Proceedings of the 2006 IEEE
Workshop on Information Assurance
United States Military Academy, West Point, NY

Secure Visualization of GIS Data

Stephen D. Wolthusen

Abstract— **Modern GIS systems increasingly rely on server-side rendering and web services for the rendering of geographical and application-specific data for both efficiency and security reasons since the underlying data sets for critical infrastructures and emergency operations are typically extremely sensitive. Given that display devices can be spread in the field on mobile devices, the ability to track and trace leaking and misuse of visualization data is of critical importance. In this paper we describe a technique to insert robust steganographic markings into the rendering process for GIS data based on context-sensitive texture adaptation along with a system architecture for marking and tracing GIS service data over a standards-based communication channel.**

I. Introduction

While geographical information systems (GIS) have long been used in defense applications and are increasingly popular and widespread for commercial, scientific, as well as – somewhat surprisingly – also for leisure use, one of their most important recent application areas emerging is assisting decision makers in gaining situational awareness in critical circumstances. Relevant environments include tactical situations, emergency response, and other critical infrastructure protection (CIP) requirements.

These application areas share two critical properties. First, the data presented via the GIS interface may be highly sensitive and must be kept confidential. Such confidential information may include application data layers containing the location and additional information on sensitive objects (e.g. of pipelines in a CIP or emergency response environment or of troop positions and movements in a tactical environment).

This first property, however, is counteracted by the second property and recent development, namely the increasing use of GIS information outside of secured areas, including on mobile devices such as personal digital assistants in the field. While particularly the latter use is highly desirable from an operational perspective, such usage patterns, also including the dissemination of hard copy maps, poses a significant risk.

While control over electronic distribution from the GIS system can be effected, there are still significant risks associated with secondary distribution such as screenshots from a secured GIS application or hard copies of GIS displays and maps. This has been recognized and, as a result, digital watermarking techniques have been developed for two-dimensional vector data frequently used in the GIS application scenario [1], [2], [3].

Particularly for mobile applications, however, direct visualization based on raw vector and application layer data is typically inappropriate since both the volume of data required may exceed the storage capacity of the device and the processing power required for rendering may not be available on such devices, resulting in slow rendering or excessive resource (e.g. battery) consumption. Both because of this and to permit access to current data from various sources, server-side processing and rendering is becoming increasingly important.

To this end, the OpenGIS consortium (OGC) has formulated a series of open standards which permit the use of web service-based mapping (WMS) and feature data servers as well as the conjoining of data sets from multiple servers into a single display by providing a visual representation of arbitrary georeferenced data sets [4], [5], [6]. This standard, however, suffers from two deficiencies that currently preclude use of WMS in sensitive application areas.

First, the standard itself makes no provision for identification and authentication of end points or users, and also does not mandate confidentiality and integrity protection mechanisms. The second is the abovementioned lack of protection of rendered data once it is prepared for display by the requesting systems.

In this paper we present a security architecture which combines on-line protection mechanism using cryptographic mechanisms with a digital watermarking mechanism that can extend the overall protection also to off-line representations of GIS data such as hard copies. This comprehensive protection mechanisms allows the use of WMS for critical and confidential data without requiring modifications to OGC standards, client or server software. To this end, section II presents an algorithm for the embedding of digital watermarks within rendered textures of GIS data, while section III discusses a transparent proxy architecture for use on client and server-side systems which permits the in-line embedding of watermarks and also ensures confidentiality, integrity, and identification & authentication properties for the communication channel. Section IV then briefly reviews related work while section V discusses the results and current research.

Gjøvik University College, Norway and Royal Holloway, University of London, UK

II. Context-Sensitive Steganographic Marking

Unlike digital watermarking for general and particularly photographic images [7], the rendered data in a GIS application are computer generated and hence do not contain noise levels a priori (see figure 1 for an example). The sole exception to this are layers containing satellite or aerial imagery, which can be trivially marked with existing image watermarking algorithms but are beyond the scope of this paper.

