Watermarking in Sensor Data Sets

Sebastian Wiendl
Betreuer: Corinna Schmitt
Seminar Future Internet SS2012
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: wiendl@in.tum.de

ABSTRACT
This paper will give a brief introduction into "Electronic Watermarking" by describing its origins in the 1950s and its gain of importance in the past 10 to 20 years and then especially handle the topic of "Watermarking in Sensor Data Sets" .
Electronic watermarking finds practical use in many business applications and is, next to cryptography, part of the security mechanisms against illegal redistribution and use of copyright-protected material. The traditional methods of electronic watermarking handle a great number of applications but struggle with unstructured data sets. Explaining a newly found method to cover that issue the second part of this paper goes into more detail by handling the technique of "Self-Identifying Sensor Data" as described in the paper [7]. Therefore this paper should be seen as a summary and comparison of the traditional methods with a new method in the field of electronic watermarking.

Keywords
Electronic watermarking, self-identifying sensor data

1. INTRODUCTION
Electronic watermarking is a method to implant a provenance mark into data. This is handle ownership and copyright issues. It can be tracked all the way back to the year 1954 in which the first patent had been filed. Since then a good amount of effort has been put into researching more sophisticated ways to watermark electronic products, nowadays digital products. Especially the past two decades stood out in the development because of the increasing need for electronic/digital watermarking. These days various techniques are being used by businesses and researchers are still working on newer and better methods. One of the newly found techniques is called "Self-Identifying Sensor Data" and covers a type of data that had previously not been able to be watermarked by the traditional techniques. In this paper, where past and future techniques are being compared, the reader will find some mathematical formulas as well as general overviews.

In section 2 the topic starts off with the definition of electronic watermarking and its development during the following years until now. After that the various types of commercial applications are shown. That section is then completed by a short summary before, in section 3, the approach of self-identifying sensor data is being discussed and shown in great detail. This part follows the more general part with mathematical formulas and algorithms. Finally, section 4 concludes with a comparison of the traditional approaches for electronic watermarking and the new technique of self-identifying sensor data.

2. ORIGIN AND DEVELOPMENT OF ELECTRONIC WATERMARKING
2.1 Definition and origin
2.1.1 Definition
Typically watermarks are known in a non-digital manner. Every Euro bill has watermarks.

Figure 1: watermark in a 10 Euro bill

The idea is to clearly mark the bills with a visible mark that cannot (easily) be recreated or removed illegally and that everyone can recognize in order to verify the authenticy of the bill. Different applications exist. Figure 1 shows the watermark used on the Euro bills. Figure 2 shows a screen-shot of electronic watermarking. Google Maps includes small watermarks to their maps which can only be (barely) seen (circled in red) when zoomed in all the way [9].

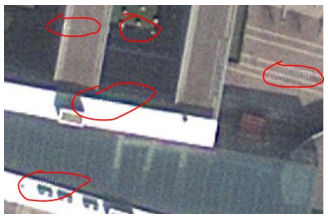

Figure 2: Screenshot from Google Maps on highest zoom factor [9]

Electronic watermarking, or nowadays rather digital watermarking commits to the same principles. More precisely it can be explained as:

doi: 10.2313/NET-2012-08-1_07

> "Digital watermarking is the process by which identifying data is woven into media content such as images, printed materials, movies, music or TV programming, giving those objects a unique, digital identity that can be used for a variety of valuable applications." [1]

A good watermark should not add (much) additional data to the unmarked data. This is to avoid the consequences of being too expensive and uneconomically.

