For an application of the Statistical Oracle, the implementations of the Comparator and the Statistical Analyzer have to be detailed. The Statistical Analyzer outputs characteristics—some for each test case. The computation of these characteristics is usually based on estimators. The implementation of the Statistical Analyzer, therefore, depends on the IUT and cannot be generalized. The Comparator collects the outputs of the Statistical Analyzer for $n$ test cases and computes the sample mean and the sample variance. Thereafter, it decides based upon statistical tests. The Comparator allows for generalization. In the following, some statistical basics are explained that are necessary for the implementation of the Comparator. More details on the necessary statistics are provided e. g. by [CB02].

Let $X_1, \ldots, X_n$ denote the random variables that model the inputs of the Comparator for a single characteristic, where $X_i$ belongs to the $i$ th test case. Since the individual test runs are completely independent of each other, the random variables $X_i$ are independent and identically distributed, say with mean $\mu$ and variance $\sigma^2$. Both, $\mu$ and $\sigma^2$, are unknown, since they depend on the IUT which is to be tested.

The sample mean of these random variables $X_1, \ldots, X_n$ is

$$\overline{X_n} := \frac{1}{n} \sum_{i=1}^{n} X_i.$$

According to the central limit theorem, it holds that

$$\frac{\overline{X_n} - \mu}{\sigma/\sqrt{n}} \xrightarrow{d} \mathcal{N}(0,1)$$

for $n \to \infty$. Thus, for practical purposes, $\overline{X_n}$ can be regarded as approximately normally distributed with mean $\mu$ and variance $\sigma^2/n$ if $n \geq 30$ (a common rule of thumb). The greater $n$ gets, the less likely deviations from $\mu$ become (which is also known as the weak law of large numbers).

Additionally, the sample variance

$$S_n^2 := \frac{1}{n-1} \sum_{i=1}^{n} (X_i - \overline{X_n})^2$$

of the random variables $X_1, \ldots, X_n$, that approaches $\sigma^2$ as $n$ goes to infinity, will also be necessary for the statistical tests.

So far, only random variables have been considered. In case of a concrete test run, $x_i$, $\overline{x_n}$, and $s_n^2$ denote the respective realizations of these random variables.

In the following $\mu_0$, denotes the mean that the random variables $X_i$ are expected to have. (The input generator is required to deliver the distributional properties to the Comparator.

Based on them, $\mu_0$ can be computed by the Comparator.) The following approach is used to decide whether the actual mean $\mu$ equals $\mu_0$ or not.

A statistical hypothesis test is to be employed to check, whether the mean is equal to the expected value. However, it is not that simple. It seems to be obvious to use the t-test. However, the null hypothesis of this test states that the mean is equal to the expected value. This hypothesis thus states that the IUT is correct in that respect. So, a Type I error is in this case that the test does not pass whereas the IUT is correct (at least regarding this aspect). This is not the error whose error probability should be controlled. It would be preferable if the probability that the IUT passes whereas it is buggy, could be chosen arbitrarily. It is however not possible to simply exchange the null hypothesis and the alternative hypothesis.

Using an intersection-union test [CB02] that combines two one sided t-tests, this problem can be overcome. Let $\varepsilon > 0$ be chosen arbitrarily to define an environment around the mean, the null hypothesis can be stated as

$$\mu \notin [\mu_0 - \varepsilon, \mu_0 + \varepsilon].$$

The probability $\alpha \in (0, \frac{1}{2})$ for a Type I error, i. e. that the IUT passes though the IUT is not correct, can be chosen arbitrarily.

Then, if

$$\frac{\overline{x_n} - (\mu_0 - \varepsilon)}{s_n/\sqrt{n}} \geq t_{n-1,\alpha/2} \qquad \text{and} \qquad \frac{\overline{x_n} - (\mu_0 + \varepsilon)}{s_n/\sqrt{n}} \leq -t_{n-1,\alpha/2}$$

hold, the null hypothesis is rejected and thus the implementation passes. $t_{n-1,\alpha/2}$ denotes the $(1 - \alpha/2)$-quantile of the t-distribution with $n - 1$ degrees of freedom.

The probability of a Type II error, i. e. that the IUT does not pass whereas there is no error regarding the tested aspect, cannot be chosen arbitrarily. However, it is important that the probability of this error is not too high, since in this case time is wasted for searching a non-existent bug. The probability of this error can however, given a fixed value for $\alpha$, be decreased by increasing the sample size $n$.