The fact that the images are computer-generated also points to a general limitation that distinguishes the media type discussed here from other media typically the subject of digital watermarking, namely that a re-drawing of map data, including small random perturbations, can eliminate all watermarks without significantly degrading the semantics of the map.

Fig. 1. Rendered map excerpt

This, moreover, not only applies to the algorithms discussed here but also to algorithms operating on the underlying vector and geospatial data [1] and any other approach that must maintain semantic fidelity. Map vendors have in some instances inserted fictitious locations and place names in their map materials to ensure that wide-scale copying of their materials can be detected and copyright violations can be proven. However, particularly for the application areas considered here, the risk of providing deliberately false data is deemed unacceptable. However, the work factor required for homogenizing, randomizing and subsequently re-drawing a complete map does, provide a level of protection against the general type of re-drawing attack described above.

General robustness of the algorithms described below is therefore limited to automated blind recovery from affine transformations, color space transformations, and cropping (see section II-B) and, given either manual intervention or an original image, also to projective transformations. In particular, the algorithm is robust against typical transformations encountered in digital to analog conversions (e.g. generation of hard copies). This provides adequate protection for threats such as inadvertent disclosure and adversaries with limited resources.

The marking process itself can be placed in two steps in the delivery process for map data. First, it is possible to emplace the marking process directly within the rendering mechanism, i.e. to embed markings while individual raster images are generated. While this is a highly efficient process, it also requires adaptation for each individual WMS server system. The second alternative is to remain independent of the actual rendering system and embed the markings based on rendered image data.

While the latter requires image analysis, it is independent of the WMS service used and can be integrated flexibly in the process chain. Therefore, the following section describes this more general second approach for embedding markings while section III describes a transparent embedding in the network communication stack.

A. Location and Placement of Textures

The first step in the embedding process is the identification of segments with well-defined boundaries. Given the characteristics of the computer-generated maps, this can be accomplished by a number of highly efficient and simple processes such as e.g. Gaussian smoothing for removing existing texture information and subsequently application of the Canny edge detector [8], [9], although other algorithms may be appropriate for more complex textures and map types [10], [11]. For the purposes of the embedding procedure, it is sufficient to derive segments with continuous edges and no further requirements are levied on these segments.

For the following algorithm, let the map be represented by a matrix A with horizontal dimension x and vertical dimension y and, without loss of generality, its origin in the top left corner. Algorithm 1 places a sliding tile of fixed but arbitrary size (here: $n = 8$) $n \times n$ pixel) over the map and collects texture tiles to be marked.

Algorithm 1 ensures a set of non-intersecting (but not necessarily consecutive) tiles of size $n \times n$ in which watermark textures (a *stipple pattern*) can now be placed. It should be noted that both shape and size of the tiles were selected arbitrarily as any regular tessellation can provide the error recovery mechanism described in section II-B whereas recovery of other tilings may present difficulties for registration [12].

For each of the tiles in the tessellation, a stipple pattern to be overlaid over the tile must be derived. Depending on robustness and capacity requirements, this pattern can either be reiterated over multiple tiles within a tessellation (see section II-B), or it can be varied to pack more information in a single segment.

Algorithm 1 Placement of Stipple Pattern

Require: Image is rotated counterclockwise such that north is aligned with the image abscissa

Require: Segment interior start and end coordinates are available as $start_k$ and end_k for each abscissa coordinate.

$i \leftarrow 0$
$j \leftarrow 0$
$tileset = \emptyset$
repeat
  $k \leftarrow start_i$
  Find a horizontal consecutive run of n pixels inside the texture segment
  repeat
    For each pixel of the run of pixels, find a vertical run of n consecutive pixels inside the texture segment.
    if Tile *tile* is found **then**
      if $\forall t \in tileset.$(*tile* does not intersect t) **then**
        $tileset \leftarrow tileset \cup tile$
      end if
    end if
    $j \leftarrow j + 1$
  until $j > end_i$
  $i \leftarrow i + 1$
until $i > y$

Algorithm 2 assumes an ideal cryptographic hash function H [13], [14]. Given the properties of ideal cryptographic hash functions, any arbitrary but fixed sequence of bits from the output bits generated by such a function retain the hash properties of the entire function, particularly the avalanche effect, allowing the selection of an output subset of functions such as the SHA-1 or SHA-256 algorithms.