2.1.2 Origin and progress

The first patent to ever handle the topic of electronic watermarking was filed in the year 1954. Emil Hembrooke from the Muzac Corporation called his work "Identification of sound and like signals" [12] and wrote that his method would allow to identify the origin of a piece of music and can therefore "to be likened to a watermark in paper" [4]. In the following 35 years several more patents have been filed. They were mostly revolving around the topic of music. The Lynch Carrier Systems Inc. designed a system to control telephony equipment and in a patent of the Musicast Inc. They used radio stations to distribute music to other businesses by adding a low frequency to the broadcast which would allow those businesses to remove advertisements [5]. All in all research was done and inventions were made but the big hype did not hit in yet. It was not until the 1990s that watermarking was put much more into focus. Reason for that was the Internet. With the extensive spread of the world wide web more and more people in the private sector gained access to the new media. The possibility of spreading information and data all over the world in almost no time gave opportunity to an exponential increase of illegal activities. Illegal sharing and selling of media with copyright protection led the music industry to immediate actions. Research had to find solutions and then, once again find more solutions when illegal distributors had found a new way to undermine their efforts. That race led to much progress in the topic of electronic watermarking.
The technology industry started groups like the Copy Protection Technical Working Group (CPTWG) (concerning digital video content) and the Strategic Digital Music Initiative (SDMI) (concerning digital music) to deal with those problems, too [4]. In general one can say that people have watermarking expected to be more sophisticated by now but it still provides several applications in commercial use.

2.2 Commercial applications

Watermarking can be used in a variety of ways to satisfy copyright and security aspects. The following applications should briefly explain and illustrate the use with several examples that are being used today.

2.2.1 Transaction tracking

In transaction tracking, also called fingerprinting, each copy work/device has an unique watermark embedded. That watermark is usually used to identify the origin of a copy. This can, for example, be used to track down the source of an illegally spread video. That way the origin of the distribution can be found and that person can be charged with the copyright issues. That technique is in use and widely spread. The DiVX corporation embedded that system into their DVD players So whenever a user creates many copies of a movie they can later be traced back to the initial player/user [8]. The problem with that technology is that collision attacks can easily undergo the system. The vulnerability lies in the big number of devices. If one person buys as few as about 20 copies of a single device an unmarked original can be created from the marked copies. That collusion attack would therefore allow to create watermark-free copies of DiVX DVD players if one had 20 original ones at hand. The effectiveness of fingerprinting for widely spread applications is therefore not perfect. Professional copiers will not be traced back but less talented users can safely be tracked back and held responsible. On the other hand transaction tracking can be well-applied for a small scale use. Pieces of work with a small availability - when it can be guaranteed that not many distributions will end up in the same wrong hands - will profit from the fingerprinting technology.[4] An example for a small scale application that led to success if the watermarking technique created by "Civolution". It "has been used successfully for identifying the source of illegal copies of the 2003 Academy Award screeners". [2]

2.2.2 Proof of ownership

The original idea when Muzak invented watermarking was to mark a piece of work in a way that can legally be used to prove ownership, even in a court trial. A major problem with that is that there is a wide variety of watermarks being applied. So how can a watermark be safely and for one hundred percent be connected with a single company/legal owner? The key to this is to not just embed a watermark independent of the original (unmarked) work, but create a cryptographic link between the original and the marked copy. S. Craver had that idea in 1996 and it is also technologically doable.[4] Actually, there is still a lot of research going on covering that topic. One research laboratory is "IBM Research". In one of their papers they describe the issue concerning invisible watermarks. [3]

2.2.3 Copy Control

Copy is the attempt to ensure that illegal pieces of work will not be created. Before the process of copying the device would check the specific piece of work for a copyright sign and then only copy if none was embedded. Taking that serious one would have to install watermark detectors in every recording and copying device. The device would then at first check the media for a watermark and only copy it if no watermark was found. Two main problems stand in the way of successfully guaranteeing a total copy control. At first it has to be ensured that truly everyone is able to detect the watermark and with that it would end up being only a weak security application. Some companies must still see an economical gain and apply copy control. Secondly, it makes sense that the whole idea only works perfectly if every company established a decoder in their devices. A couple of reasons strongly stand against that from the companies point of view: A detector is a piece of hardware (and software) that has to be added to every device. It will therefore add costs but no practical value to each device. Some people would then rather buy equipment that has no decoder in order to make their desired copies and this will lead to lower sales rates for the participating companies. The only way to ensure that every company joins the initiative would

doi: 10.2313/NET-2012-08-1_07

be to force them by "a combination of laws and contractual obligations." [4]

2.2.4 Authentication

For authentication purposes a digital watermark functioning as a signature can visibly be added to, e.g. an image, as seen in Figure 3.

