引文格式:袁天洋,朱长青,陈会仙,等.面向OpenDRIVE格式高精地图的脆弱水印法[J].武汉大学学报(信息科学版),2026,51(2):403-412.DOI:10.13203/j.whugis20230500

Citation: YUAN Tianyang, ZHU Changqing, CHEN Huixian, et al. Fragile Watermarking Algorithm for High-Definition Map of OpenDRIVE Format[J]. Geomatics and Information Science of Wuhan University, 2026, 51(2): 403-412. DOI: 10.13203/j.whugis20230500

面向OpenDRIVE格式高精地图的脆弱水印法

袁天洋[1,2,3] 朱长青[1,2,3] 陈会仙[4] 任 娜[1,2,3,5] 吕旭超[1,2,3]

1 南京师范大学虚拟地理环境教育部重点实验室,江苏 南京,210023
2 江苏省地理环境演化国家重点实验室培育建设点,江苏 南京,210023
3 江苏省地理信息资源开发与利用协同创新中心,江苏 南京,210023
4 自然资源部地图技术审查中心,北京,100830
5 湖南省地理信息安全与应用工程研究中心,湖南 长沙,410017

摘 要:针对高精地图数据完整性保护问题,提出了一种基于零宽度字符和信息摘要算法的OpenDRIVE格式高精地图脆弱水印法。水印嵌入过程中,先按照文档节点的树状结构与道路要素的起始点坐标排序建立位置关系,再提取节点外部文本生成脆弱水印信息,并结合建立的位置关系将水印嵌入到对应节点。在验证时,对比生成的水印信息与提取的水印信息,根据两者差异鉴定数据完整性。实验结果表明,所提算法生成的水印不可见性良好,不会改变地图所包含的信息,验证水印时能够精准定位文档中发生的篡改,引入的数据增量与原始文件大小的比值通常可控制在5%以内。

关键词:脆弱水印;高精地图;零宽度字符;XML;文本水印

中图分类号:P208 **文献标识码**:A **收稿日期**:2024-03-19

DOI:10.13203/j.whugis20230500 **文章编号**:1671-8860(2026)02-0403-10

Fragile Watermarking Algorithm for High-Definition Map of OpenDRIVE Format

YUAN Tianyang[1,2,3] *ZHU Changqing*[1,2,3] *CHEN Huixian*[4] *REN Na*[1,2,3,5] *LÜ Xuchao*[1,2,3]

1 Key Laboratory of Virtual Geographic Environment, Ministry of Education, Nanjing Normal University, Nanjing 210023, China
2 State Key Laboratory Cultivation Base of Geographical Environment Evolution (Jiangsu Province), Nanjing 210023, China
3 Jiangsu Center for Collaborative Innovation in Geographical Information Resource Development and Application, Nanjing 210023, China
4 Map Supervision Centre, Ministry of Natural Resources, Beijing 100830, China
5 Hunan Engineering Research Center of Geographic Information Security and Application, Changsha 410017, China

Abstract: **Objectives**: High-definition maps are regarded as key data resources in the development of digital transportation, making their data integrity protection critically important. However, conventional hash algorithms used for file verification are not capable of inferring the location of tampered content from a single checksum anomaly. Therefore, an algorithm that can not only verify the integrity of high-definition map data but also accurately locate tampered regions is essential for mitigating network security risks in digital transportation and safeguarding public safety. To address this issue, this paper proposes a fragile watermarking algorithm specifically designed for high-definition maps in the OpenDRIVE format. **Methods**: The proposed algorithm is designed based on unicode zero-width characters and message digest algorithm 5 hash algorithm. The proposed algorithm uses zero-width character sequences to embed watermark information, with extensible markup language nodes serving as the basic units for watermark generation, embedding, extraction, and verification. The watermark information corresponding to each node is composed of a combination of a tree watermark and a road feature watermark. To avoid affecting the normal use of map da-

基金项目:国家自然科学基金(42071362)。

第一作者:袁天洋,硕士生,主要研究方向为测绘地理信息安全。221302129@njnu.edu.cn

通信作者:朱长青,教授。649397417@qq.com

ta, the watermark sequence is embedded at the end of the line corresponding to the starting tag or the unique tag of each node. During data distribution, the administrator can use this algorithm to generate fragile watermarks, embed them into high-definition map data and verify whether the watermarks are successfully embedded in the file. When the user needs to verify the integrity of the file, this algorithm can be used to extract and verify the watermark. If the two are identical, the file is considered intact, otherwise, the algorithm determines the tampered location based on the detected anomalies. **Results:** Experimental results demonstrate that embedding watermarks into high-definition map documents does not cause any visible abnormal display. The watermark embedding process introduces no significant increase in file size, and the ratio of the size increment to the original file size can be controlled within 5%. In addition, the proposed algorithm exhibits high sensitivity to both node-level and watermark-level attacks and is capable of accurately locating tampered positions. In cases of whole-element deletion, the geometric information of the deleted element can still be inferred. **Conclusions:** The proposed algorithm is suitable for the integrity protection of high-definition map data, and can also be applied to the integrity protection of other structurally similar data such as hypertext markup language and cascading style sheets, as long as cancelling some preprocessing requirements, reconfirming the appropriate watermark embedding position and clarifying the definition of features and the sorting rules between features.