The watermark requires the use of a secret key to ensure unforgeable derivation of the watermark pattern; however, the following briefly reviews constraints on the length of the mark to be embedded itself.

Given that one of the objectives for the marking process described in this paper is the ability to verify the origin and authenticity of a given map (e.g. including the data source and the individual or device for which the map was rendered), the probability of false positives or collisions between keys in detecting watermarks must be minimized.

Assuming that m is the number of unique key vectors, an ideal cryptographic hash function, and a payload size of k bits, the number of compact representations is $N = 2^k$; the ratio of the number of compact representations without collisions to the total number of vectors is $\prod_{i=0}^{m-1}\left(1 - \frac{i}{N}\right)$. Since $1 - x \leq e^{-x}$ (for $x > 0$) one obtains

$$\prod_{i=0}^{m-1}\left(1-\frac{i}{N}\right) \leq \prod_{i=0}^{m-1} e^{-\frac{i}{N}} = e^{-\sum_{i=0}^{m-1}\frac{i}{N}} = e^{-\frac{m(m-1)}{2N}}$$

Thus, the probability of a collision after inserting $\sqrt{2N}$ keys $1 - \frac{1}{e}$. For a 64 bit payload one collision in approximately 6 billion recoveries can be expected, which may be considered acceptable for most applications.

It is therefore sufficient to use an embedding mechanism that provides a probability distribution (owing to distortions and possible attacks) for the presence or null hypothesis for a key length of 64 bits for each tile of the tessellation under consideration; the following description shall use this key length l.

Algorithm 2 transforms the key into a pattern to be used for detection; since the probability distribution of the key after application of the hash function is assumed to be equiprobable, it is sufficient to partition the result of the hash function into tile coordinates through a linear mapping.

The actual embedding can then occur by forming a histogram over the tile to determine the weighted power density of the image and selecting an offset from the average depending on application criteria (for example, robustness to black-and-while printouts requires higher contrast values) to be used for the stipple pattern.

While the algorithms described in this section can be applied directly on image material, it is possible to further improve results by incorporating information that can be obtained by the communication adaptation layer described in section III since additional information such as geopositions referenced and transparency of layers can be learned directly from the communication channel.

Algorithm 2 Derivation of Stipple Pattern

Require: A tile T represented as a $n \times n$ matrix

Require: A key *key* of arbitrary length

Select l fixed but arbitrary bits from the the output bitstring of $H(key)$ as a vector $H_{key}^{1 \leq i \leq l}$
for all $0 < i < \frac{l}{2n}$ **do**
  $T[H_{key}[i] + H_{key}[2(i+1)] + H_{key}[4(i+2)] + \cdots H_{key}[128(i+7)], H_{key}[(i+8)] + H_{key}[2(i+9)] + H_{key}[4(i+10] + \cdots + H_{key}[128(i+15)]] \leftarrow \overline{T}[H_{key}[i] + H_{key}[2(i+1)] + H_{key}[4(i+2)] + \cdots H_{key}[128(i+7)], H_{key}[(i+8)] + H_{key}[2(i+9)] + H_{key}[4(i+10] + \cdots + H_{key}[128(i+15)]]$
end for

Figure 2 shows a map excerpt enlarged to the single pixel level with an added raster to permit easier identification of individual pixels; figure 3 shows the same excerpt after application of algorithms 1 and 2. However, it should be

noted that the color and contrast of the stipple pattern itself in figure 3 has been exaggerated for illustrative purposes and that 8 instead of 4 pixels (with a key length of $l = 128$) for $n = 8$ were used. As can easily be seen that the above description also applies to any scale of raster items such as groups of pixels of arbitrary size.