Figure 3: Digital image with a watermark

This type of visible watermark is not the only way to use authentication. There are many ways to invisibly embed a digital signature. One approach for this method can be found in [6]

2.2.5 Legacy system enhancement & database linking

Watermarking can not only be used as in a security type of application as mostly described before. Record companies can embed a digital watermark in their audio files that devices of cooperating companies can record, filter and decode. That would, for example, give them the song title, the artists name and the album name. More modern applications are not necessarily depending on that. The application "Shazam"[10], which can be downloaded on PC, MAC and as well on mobile devices using the common markets for apps, has its own way to do that. As described in a paper released by the "Shazam"-company [11] they have created their own database containing self-created acoustic fingerprints. The basic idea is that the "fingerprint hash is calculated using audio samples near a corresponding point in time, so that distant events do not affect the hash" [11]. Basically once the database is created the same algorithm used to create the hash can be applied to any recorded song and then the hash created by the algorithm will point to the song in the database. Now song title, artist and more information is available about the previously unknown piece of music - without the use of watermarking. But one or more decades in the past there where not as sophisticated devices as there are available these days. That is when patents were filed and some companies still used watermarking for the distribution of music. As already stated in section 2.1.2 the Musicast Inc. used radio stations to distribute music to other businesses by adding a low frequency to the broadcast which would allow those businesses to remove advertisements. [5]

2.3 Summary and Conclusion

As seen in the previous chapters there is a wide variety of applications for electronic watermarking.
With the big hype in the 1990s various fields established themselves and there still is research and development of new methods going on. Some applications are slowly becoming redundant because of new inventions ("Shazam") and others are less safe than originally desired. Although only weak security can be provided by the techniques many companies still see an economical advantage in using them and will continue to do so. In many fields electronic watermarking is used as an addition to cryptographic methods. That combination is very popular and provides good services. All in all there are no reliable suggestions to make about the future development and use of electronic watermarking. As I.J Cox and M.L. Miller say: "If the past is a prediction to the future, then it is clear that watermarking technology will continue to be used in businesses."[4]

3. SELF-IDENTIFYING SENSOR DATA

For many fields of research a lot of datasets are needed and are not necessarily gathered by the researchers themselves. There are big companies / institutes that generate those datasets and offer them for research. Of course, they still have the copyrights on it and want to ensure that their work is not used without references or worse. That issue is being handled in the following.
After the previously handled more general approach and overview this chapter will go more into the details of one specific application of electronic watermarking. As described in the paper "Self-Identifying Sensor Data" [7] the basic idea and fields of application of a new way to watermark sensor data sets will be laid out and then explain the mathematical background behind the approach. After that a summary of the papers evaluation and analysis will explain the quality trade offs of the approach. Finally the whole concept will be looked at in matters of deployment issues and future work.

3.1 General approach

Transmitted sensor data will always carry some noise with it. That means that, e.g. a thermometer might tell you the temperature precisely in a scale of 10^{-3} degrees Celsius but the sensor could still deliver you the temperature up to an accuracy of 10^{-8} degrees. Therefore the last few digits (least significant digits) are noise and not of any (big) importance. Because of that the authors of the paper [7] suggest to embed a provenance mark into the data by replacing the noise. Three important categories will be used to rate the technique:

Perceptibility

Perceptibility in this case means that the embedded watermark should not (significantly) change the data set. If an embedded provenance mark changed the data radically then it would not fit the criteria. Of course, including a mark will change data, but - as previously stated - including a mark only into the parts that contain noise will have no effect on the true data itself.