## 5 Testing Image Processing Applications

Now that the foundations have been laid through the Statistical Oracle, this section describes how to apply random testing using the Statistical Oracle to test implementations of dilation. First this operator is introduced. Thereafter, testing will be adressed.

### 5.1 Preliminaries

A more detailed introduction to the following preliminaries can be found e. g. in [So03].

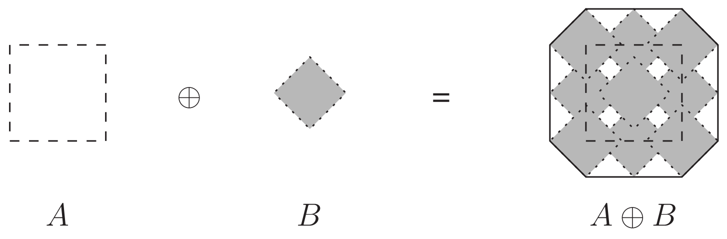

Figure 2: Illustration of the Minkowski Addition

**Sets and Dilation** Let $A$ and $B$ be subsets of $\mathbb{R}^2$. $\check{A}$ denotes the *reflection* of $A$ at the origin. The *Minkowski addition* $A \oplus B$ is defined as

$$A \oplus B := \{x + y : x \in A, y \in B\}.$$

The set $B$ is in this case called the *structuring element.* Figure 2 illustrates the Minkowski addition. $\delta_B(A)$ denotes the *dilation* of $A$ with the structuring element $B$ and is defined as

$$\delta_B(A) := A \oplus \check{B}.$$

Obviously, dilation and Minkowski addition are equivalent if the structuring element is symmetric (with respect to the origin).

**Images** A *digital image* can be seen as a function

$$I : \mathcal{D}_I \subseteq \mathbb{Z}^2 \to \mathcal{R}_I$$

mapping each pixel of the domain $\mathcal{D}_I$ onto the range $\mathcal{R}_I$. Very important are *binary images* where $\mathcal{R}_I = \{0, 1\}$.

A binary image can also be seen as the digital version of a set where the value of a pixel indicates whether this pixel belongs to the set (value 1) or not (value 0). Therefore, dilation can easily be interpreted as a transform mapping a binary image onto another binary image (given a structuring element).

### 5.2 Testing an Implementation of Dilation for Binary Images

Now, it will be described how to apply random testing with the Statistical Oracle to test an implementation of dilation for binary images.

**Random Test Input Generator** Very important in random testing is the random input. This is simple for numbers, more complex for strings. But what about images? How to

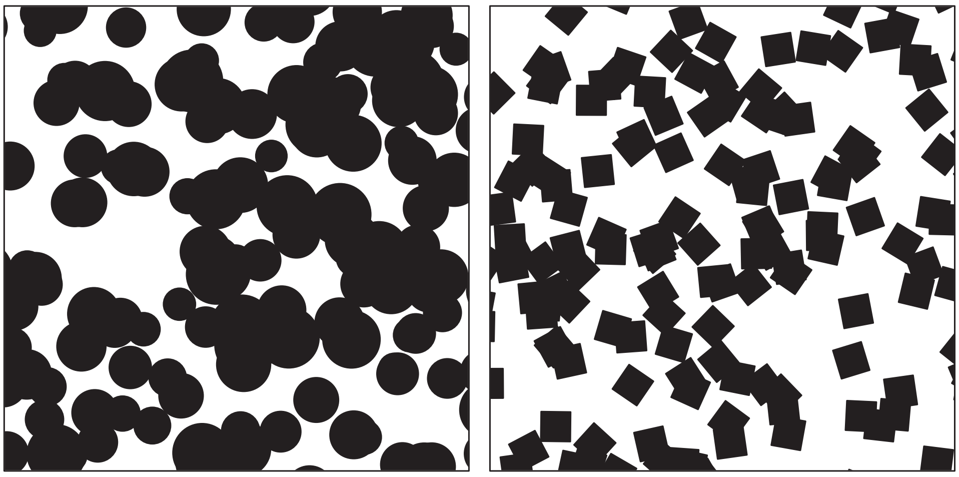
Figure 3: Possible realizations of the Boolean model

randomly generate images and at the same time have the possibility to implement an oracle for these input images?

Here models from stochastic geometry can be used, specifically the Boolean model (cf. e. g. [Mo97]). Figure 3 shows two possible realizations of the Boolean model. The Boolean model is simply the union $\bigcup_i (B_i + X_i)$ of i. i. d. random grains $B_i$ (such as discs with random radius or squares with random rotation) each translated into another point $X_i$ of the underlying Poisson process. A detailed introduction to the Boolean model is given in e. g. [Mo97]. See e. g. [MSS04] for the simulation of the Boolean model.