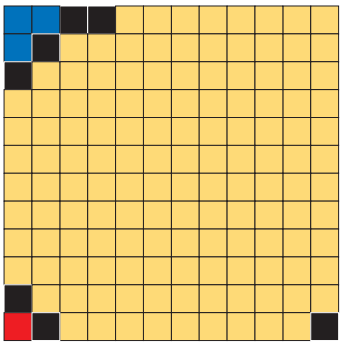

Fig. 2. Magnified rendered map excerpt

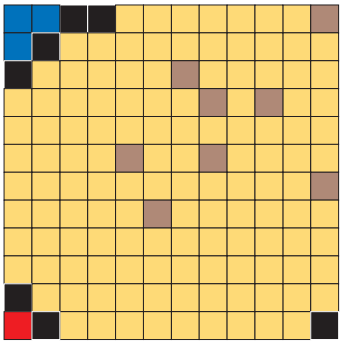

Fig. 3. Modified rendered map excerpt

It should be noted that in addition to simple repetition of a texture within a given tessellated segment, which may frequently be desirable since this obviously results in regular textures, it is also possible to generate irregular textures using algorithm 2 by simply using the result of an initial hash function as the seed of an iterative pseudo-random function. However, while this still yields the ability to recover a marking in case of some modifications or attacks, automatic recovery may no longer be possible if initial tiles of a tessellation are no longer recoverable or have otherwise been lost (e.g. through cropping).

The derivation of the key itself is beyond the scope of this paper since it will typically be application dependent. However, as with all watermarking algorithms it is desirable to include information on the media to be marked and a timestamp within the key source material to preclude certain types of logical attacks on claims made by originators of digital watermarks [7]. In case of geospatial data, the semantic media reference can e.g. consist of a geospatial position located within the rendered map together with a timestamp. This, however, must also be accompanied by random secret key material to prevent forgeability of the key material itself by third parties.

B. Recovery Mechanism

Recovery of the stipple pattern requires several image processing steps depending on the intermediate processing that the marked map has undergone. In the general case (i.e. when the map has undergone processing and transformations) initial step for this process is the registration of the map image to be re-segmented.

Assuming a non-blind detection process, the initial registration can be partly automated by trivially obtaining a segment-based match and performing an affine transformation based on three points identified both in the original and the representation under test.

In case the representation under test is in color, the next processing step is the binarization of the image.

Given that intervening processing steps may have introduced rounding errors and shifts, it may then be necessary to re-match the individual modified pixels. For each axis, tiles can be identified by considering a fixed but arbitrary line (column) in the raster image as a second-order stationary process with mean μ while scaling k and μ in parallel:

$$R(k) = \frac{E[(X_i - \mu)(X_{i+k} - \mu)]}{\sigma^2}$$

This autocorrelation permits identification of the raster period of tilings and, given a stipple pattern as described above, permits reverting of any affine transformation if three points in each stipple pattern can be identified. In case of non-blind recovery, this can be accomplished by comparing the values of the original and the representation under test, while a blind detection mechanism must resort to the assumption that the local environment around a stipple point location is homogeneous and comparing the stipple value (after binarization) with surrounding pixels. Where no cropping has occurred, a direct application of algorithm 1 can occur.

In case of repeated tile patterns within a tessellated segment, these probability values can be improved significantly by considering the pattern locations as conditional probabilities.

C. Efficacy

By utilizing the semantics of the underlying data types, the approach presented here can provide a more visually appealing and at the same time robust marking mechanism for rendered data compared to the direct application of general-purpose watermarking algorithms, even if those algorithms are optimized for binary data. General image watermarking algorithms, moreover, are frequently unsuitable for this purpose since they are typically optimized for

the psycho-visual model underlying photographic representations and introduce noise and fuzziness which, while acceptable and largely unnoticeable in photographs, are visually disturbing have the potential to alter the semantics of rendered GIS data with its well-defined semantics (e.g. boundaries of a textured area).