Robustness

Robustness is a characteristic that allows the data set to counter several transformations without being changed in a way that makes the data useless. This method is robust against: truncation and quantization of digits, random bit flips, sampling and reordering of data within the sets. All those transformations can, of course, only be withstood up to a certain point. e.g. random bit flips on each bit of the data would change it into a totally new dataset, etc.

doi: 10.2313/NET-2012-08-1_07

Capacity
Capacity includes two factors:

One-bit watermarking
This allows to determine whether a data set is marked or not. The technique described in [7] uses two bits in each bit vector to save information. These *check bits* are the two most significant bits of all the least significant bits. The parameter check bit *(pc)* contains information about the parameters used for embedding the watermark and the mark check bit *(mc)* is a hash of the provenance mark and the significant bits. The more uncorrupted data points a set contains the more likely the two bits can be retrieved. Then one-bit checking can successfully be applied.

Blind watermarking
Blind watermarking allows one to extract an embedded watermark without having any knowledge of the specific mark. The approach to support that is as following:
In order to guarantee that an uncorrupted provenance mark can be found and extracted the provenance mark will be split in several pieces. Each bit vector will contain a piece (which specific one will depend on the significant bits of the vector). Overall each and every piece of the provenance mark will appear in several bit vectors at different bit sections (all within the least significant bits). This creates a lot of wanted redundancy. If one data point is corrupted (or even if all data points suffer from e.g. truncation) the provenance mark can still be extracted because the pieces appear multiple times at different locations. In order to put the provenance mark together after retrieval the least significant bits will be used to make an educated guess on the provenance mark. If the guess is wrong the next step would be to check various similar marks.

3.2 Detailed Description

3.2.1 Formal Problem Statement and Preconditions

Before diving straight into the details of the technique some mathematical foundation has to be stated:
A *data set* is a *list* of *vectors* where each vector is a bit vector representing a data point. *Transformations* will be seen as *functions* taking a list as an input and returning a list different from the input list. Here the two main types of functions will be encoding and retrieval functions. An *encoding* transforms a list to a coded list whereas a *retrieval function* will return the original list when getting the encoded list as an input. An *encoding-retrieval function pair* is robust when a retrieval of an encoded dataset, which had been corrupted, still returns the original dataset. Two final assumptions will help the process: The length of the provenance mark L_m will be known and additionally all data/bit vectors will represent positive integral data (integers) of a certain length.

The latter assumption leaves special cases to be handled separately:
All bit vectors that do not have the specified length can be changed by adding leading zeros. Other cases are:

Negative integers
Negative numbers can be transformed (and later also transformed back to their original representation) to signed integer bit vectors. Embedding a provenance mark will not have any effect on the sign-bit because only the least-significant bits will be changed.

Floating point numbers
Floating point numbers can be transformed to a decimal point representation and then multiplied by a power of ten (10^n) big enough that all bit vectors will finally only represent integers. After having embedded the watermark a multiplication with 10^{-n} will return the floating point representation. *Important:* If truncation has changed the length of a bit vector then it won't be 10^n; similar exponents should be tried then (e.g. 10^{n+1}, 10^{n+2}). Sure enough, one could say that there is no need to transform the bit vectors. It is possible to simply use transformation matrices *during* the encoding and decoding process. In this paper/approach that will not be done. A pre-formatting of the data will keep the calculations during the application simpler and easier to understand!

Low-entropy datasets
Data sets with only a small amount of distinct numbers provide the problem that many provenance pieces would be the same (as they are created from the significant bits which are equal/very similar in such data sets). The solution is to introduce smaller provenance pieces. Some of the least-significant bits can then be seen as significant bits used to create the provenance piece. This will deliver the wanted diversification.

