Annales UMCS Informatica AI 5 (2006) 401-406

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI

http://www.annales.umcs.lublin.pl/

Detecting changes in environment of mobile robot

Krzysztof Sapiecha[*], Arkadiusz Chrobot

Department of Computer Science, University of Technology in Kielce,
Al. 1000-lecia Państwa Polskiego25-314 Kielce, Poland

Abstract

The usefulness of four signature schemes for detecting changes in environment of mobile robot is investigated. Computational and memory complexities of software implementations of the schemes are experimentally compared. Finally, the best of the schemes for considered application is chosen.

1. Introduction

The necessity for detecting changes in images emerged during research on applying mobile robots (equipped with cameras) to investigate physical environment [1]. If the surroundings change quickly, a single mobile robot will be driven by human operator (remote control mode). The operator decides which path the robot will go in order to detect all changes in the environment in the fixed period of time [1,2]. In steady environment the robot does not have to be supervised. Instead it may move along a path established in advance by the operator (local control mode). In the fixed points of the path robot takes photos of surroundings. An analysis of the photos allows the robot for detecting changes in the surrounding space, which have happened since its last tour. The robot informs the operator if any change has been detected. The operator decides on further robot's actions.

There are three kinds of possible changes to be detected [1]:

– a new object or group of objects may appear,
– an object or group of objects may disappear,
– an object or group of objects may change its properties or location.

It is assumed that 10% of pixels in previous and up to date photos may change, at most. The robot decides whether to send an image to the operator or store it into memory. The decision is based on the results of analysis of taken photos. The algorithm for change detection must detect all changes, which are

[*]Corresponding author: *e-mail address*: e-mail: k.sapiecha@tu.kielce.pl

important. It should be also fast enough to assure the validity of control sequences of the robot. In other words, the robot should communicate with the operator if and only if it is necessary.

Signatures are one of the means, which could be used for change detection. Usually they are used for protecting integrity and authenticity of digitally transmitted messages. One of the main advantages of digital signatures is that they express large data object, like text files or images, in small amount of information (compression ratio about 10^6).

In the paper the usefulness of signature schemes for detecting changes in images taken by mobile robots is discussed. Motivation for this research is given in section 2. Section 3 includes descriptions of evaluated signature schemes. Computational results of the application of the schemes to images representative for the problem are presented in section 4. The paper ends with short conclusions.

2. Motivation

Changes in mobile robot environment are detected by comparing image that shows current state of surroundings with previously taken photos. Unfortunately, because processing capabilities, as well as memory capacity of the robot are limited, none of the conventional methods of images comparing can be applied to perform this task. These methods have also one additional drawback. Usually they are designed to detect specific kind of change [3]. The robot has to detect any kind of changes, but the information about the type of change is not important. Moreover, usually the ratio of changes is limited (in the case of our research it is less than 10% of the whole image). Therefore, in this case signature analysis could be used as an alternative for conventional methods.

Algorithm for generating signatures binds with each large data object, a unique (in most cases) signature of small size. These signatures could be stored in robot's memory and hence the consumption of this resource could be reduced. Unfortunately, sometimes such mapping is ambiguous. It is possible that signature collision may occur. In this case the same signature will be assigned to two different objects.

Signature calculation algorithm useful for detecting changes should meet at least three requirements. First, the signature collision ratio should be as small as possible. Next, the size of the signature obtained with such algorithm should be as small as possible, and it should be calculated in a relatively short time.

3. Signature schemes

There were four signature schemes evaluated. The first one comes from the Ronald Rivest MD4 (Message-Digest) algorithm [4]. It assigns a 128-bits long signature to any message of an arbitrary size. The algorithm consists of the following five steps:

1. A single "1" bit and some number of "0" bits are added to the message to make its length congruent to 448, modulo 512.
2. The length of original message (before it was extended) is written as 64 bit value and added to the "padded" message. Now, the length of the message is an exact multiple of 512.
3. Four 32-bit variables, named A, B, C, D are initialized as follows:
   – A = 01234567
   – B = 89abcdef
   – C = fedcba98
   – D = 76543210

   (all four numbers are in the hexadecimal code). These variables are used in the algorithm to compute the signature.
4. Each 512-bit part of the message and variables A,B,C and D are used to compute 128-bit signature. The computation is performed in three rounds. Each round uses one of three different procedures (FF, GG and HH) to compute partial result. These procedures are defined as follows:

```
FF(a,b,c,d,M[k],s):   a = (a+F(b,c,d) + M[k]) << s
GG(a,b,c,d,M[k],s):   a = (a+G(b,c,d)+M[k] + 5A827999) << s
HH(a,b,c,d,M[k],s):   a = (a+H(b,c,d)+M[k] + 6ED9EBA1) << s
```

where << denotes shift left operation, M[k] denotes the k-th 512-bit long part of the message and F,G,H are defined as follows:

```
F(X,Y,Z) = X*Y or not(X)*Z
G(X,Y,Z) = X*Y or X*Z or Y*Z
H(X,Y,Z) = X  xor Y xor Z
```

5. The result is 128-bit value stored in variables A, B, C, D. The least significant byte is stored in variable A, the most significant byte is stored in variable D.