III. Secure Communication Layer Adaptation

As described in section I, existing services for the delivery of rasterized geospatial visualization data, particularly the OGC Web Map Service, do not incorporate essential security properties such as the integrity, confidentiality, and authenticity of the communication channel or the identification and authentication of the communication endpoints. Given the widespread use of such services in deployed servers ad applications, however, changes to the standard and subsequent adoption of such changes can be assumed to be relatively slow. Conversely, proprietary extensions to the standard services are equally undesirable since this would jeopardize the interoperability critical in many application areas.

The following section describes an adaptation layer for communication security that can be emplaced on both WMS clients and servers and which is implemented for the Microsoft Windows family of operating systems. It should be noted, however, that similar mechanisms can also be implemented on other operating system platforms such as Unix derivatives or Linux and that the external interface presented by the adaptation layer described here complies with the relevant IETF standards for securing the transport layer [15], [16].

A. Communication Security

The objective of the communication security layer is to ensure mandatory mutual identification and authentication between systems exchanging geospatial data over web-based services such as OGC WMS and to secure the communication channels' confidentiality, integrity, and authenticity.

This is accomplished by inserting an intermediate layer into the the TCP/IP network stack of the operating system in such a way that it is non-bypassable for unprivileged users and applications. For the Windows family of operating systems, several steps are required to perform this embedding since there exist a number of different interfaces that could otherwise allow bypassing. Within the protocol stack [17], all layers and interfaces must, however, ultimately bind to the NDIS (Network Driver Interface Specification) layer, which provides the direct interface to physical network devices. Access to this layer must be controlled since otherwise it is possible for application programs to gain raw access to the network interface.

While some services may interface directly with the NDIS layer (e.g. the Telephony service), most applications, however, are directed through the Transport Driver Interface (TDI), which consists of transports or protocol drivers. While various protocols such as NetBIOS are available, the dominant protocol is clearly TCP/IP; this protocol is in turn provided by the WinSock protocol driver. Both NDIS and WinSock are remarkable in that, though they have undergone significant revisions, their origins pre-date the Windows NT family of operating systems; in the case of NDIS the original protocols were designed for the IBM DOS operating system, while WinSock originated as an extension of the Windows 3.0 extension to the DOS operating system [18], [19]. The latter origin is the explanation for the fact that the WinSock layer is not a proper kernel-mode driver but rather a hybrid and the reason why bindings to the TDI layer must be controlled to prevent manipulations of WinSock layer.

The WinSock API itself provides the interface DLL (dynamic-link library) exposing the public interfaces to user programs and communicates with the SPI (Service Provider Interface) layer, which in turn is controlled by the transport service provider DLL. The transport service provider DLL makes use of several additional libraries, namely so-called transport helper DLLs and namespace helper DLLs. The transport service provider DLL then forwards the generated calls to the System Support Library DLL, which formats the communication requests and directs them to the TDI layer. Another translation mechanism is required for several variants of network communication that is handled through a file handle mechanism; the requisite additional state information is retained by an Ancilliary Function Driver (AFD) DLL since the underlying communication layer may not provide this state information.

There exists an interface for adding functionality such as filtering and traffic modifications to the WinSock layer in the form of additions to the transport service providers. Such components are called Layered Service Providers (LSP) and are inserted into the Service Provider Chain Catalog Entries for the relevant protocols (in the case of the application protocols discussed here, this can be accomplished by way of TCP only). This insertion of filter layers is protected from manipulations by unprivileged users since all relevant data structures are protected by access control lists restricted to the system and administrators.

However, since other application programs such as virus scanners or personal firewalls also make use of this mechanism and may exhibit programming errors that can leave the layering mechanism unstable, it is desirable to verify the proper bindings of all relevant portions of the network stack for both security and reliability reasons.