3.2.2 Possible Transformations

The corruption model consists of several transformations than can change the data.
Rounding occurs when bit vectors are rounded to the closest multiple of an integer n. Similar to that it might happen that some of the least significant bits are being cut off. That is called *Truncation*. Many datasets might contain a high number of single data points but there is not always a need for such an extensive amount of single data points. During *Deletion/Sampling* data points are being removed from the dataset or a simple subset is being used. In an unstructured set an order can often not be found and then *Reordering* might happen to the set. The new set is then a permutation of the original set. Finally, a phenomenon that computer scientists know very well an occur randomly. A *Bit flip:* changes a number of bits in a bit vector. Bit flips are much less likely to appear than the other transformations. This is a big advantage because e.g. rounding only affects the least significant bits where-else random bit flips can also affect the important data contained in the more significant bits!

3.3 Embedding, Checking and Retrieving of a Provenance Mark

This section goes into the (mathematical) details of embedding, checking and retrieving of a provenance mark. At first it will lay out how provenance pieces are created and how they will be embedded into a bit vector. The next step is to explain how the check bits derive from the provenance

doi: 10.2313/NET-2012-08-1_07

pieces and the significant bits. After that the final big step is to retrieve a provenance mark and handle those cases in which corruption has occurred in the data set.

3.3.1 Embedding a provenance mark

Figure 4 shows how a bit vector will look like after the insertion of a provenance mark.

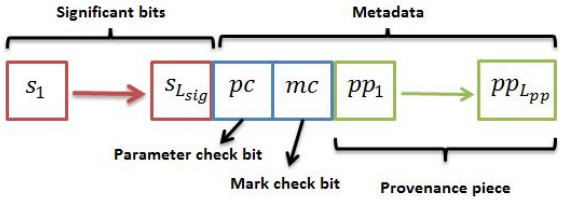

Figure 4: Bit vector with an embedded provenance piece [7]

The significant bits stay unchanged and only the least significant bits, also called metadata, will contain the various provenance pieces as well as the check bits.

Requirements and Terminology

- The number of the insignificant bits is called: L_{md}
- The number of the significant bits is called: L_{sig}
- The length of a data point therefore is: $L_{sig} + L_{md}$
- Having the two check bits, requires: $L_{md} \geq 3$

Provenance Pieces

If the length of a complete provenance mark is greater than L_{md} -2 then it will be split up in smaller pieces. Each piece contains a part of the whole provenance mark and in total there will be N_{pp} pieces. For each data point a *hash* function will take N_{pp} and the significant bits s as input and return a value k. That means that the specific data point will have the *kth* provenance piece embedded into it.

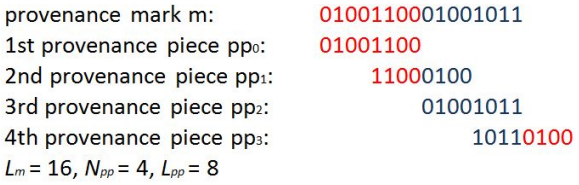

Figure 5: Provenance pieces example [7]

As you can see in Figure 5 the provenance pieces basically overlap. In the example from [7] the provenance mark technically could be split in only two different pieces that would cover all its data, already (pp_0 and pp_2). pp_1 and pp_3 cover redundant information. If, for example, rounding appears in the process then all pieces could, e.g. lose their last two bits. Without redundancy the data would be lost. But having pp_1 and pp_3 information that got lost in pp_0 will not be lost in pp_1 because the same information is placed into the bits of higher significance. Same counts for pp_2 and pp_3 and, of course, vice versa. This will allow to still retrieve a correct provenance mark, even if corruption has annotated the dataset. Given a big enough number of data points a high redundancy will make this approach very robust.

Mathematically:
A provenance mark m has the length L_m and m_j stands for the jth bit of m. For each provenance mark pp^k the ith bit pp^k_i is calculated like this:

$$\text{pp}^k_i = \text{m}_j \text{ with } j = (k\tfrac{L_m}{N_{pp}} + i) \bmod L_m$$

Conditions: In order to guarantee that each bit of the mark will appear as often as every other one in all the provenance pieces L_{pp} has to divide $N_{pp} \times L_m$.
$N_{pp} \leq L_m$ to ensure that no two provenance pieces are equal.