**Considered Characteristics** It is necessary to decide which characteristics of the output, i. e. the dilated image, can be used by the Statistical Oracle. If the image is a realization of the Boolean model, it can be represented by the set

$$\bigcup_i (B_i + X_i).$$

Let $B$ be the structuring element used in the dilation. Then, the result of the dilation is the set

$$\begin{aligned} \delta_B\left(\bigcup_i (B_i + X_i)\right) &= \left(\bigcup_i B_i + X_i\right) \oplus \check{B} \\ = \bigcup_i ((B_i + X_i) \oplus \check{B}) &= \bigcup_i (B_i \oplus \check{B}) + X_i, \end{aligned}$$

using some properties of Minkowski addition. Thus, the result is the Boolean model with grains $B_i \oplus \check{B}$. For example, if $B_i$ is a disc with random radius $R_i$ and $B$ is a disc with radius $r$, $B_i \oplus \check{B}$ is simply a disc with random radius $R_i + r$. Similar properties hold if $B_i$ and $B$ are both squares or rectangles (without rotation). The first important fact is, thus, that dilation transforms a realization of the Boolean model into a realization of the Boolean model (with different grains).

The Boolean model has been studied for a long time. As a result, explicit formulae for specific area $A_A$, boundary length $L_A$, and Euler number $\chi_A$ are known (see e. g. [Mo97]):

$$\begin{aligned} A_A &= 1 - \exp(-\lambda\overline{A}) \\ L_A &= \lambda\overline{L}\exp(-\lambda\overline{A}) \\ \chi_A &= \lambda\left(1 - \frac{\lambda\overline{L}^2}{4\pi}\right)\exp(-\lambda\overline{A}) \end{aligned}$$

where $\lambda$ is the intensity of the Poisson process $\{X_1, X_2, \ldots\}$. $\overline{A}$ and $\overline{L}$ are the mean area and boundary length of the so-called *primary grain* $B_1$.

Using methods from [OM00] or [KSS04, SS05], the specific area, boundary length, and Euler number of the underlying random set—the Boolean model—can be estimated from a binary image without bias.

**Putting the Pieces together** Random testing of dilation of binary images can be done as follows (Figure 1 gives an overview):

1. The Random Test Input Generator generates realizations of the Boolean model. These realizations are sets. Thereafter, these sets are transformed into binary images. These are the test case inputs delivered to the IUT (together with a structuring element $B$). Furthermore, the Random Test Input Generator passes the intensity $\lambda$ of the Poisson process as well as $\overline{A}$ and $\overline{L}$ of the dilated primary grain $B_1 \oplus B$ to the Comparator.

2. The IUT computes the resulting image and passes it to the Statistical Analyzer.

3. The Statistical Analyzer computes the estimators for $A_A$, $L_A$ and $\chi_A$. Each such estimator can be modeled as a random variable $X_i$ (c. f. Section 4). Then $A_A$, $L_A$ and $\chi_A$ are the expected means, respectively, of these random variables (i. e. $\mu_0$)—due to the unbiasedness of the estimators. It passes each realization $x_i$ to the Comparator—for each of these random variables.

4. The Comparator accumulates the realizations $x_i$ of each such random variable $X_i$ for $n$ outputs and computes the realization $\overline{x_i}$ of the sample mean. Finally, it decides using a statistical test as described in Section 4 whether the IUT passes or not. (The error probability $\alpha$ can also be given to the Comparator.)

One has to be careful to choose only such estimators that are unbiased. Otherwise $A_A$ and so on would not be the expected means of the estimator.

As mentioned in Section 4, the sample size $n$ has to be at least $30$ to guarantee approximate normal distribution of the mean and it should be chosen much bigger to reduce the probability of a Type II error, i. e. a false alarm.