With LSPs installed on both client and server systems, access control lists can now be used to require strong authentication for some channels. In this case, the connection to be opened (typically by a request from a client system)

is augmented by the initiation of a TLS session including mutual authentication. To this end, both clients and server systems require the presence of a public key certificate for access within the LSP[1], which is then used in session establishment according to the IETF RFCs 2246 and 2817 [15], [16][2].

Once identification and authentication have been established (or, failing that, the connection has been reset), the LSPs can then perform a cipher suite negotiation using the standard TLS mechanisms; however, to ensure that no weak ciphers are used as may be the case for backward compatibility reasons in other application areas, cipher suites and protocol revisions must be restricted in advance at each LSP. Only at this stage, i.e. after establishing the secure communication channel, is the WinSock API allowed to signal establishment of the communication channel to the respective application channels.

Given the computational complexity and delays inherent in the identification and authentication and cipher suite negotiation stages, particularly for mobile devices with limited resources and operating on network links with high latency, application usage patterns can affect performance. In case of service-oriented applications such as WMS where no continuous TCP link between hosts is established, this TLS-induced latency can be reduced significantly by caching the session IDs for hosts with which a TLS connection had been established and resuming the current session if both hosts still have valid session ID entries in their respective caches.

Otherwise, a full session re-negotiation must take place as outlined above. In each case, however, the TLS layer now transparently inserted provides confidentiality and integrity protection and can also detect several other types of attacks such as replay attacks and can therefore, given a suitable cipher suite, provide adequate security and performance adapted to the resources of the devices communicating.

A.1 Out-of-Band Communication Channels

In addition to the transparent translation of the communication channel between client and server systems, the client-side LSP can also perform additional services by establishing out-of-band communication channels. This may e.g. include individual user account identification (since the LSP operates at the machine level, the client-side certificate does not necessarily identify an individual user) to be used for access control or billing information that can be transmitted either via the server LSP or directly to a (secured) server-side interface.

[1]Since this certificate is not visible to the end user it can but need not be tied to a specific user and may originate in a different CA structure than other certificates used on the system in question.

[2]The revised TLS 1.1 standard was not yet at the RFC stage and hence was not included. This does, however, open the possibility of manipulations of the RSA-encrypted pre-master secret, so an upgrade to TLS 1.1 on standardization is required for security reasons.

B. Semantic Analysis of OGC WMS Data Streams

An additional benefit that, particularly in the case of OGC WMS, can be derived from the communication adaptation layer is the ability to capture parameters and metadata associated with the requests for and rendered geospatial data itself.

The OGC WMS provides several metadata parameters that can be used to categorize and identify the maps and layers generated from the georeferenced data sets including the source coordinates of the data set and a time stamp for the data set (not to be mistaken with a time stamp used in the generation of the watermark pattern key).

Standard WMS descriptors of interest in addition to the above in the determination of the watermark segmentation and texture computation are the background color and transparency values which can be specified in each WMS `GetMap` request and for each layer.

While there exist a number of additional parameters, particularly information on layers and styles, these are not fully defined within the standards documents and are hence subject to extension and interpretation by WMS system vendors. However, since the actual data is formatted in XML, the analysis and extraction of such proprietary extensions for possible use in the embedding process is easily extended.

For the purposes of the marking process described here, the standard OGC WMS parameters outlined above are, however, sufficient to select appropriate shapes and textures that match colors and transparency values; by identifying multiple layerings, the contrast and color of such combined layers can be further refined to achieve an optimum balance between robustness and visibility.

IV. Related Work

GIS systems have rapidly gained widespread acceptance in military applications and are also increasingly used in emergency and critical infrastructure protection environments. Initial usage, however, has primarily concentrated on static planning and assessment applications [20], [21]. Since 2002, the Open GIS Consortium (OGC) standards for the representation of visualization output for georeferenced data provide an interoperable mechanism for retrieving, modifying, and layering rendered maps [4], [5], [6]. A general review of legal and copyright issues in digital watermarking for GIS data is provided by Lopéz [22]. While the review by Lopéz also reviews technical solutions, these results have since been superseded.