Check bits

The check bits both are calculated by creating a hash of the significant bits s with either N_{pp} (for the parameter check bit pc) or with the provenance mark m (for the mark check bit mc).

Mathematically:

$$\text{pc} = \text{hash}(2, \text{s} \ @ \ N_{pp}) \quad (1)$$
$$\text{mc} = \text{hash}(2, \text{s} \ @ \ \text{m}) \quad (2)$$

3.3.2 Checking a provenance mark

One-bit watermarking is used to figure out if a provenance mark m' is actually the provenance mark m that originally had been embedded. For all normal cases this would not be needed but since transformations can corrupt a dataset and change it into an annotated dataset we need one-bit watermarking. This will allow us to still use the corrupted dataset.

Retrieving L_{sig} and Npp

Remember:

- L_{sig} is the number of significant bits in a data point
- N_{pp} is the number of distinct provenance pieces

For the retrieval process values for L_{sig} and Npp will be guessed. Then using that the equation (1) from above is being applied again with the respective values of the current (corrupted) data point d.

$$\text{hash}(2, d_0 ... d_{L_{sig}-1} \ @ \ N_{pp}) = d_{L_{sig}}$$

If L_{sig} and Npp have been guessed correctly and the data point was uncorrupted this equation will work and then $d_{L_{sig}}$ is exactly the parameter check bit pc. In all the other cases there is a probability of about $\frac{1}{2}$ that the data point is *pc-consistent*. This is because of the specific hash function used to calculate pc.

Definition:
The *pc-consistency score* will be "the proportion of data points" in a dataset "that are pc-consistent for guesses L_{sig} and Npp"[7]. Correct guesses and uncorrupted first $L_{sig}+1$ bits will lead to a score of 1, where-else a score of about $\frac{1}{2}$ can be expected. An incorrect guess when having a score of

doi: 10.2313/NET-2012-08-1_07

1 only appears with a probability of 2^{-n} (with n being the amount of data points in the set). Having a finite amount of data points with finite length in the datasets there are limited guesses. Calculating with different parameters L_{sig} and Npp will allow to finally pick the combination of L_{sig} and Npp which led to the highest *pc-consistency score*. Those will be used for a check of the provenance mark.

Checking

Similarly to the *pc-consistency (score)* a *mc-consistency (score)* can be calculated. This equation is derived from equation (2) (Check bits) from above:

$$\text{hash}(2,\ d_0 ... d_{L_{sig-1}}\ @\ \text{m}) = d_{L_{sig+1}}$$

Also similarly a mc-consistency is defined as the proportion of datapoints in a dataset that are *mc-consistent* for L_{sig} and m. If L_{sig} and m are correct and the first L_{sig+2} bits are uncorrupted the score will be 1, otherwise $\sim \frac{1}{2}$.

3.3.3 Retrieving a provenance mark

Retrieving a provenance mark from a corrupted dataset (*Blind Watermarking*) is a process that includes several steps. This is because each bit vector representation of a data point only contains a specific part of the whole provenance mark. Additionally, each piece might contain corrupted bits. For the retrieval of the provenance mark each bit vector will be split up in its specific pieces s, pc, mc, pp, and k, which tells us which provenance piece had been implanted in the current bit vector. That can easily be done using L_{sig} and Npp. Even if some marks are corrupted that is no problem because each mark appears several times and also at several places (different bits). Now, in order to find the correct mark guesses have to be made and rated. For that a "suggest-function" [7] is being used:

$$suggest(d,i) = \begin{cases} (pp_j, j) & \text{if } j + k \bmod L_m = i \\ & \text{and } 0 \leq j < |pp| \\ * & \text{otherwise} \end{cases}$$

Each provenance piece d contains parts of the whole provenance mark. The suggest-function *suggest(d,i)* tells whether the *ith* bit of the whole provenance mark is contained in d. Additionally it returns a number representing the confidence of the result (higher numbers represent a lower confidence). This is because a smaller bit pp_j (*jth* bit of the provenance piece d) has higher significance and is less likely to be corrupted. *Suggest* will return * if d does not contain i or the bit pp_j with the corresponding confidence.