## 6 Conclusion and Perspectives

The present paper dealt with random testing of image processing applications, specifically implementations of dilation. Therefore, the Statistical Oracle has been applied with a statistical test in conjunction with samples from the Boolean model as random input. It has been shown that in this case, the output is also a sample from the Boolean model (with other parameters). Choosing specific area, boundary length, and Euler number as parameters computed by the Statistical Analyser has proven beneficial. Theoretical formulae are known for these characteristics for the Boolean model. The presented combination fits perfectly for the purpose. The input $\alpha \in (0, \frac{1}{2})$ which is the probability of a false pass can be chosen arbitrarily. Thus, reliability can be controlled. Notice, however, that the presented test only checks some characteristics of the output and only uses a subset of all possible inputs. Therefore, the reliability is only with respect to this class of inputs and with respect to the specific area, boundary length, and Euler number of the output. For other images, the implementation may behave completely different—at least in theory. And the output may be wrong despite area and so on are correct, which is not very likely. To increase the types of inputs, the test should be executed with different settings of the parameters of the Boolean model (different intensity of the Poisson process and different grains). Furthermore, at least special inputs (such as inputs with all pixels set to $0$ or $1$) should be tested in addition. To keep the Type II error small, the sample size $n$ has to be increased.

As described in [MG04], the Statistical Oracle can be used whenever characteristics from the output are known. The application in the field of image processing is only one possibility.

Erosion, another image transform, can be expressed in terms of dilation and complement. Therefore, this test could also be used for erosion. Furthermore, the implementation of dilation is usually based on distance transform and threshold. For this reason, it should also be possible to adapt this test for distance transform.

## References

[Ag78] Agrawal, V. D.: When to Use Random Testing. IEEE Transactions on Computers **27**, 1978; pp. 1054–1055.

[Bi00] Binder, R. V.: Testing Object-Oriented Systems: Models, Patterns, and Tools. Addison-Wesley, 2000.

[CB02] Casella, G.; Berger, R. L.: Statistical Inference. Wadsworth Group, Duxbury, CA, USA, 2002.

[DN84] Duran, J. W.; Ntafos, S. C.: An Evaluation of Random Testing. IEEE Transactions on Software Engineering **10**, 1984; pp. 438–444.

[Fr98] Frankl, P. G. et al.: Evaluating Testing Methods by Delivered Reliability. IEEE Transactions on Software Engineering **24**, 1998; pp. 586–601.

[Fr99] Frankl, P. G. et al.: Correction to: Evaluating Testing Methods by Delivered Reliability. IEEE Transactions on Software Engineering **25**, 1999; p. 286.

[HT90] Hamlet, R. G.; Taylor, R.: Partition Testing Does Not Inspire Confidence. IEEE Transactions on Software Engineering **16**, 1990; pp. 1402–1411.

[Ha94] Hamlet, R. G.: Random testing. In (Marciniac, J. J. Ed.): Encyclopaedia of Software Engineering. John Wiley & Sons, 1994.

[Ho01] Hoffman, D.: Using Oracles in Test Automation. In: Proceedings of the Nineteenth Annual Pacific Northwest Software Quality Conference (PNSQC), Portland, Oregon, USA, 2001; pp. 90–117.

[Ho99] Hoffman, D.: Heuristic Test Oracles. Software Testing & Quality Engineering **1**, 1999; pp. 29–32.

[KSS04] Klenk, S.; Schmidt, V.; Spodarev, E.: A New Algorithmic Approach to the Computation of Minkowski Functionals for Polyconvex sets. Preprint, University of Ulm, 2004. (submitted)

[MG04] Mayer, J.; Guderlei, R.: Test Oracles Using Statistical Methods. In: Proceedings of the First International Workshop on Software Quality, Lecture Notes in Informatics **P-58**, Köllen Druck+Verlag GmbH, 2004; pp. 179–189.

[MSS04] Mayer, J.; Schmidt, V.; Schweiggert, F.: A Unified Simulation Framework for Spatial Stochastic Models. Simulation Modelling Practice and Theory **12**, 2004; pp. 307–326.

[Mo97] Molchanov, I.: Statistics of the Boolean Model for Practioners and Mathematicians. John Wiley & Sons, Chichester, 1997.

[OM00] Ohser, J.; Mücklich, F.: Statistical Analysis of Microstructures in Materials Science. John Wiley & Sons, Chichester, 2000.

[SS05] Schmidt, V.; Spodarev, E.: Joint Estimators for the Specific Intrinsic Volumes of Random Sets. Stochstic Processes and their Applications, 2005 (to appear)

[Sc79] Schneck, P. B.: Comment on "When to Use Random Testing". IEEE Transactions on Computers **28**, 1979; pp. 580–581.

[So03] Soille, P.: Morphological Image Analysis: Principles and Applications. Second Edition, Springer-Verlag, New York, 2003.

[We82] Weyuker, E. J.: On Testing Non-Testable Programs. Computer Journal **25**, 1982; pp. 465–470.