Voigt *et al.* describe techniques for embedding digital watermark data in two-dimensional vector data, particularly also for use in map data [1], [2], [3]. Their approach directly manipulates symbols (e.g. by shifting symbols within a threshold) and hence may influence the se-

mantics within the maximum dislocation limits set by the algorithm parameters unless purely limited to annotation symbols. This approach, however, is also susceptible to the normalization and re-drawing attack described in section II. Similar re-normalization attacks are also trivially possible through polygon coalescion in the vector insertion algorithm proposed by Thoen and Huber [23], which also operates directly on the vector data, albeit without regard to the semantics of the vector representation itself.

V. Conclusion

In this paper we have presented a new digital watermarking approach for use with rasterized rendered geospatial data visualizations which provides the ability to embed markings in tessellated textures of the segments formed by the map data. This approach differs from common digital watermarking techniques by incorporating significant information on the type and format of the source material, thereby allowing the robust marking of a media type (computer generated imagery) which the more common noise-based mechanisms (e.g. spread-spectrum-based approaches) can only handle with significant trade-offs between robustness and quality. Both the structure of the segmented map data and the watermark texture pattern itself provide the means for efficient and in most cases automatic registration and recovery while repeated texture patterns can reduce the probability of false recovery and act as a means for error correction.

The above provides a means for tracking and tracing rendered data once it has been delivered to edge devices and also after generation of hard copies and can hence assist in both locating the sources of undesired disclosures and also act as a deterrent to negligent handling and inappropriate distribution of potentially sensitive GIS data.

Given the deficiencies of the OGC WMS standard in particular for providing identification and authentication and secure communication, the second part of this paper provided a solution for inserting a transparent TLS security layer into the transport layer which provided the necessary security extensions and can also be used to communicate out of band information such as access control data to control mechanisms added to the WMS standard without requiring modifications to application programs or standard violations. Moreover, the transport layer can at the same time not only act as the most implementation-independent interception point for insertion (and possibly also detection and recovery) of digital watermarks but also, through analysis of the request and reply stream of WMS messages easily analyze and incorporate information into the watermarking process such as the precise geoposition of the area of interest depicted in a rendered visualization.

A. Future Work

The algorithm described in this paper is optimized for use in regular, segmented, and computer generated images as they will typically be used in GIS environments in that it does not disturb texture boundaries (and hence affects the semantics of rendered data) or the actual global texture pattern. Future work particularly includes detailed analyses of marking robustness in case of layering, which may introduce a number of distortions including layer opacity and particularly rounding errors in transforming projections; particular attention must also be paid to the effects of including image data such as satellite or aerial imagery on the recoverability of the texture pattern.

Additional future work will also focus on potential impacts on subjective map readability induced by the watermarks, particularly in situations where high-robustness markings are to be displayed on small-scale screens.

Acknowledgments

Parts of the research resulting in this paper were conducted on behalf of the Fraunhofer Institute for Computer Graphics, Darmstadt, Germany. Rendered GIS data were made available by GIStec GmbH, Darmstadt, Germany.

References

[1] M. Voigt and C. Busch, "Watermarking 2D-Vector Data for Geographical Information Systems," in *Security and Watermarking of Mulitmedia Contents IV* (E. J. Delp III and P. W. Wong, eds.), vol. 4675 of *Electronic Imaging*, (San Jose, CA, USA), pp. 621–628, International Society for Optical Engineering, Jan. 2002.

[2] M. Voigt and C. Busch, "Feature-Based Watermarking of 2D-Vector Data," in *Security and Watermarking of Mulitmedia Contents V* (E. J. Delp III and P. W. Wong, eds.), vol. 5020 of *Electronic Imaging*, (Santa Clara, CA, USA), pp. 359–366, International Society for Optical Engineering, Jan. 2003.