Example:
provenance mark m = 0011
provenance piece pp = 01
Then suggest(pp,0) = *
Then suggest(pp,1) = (0,1)
Then suggest(pp,2) = (1,2)
Then suggest(pp,3) = *

But since the suggestion of a single data point is not the final goal suggest must be a function on the whole dataset *DS*. The new *suggest(DS,i)* will only take those values into account that are actual values, i.e. * will not be taken into the equation. The goal is to have *suggest(DS,i)* return the overall best guess for bit i of the complete provenance mark.

A best guess can be defined in many ways. There are two main ways ([7]):

- *All-Vote:* pick the suggestion that most data points suggest (independently of the respective confidences)

  $allVote(L) = \text{round}(\frac{\sum_{(b,c)\in L} b}{|L|})$

- *Best-Vote:* pick the suggestion that has the most confident suggestions in data points

  $bestVote(L) = \text{round}(\frac{\sum_{(b,c)\in L'} b}{|L'|})$ with L' being the subset of L containing only the best confidence values

Using either of the two methods for *suggest(DS,i)* the provenance mark can be constructed:

$$m_i = \text{f}(suggest(DS,i))$$

The newly created mark now needs to be checked using one-bit checking. This method is very robust. From a corrupted dataset with mostly uncorrupted check bits the provenance mark can still be constructed. Search using mc-consistency will then hopefully lead to a correct provenance mark.

3.3.4 Directed Search

Searching can become a very expensive and extensive process. Certain aspects have to be taken into account in order to keep the time consumption in a decent limit. For a directed search all the bits of a (guessed) provenance mark will be ordered by confidence. This will (as before) now lead to the main guess. But in case the best guess is not correct it is easy to try out similar possibilities.

Provenance mark:	1	1	0	1	0
Confidence in bits:	0.5	0.8	0.2	1	0.4
Order by confidence:	1	1	1	0	0

Figure 6: Example for ordering by confidence

As seen in the example above (Figure 6) this order is easily done. Now, after an incorrect guess the bit with the lowest confidence will be flipped (green bit) and then the new mark will be tried out.

The new mark would be: 11110

Using a recursive algorithm as seen in [7] will try out all possible marks (starting with the most confident guess and ending with the least confident guess):

```
search(n, m):
    if n=0 then
        check possible provenance mark m
    else
        search(n-1 ,m)
        flip bit i_n of m
        search(n-1, m)
```

search(n, m) will be called with the best guess for the provenance mark (m) and n, which is either the length of m or a limit l set by the user (l<|m|).
l would lead to a checking of the 2^l most confident guesses.

doi: 10.2313/NET-2012-08-1_07

3.4 Evaluation and Analysis

The authors of [7] included several pages of evaluation and analysis in their paper. Their algorithm was tested with a dataset that suffered the typical transformations (rounding, truncation, sampling of data points). This subsection provides a short summary of their results. One big point is that the method becomes the more robust against transformations the bigger the dataset is. A very big dataset will provide more uncorrupted data points and, in general, more redundancy. Secondly, embedding more metadata in every single bit vector will lead to better results but decrease the percebtibility. Although the mean over all data points will stay pretty much the same one might not want to replace too much data with a provenance mark. A compromise will result in good robustness and perceptibility. One would think, and that is clearly correct, that a higher number of provenance pieces k provides more redundancy and will therefore lead to a higher robustness because each bit of the provenance mark will appear in more distinct data points. This idea makes perfect sense, but only up to a certain point. Having to many distinct provenance pieces the method becomes less robust against sampling. More data points would be needed to guarantee that all different provenance pieces are included in the subset. As already mentioned earlier the *pc-consistency score* and the *mc-consistency score* end up being binomially distributed given a big enough dataset.