   MD5 is a modification of MD4 [5]. The first three steps of the algorithm are the same as in MD4. In the fourth step of the algorithm four rounds are performed instead of three. There are also used four procedures (FF,HH,GG,II) which are defined as follows:

```
FF(a,b,c,d,M[k],s):   a =b+((a+F(b,c,d) + M[k]+T[i]) << s)
GG(a,b,c,d,M[k],s):   a = b+((a+G(b,c,d)+M[k] + T[i]) << s)
HH(a,b,c,d,M[k],s):   a = b+((a+H(b,c,d)+M[k]) + T[i]) << s)
II(a,b,c,d,M[k],s):   a=  b+((a+I(b,c,d)+M[k]) + T[i]) << s)
```

   where $\lfloor 2^{32} \cdot |\sin(i)| \rfloor$, F, H are defined as in MD4 and G,I are defined as follows:

G(X,Y,Z) = Y*Z or Y*not(Z)
I(X,Y,Z) = X xor (X or not(Z))

In this work MD5 and MD4 algorithms are applied to images. Therefore, a message here is a digital representation of an image.

The LFSR algorithm looks like an algorithm for cyclic code calculation, but it is byte-wise. The information, for which the signature is calculated, is successively divided by an indivisible polynomial, with the help of linear feedback shift register (LFSR), which means that each byte of the message is xored with bytes of LFSR determined by the polynomial. The result is stored in the most significant byte of LFSR, but after the content of LFSR is shifted right exactly one byte [6,7].

The last tested signature scheme is the algebraic signatures calculation one [8]. The information for which the signature is calculated is assumed to be composed of Galois Field elements and it is divided into fixed size pages. There is also a vector of primitive elements of Galois Field. This vector is called the *base of the signature*. A signature for each of the pages of the information is calculated according to the following formula:

$$sig_{\alpha}\left(P\right)=\sum_{i=0}^{l-1} p_i \cdot \alpha^i$$

where: l is the length of the page, α is the one of elements of base vector, p_i is the i-th element of the page

The length of the signature depends on the number of elements in the base vector. This procedure is repeated for pages made of signatures, until a single signature is obtained.

4. Computational results

The collision ratio, the time of execution and the usage of memory for all of the algorithms described in section 3 were calculated. The algorithms were implemented in software. A special program was written. It was compiled using C compiler form GNU Compilers Collection (version 3.1), with enabled optimization. The program was then run on the Athlon XP 2500+ based PC computer, with 512MB RAM, and the Mandrake Linux 9.0 operating system.

Test input data for calculation of the collision ratio consisted of two image sets. The images in the first set were splat into six different groups, according to their resolution and the ratio of changes. The changes were randomly generated with normal distribution, and included, consequently 0.005%, 0.01%, 0.03%, 0.05% and 0.1% of all pixels in image. The images in the second set were splat into four groups, each containing ten images. The split criterion was the same as previously, but the changes included a 1600 pixels big square region of the picture. The location of such a region was also randomly selected with the Gaussian distribution.

No collision was detected during the tests for any of evaluated signature schemes.

Computing time tests were made for six groups of images. Each of them contained six images of the same resolution. Figure 1 shows the results of the tests.

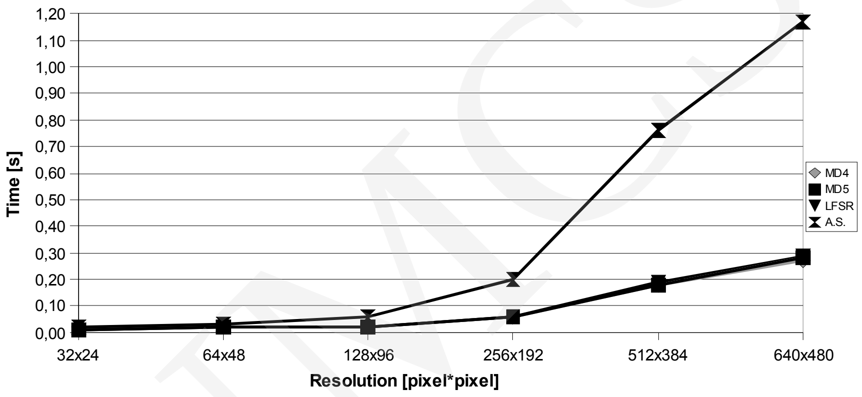

Fig. 1. Computing time tests

There is no meaningful difference in time of calculation for MD4, MD5 and LFSR signatures. Calculation of algebraic signatures takes the most of processor time.