[3] M. Voigt, B. Yang, and C. Busch, "Reversible Watermarking of 2D-Vector Data," in *Proceedings of the ACM Multimedia and Security Workshop 2004 (MM&Sec '04)*, (Magdeburg, Germany), pp. 160–165, ACM Press, Sept. 2004.

[4] J. de la Beaujardière, "Web Map Service Implementation Specification," Tech. Rep. OGC 01-068r2, Open GIS Consortium, Wayland, MA, USA, Nov. 2001.

[5] J. D. Evans, "Web Coverage Service," Tech. Rep. OGC 03-065r6, Open GIS Consortium, Wayland, MA, USA, Aug. 2003.

[6] P. A. Vretanos, "Web Feature Service Implementation Specification," Tech. Rep. OGC 02-058, Open GIS Consortium, Wayland, MA, USA, Sept. 2003.

[7] M. Arnold, M. Schmucker, and S. D. Wolthusen, *Techniques and Applications of Digital Watermarking and Content Protection*. The Artech House Computer Security Series, Norwood, MA, USA: Artech House, 2003.

[8] J. Canny, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679–698, Nov. 1986.

[9] J. C. Russ, *The Image Processing Handbook*. Boca Raton, FL, USA: CRC Press, 4th ed., 2002.

[10] J. B. Mena, "State of the Art on Automatic Road Extraction for GIS Update: A Novel Classification," *Pattern Recognition Letters*, vol. 24, pp. 3037–3058, Dec. 2003.

[11] Y. Ji, K. H. Chang, and C.-C. Hung, "Efficient Edge Detection and Object Segmentation Using Gabor Filters," in *Proceedings of the 42nd Annual Southeast Regional Conference*, (Huntsville, AL, USA), pp. 454–459, ACM Press, Apr. 2004.

[12] B. Grünbaum and G. C. Shephard, *Tilings and Patterns*. New York, NY, USA: W. H. Freeman & Co., 1986.
[13] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, FL, USA: CRC Press, 1996.
[14] M. Luby, *Pseudorandomness and Cryptographic Applications*. Princeton Computer Science Notes, Princeton, NJ, USA: Princeton University Press, 1996.
[15] T. Dierks and C. Allen, "The TLS Protocol Version 1.0." IETF Network Working Group Request for Comments 2246, Jan. 1999.
[16] R. Khare and S. Lawrence, "Upgrading to TLS Within HTTP/1.1." IETF Network Working Group Request for Comments 2817, May 2000.
[17] M. E. Russinovich and D. A. Solomon, *Microsoft Windows Internals*. Redmond, WA, USA: Microsoft Press, 4th ed., 2004.
[18] D. B. Andersen, "Windows Sockets 2 Service Provider Interface," tech. rep., Intel Corp., May 1997. Version 2.2.1.
[19] D. B. Andersen, "Windows Sockets 2 Application Provider Interface," tech. rep., Intel Corp., May 1997. Version 2.2.1.
[20] M. de Zuviria and S. McClure, "Comparative Study of GIS Data Products Used in Various-Sized Municipalities for Emergency Management and Critical Infrastructure Protection across Canada," Tech. Rep. PS48-6/2004E, Public Safety and Emergency Preparedness Canada, Ontario, Canada, Dec. 2003.
[21] M. Terner, R. Sutton, B. Hebert, J. Bailey, H. Gilbert, and C. Jacqz, "Protecting Critical Infrastructure," *GeoIntelligence*, vol. 1, pp. 8–12, Mar. 2004.
[22] C. Lopéz, "Watermarking of Digital Geospatial Datasets: A Review of Technical, Legal and Copyright Issues," *International Journal of Geographic Information Science*, vol. 16, pp. 589–607, Sept. 2002.
[23] W. Thoen and W. Huber, "GIS & Steganography." Directions Magazine, Apr. 2002. On-line magazine: `http://www.directionsmedia.net`, last accessed Feb. 20, 2006.