3.5 Issues and Summary

The presented technique of embedding a provenance mark into a sensor-dataset appears to be robust against transformations (up to a certain point) and is meant to be available as an open-source tool. This sounds like a perfect solution but some issues remain:
First of all, the mechanism is no security measurement. Given enough criminal energy a provenance mark could be removed from the dataset. But the authors of [7] see their technique rather as support for the " 'fair use' policies typical of publicly available sensor network data" [7]. Another point is that, even if the provenance mark is only embedded into the metadata part of each data point, some publishers might not want that. It still would (not significantly) change their data and with them offering it they might feel like they were offering some wrong data. Two more factors are that it is not clear at what stage of the process the provenance mark should be embedded (raw sensor data or processed data). With that being unclear confusions and problems could arise. Also might the presented algorithm (especially the use of the hash-function) lead to a high time consumption in very big datasets. This would force one to use a cryptographically less expensive (and therefore less secure) method. Overall there are good applications for the technique and, overcoming the few issues mentioned, it can be applied for real-life applications.

4. CONCLUSION

This paperpresented the topic "Electronic Watermarking" starting with its origin, development and applications. After that general approach one specific, new method is presented: Self-identifying sensor data uses the noise in a dataset to embed provenance marks into each data point. This approach is to guarantee that data can successfully be related to a creator/owner, even if several transformations corrupt the dataset. In comparison to the "traditional" applications of watermarking the technique for sensor data is new and quite different. The main difference in the new technique derives directly from the need for it. Oppositely to the common data sets which watermarking is being applied to sensor data will be given in an unstructured way. Normal database watermarking techniques can therefore not be applied to it. Secondly, sensor data also varies between user communities and therefore an adaption providing more robustness had to be provided. Summarized one can say that the new technique is stronger because it opposes no structural prerequisites to the raw data. It can therefore find applications in many fields where the traditional methods could not work.

5. REFERENCES

[1] *Digital Watermarking Alliance*, `http://www.digitalwatermarkingalliance.org/`, Webpage found on: April, 24th, 2012

[2] *Civolution*, `http://www.civolution.com/technology/digital-audio-and-video-watermarking/`, Webpage found on: April, 24th, 2012

[3] *IBM Research*, `http://domino.watson.ibm.com/library/cyberdig.nsf/0/21bddc7a17e34fd1852565930070afbb?OpenDocument`, Webpage found on: April, 24th, 2012

[4] Ingemar J. Cox, Matt L. Miller: *The first 50 years of electronic watermarking*, NEC Research Institute, 4 Independence Way, Princeton, NJ 08540, USA, April 19, 2002

[5] William M. Tomberlin, Louis G. MacKenzie, Paul K. Bennett: *System for transmitting and receiving coded entertainment programs*, United States Patent, 2,630,525, 1953

[6] Anoop M. Namboodiri, Anil K. Jain: *Multimedia Document Authentication using On-line Signatures as Watermarks*, Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824

[7] Stephen Chong, Christian Skalka, Jeffrey A. Vaughan: *Self-Identifying Sensor Data*, Stockholm, Sweden, April 12-16, 2010

[8] Prof.Dr.Eng. Monica Borda: *Fundamentals in Information Theory and Coding*, ISBN: 978-3-642-20346-6, Springer-Verlag, Berlin Heidelberg, 2010

[9] *Google Maps*, `http://maps.google.de/maps?hl=de&tab=wl`, Webpage found on: April, 24th, 2012

[10] *Shazam*, `http://www.shazam.com`, Webpage found on: April, 24th, 2012

[11] Avery Li-Chun Wang: *An Industrial-Strength Audio Search Algorithm*, Shazam Entertainment, Ltd., Palo Alto, CA, USA

[12] Emil Frank Hembrooke. Identification of sound and like signals. United States Patent, 3,004,104, 1961.

doi: 10.2313/NET-2012-08-1_07