Figure 2 shows memory usage for each of the evaluated signatures.

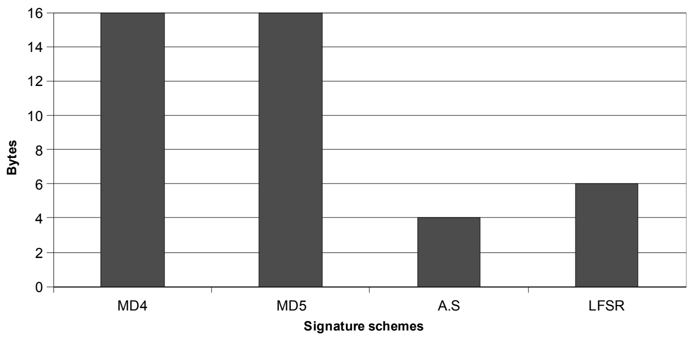

Fig. 2. Memory usage

In this case the best results are obtained with algebraic signatures and LFSR signatures. It should be stated that the size of these signatures depends consequently, on the size of base vector (in the case of algebraic signatures), and the size of indivisible polynomial (in the case of LFSR signatures). It should be also noticed, that the algebraic signature calculation algorithm needs a significant memory overhead for its partial results.

Figure 3 shows the results of both previously described tests put together. The result for LFSR algorithm is significantly better, compared to the other signature scheme results.

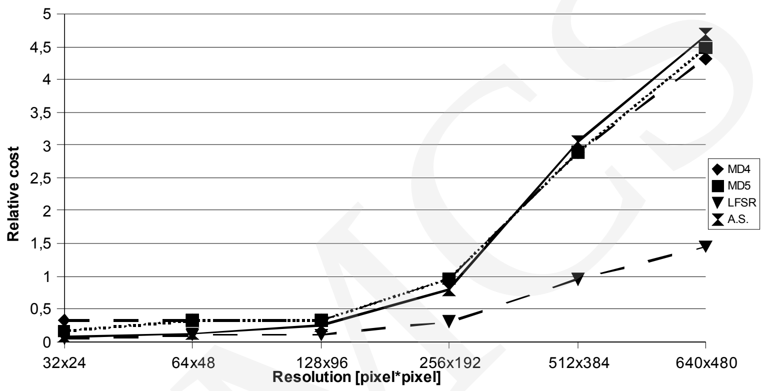

Fig. 3. Relative cost of signaturing (relative time * relative memory usage)

Conclusions

Respecting the test results, the LFSR signature scheme should be chosen to handle the described problem. It should be stated that test images were computer generated. Real world images contain noise, which should be removed, before the signature calculation process begins. However, this inconvenience is common for all algorithms of detecting changes in images. Hence, this does not deny the conclusion drawn.

The LFSR signature scheme can be easily implemented in hardware. Such LFSR is inexpensive and can meet even very strong time constraints.

References

[1] Sapiecha K., Łukawska B., Paduch P., *Experimental Data Driven Robot for Pattern Classification*, Annals UMCS, Informatica, (2005).
[2] Fekete S.P., Klein R., Nüchter A., *Online Searching with an Autonomous Robot*", http://arxiv.org/PS_cache/cs/pdf/0404/0404036.pdf
[3] Ballard D.H., Brown C.M., *Computer Vision*, Prentice-Hall Inc, New Jersey, (1982).
[4] Rivest R., *The MD4 Message-Digest Algorithm*, http://www.ietf.org/rfc/rfc1320.txt
[5] Rivest R., *The MD5Message-Digest Algorithm*, http://www.ietf.org/rfc/rfc1321.txt
[6] Sapiecha K., *Testowanie i diagnostyka systemów cyfrowych*", PWN, Warszawa, (1987), in Polish.
[7] Kutyłowski M., Strothmann W.B., *Kryptografia. Teoria i praktyka zabezpieczania systemów komputerowych*, Oficyna Wydawnicza Read Me, Warszawa, (1999).
[8] Litwin W., Schwarz T., *Algebraic signatures for Scalable Distributed Data Structures*, http://www.cse.scu.edu/~tschwarz/Papers/icde2004.pdf