Key words: fragile watermarking; high-definition map; zero-width characters; XML; text watermarking

中国交通运输行业信息化数字化已取得了长足发展[1]。高精地图是一种主要用于高级别辅助驾驶和智能驾驶的专用电子地图[2],是数字交通发展中的关键数据资源。高精地图实现了对普通导航地图的颠覆性升级,其服务对象由人转变为机器,在部分应用情境下具备制作与使用同步实时更新的特征,不仅服务于导航,还直接参与导航决策过程,是驾驶环境的数字孪生体,具有高精度、高丰富度和高动态性等特点,在自动驾驶中发挥着不可替代的指挥员作用[3]。在高精地图数据分发和共享的过程中,数据可能被无意或恶意篡改。然而,应用于文件校验的杂凑算法在设计时未考虑实现根据单一校验码异常的情况推断消息内部被篡改内容所在位置的功能[4]。因此,亟需一种既能验证高精地图数据完整性,又能精准定位篡改位置的技术手段,以有效防范和化解数字交通领域的网络安全风险,切实保障人民生命财产安全。

数字水印技术是一种能将版权和用户信息作为水印内容隐蔽嵌入到数据中,并与原始数据融为一体、不可分离的技术[5]。其中,鲁棒水印主要用于版权保护与溯源追踪,脆弱水印主要用于数据的真伪辨别和完整性鉴定。脆弱水印能够有效检测数据是否被篡改、何处被篡改以及篡改到何种程度,从而保障数据可靠、可信、可用[6]。尽管在鲁棒水印检测过程中产生的中间参数也可用于推断数据完整性是否遭到破坏,但是无法定位篡改发生的位置,且存在漏检篡改的风险。因此,在完整性验证应用场景中,脆弱水印方案更具优势。

目前,针对高精地图数据的专用数字水印算法研究相对较少。OpenDRIVE格式高精地图数据本质上是一种符合可扩展标记语言(extensible markup language,XML)规范[7]的非格式化文本数据[8]。针对该格式的数据,可借鉴已有的文本水印算法。现有相关算法大致可分为以下4类:

1)基于自然语言处理的水印算法。这类算法通过对文本中的短语、词汇、字母进行等价或近似的替换,实现水印的嵌入。王炳锡等[9]、LI等[10]和MIR等[11]提出的水印算法适用于面向人类的文本应用场景,但在面向机器的文本应用场景中,严重影响文档可用性。蔡毅等[12]、姚荣华等[13]和CHEN等[14]利用XML文档标签属性不区分大小写的特性嵌入水印,虽然不会增加存储与传输负担,但水印容量高度依赖英文字符数量,不可见性较差,且不具备抵抗大小写转换攻击的能力。

2)基于词频统计的零水印算法。斯琴等[15]和AL-WESABI等[16]设计的文本水印算法不会改动原始文本,能够抵抗常见攻击,但是需要搭建或租用版权保护服务器,对使用者网络环境有一定的要求。

3)基于零宽度字符的水印算法。李兆璨等[17]、陈旖旎等[18]和彭登[19]将水印信息转化为零宽度字符序列并嵌入在文本中。这类算法生成的水印不可见性良好,水印容量不受原始文本限

制,不会影响机器读取数据。但是若不对水印序列长度进行限制,过长的水印序列极易使得文档大小发生剧烈变化,严重占用存储资源,影响读取和传输效率。此外,部分零宽度字符在特定软件环境下虽不可见,但在更换浏览软件后则可能出现肉眼可察觉的异常显示。

4)基于字符变形和字库替换的水印算法。XIAO等[20]和QI等[21]设计变形拉丁字符并向英文字符编码库中添加编码,孙杉等[22]和姚晔等[23]设计变形汉字字符并向中文字符编码库中添加编码,再利用变形字符替换正常字符的方式嵌入水印。这种方法不可见性良好,不会增加存储开销。但是,在面向机器读取的情境下,机器需要额外安装含变形字符的编码库以正确读取数据,用户会增加额外的软件开销。

综上所述,现有相关算法各具优缺点,且多为鲁棒水印算法,针对高精地图数据完整性验证的脆弱水印研究仍较为有限。更为关键的是,现有相关算法应用于高精地图文档时,难以同时满足以下三个需求:(1)水印嵌入后无肉眼可见的异常显示;(2)水印嵌入引起的文件大小变化有限且可控;(3)能够精准定位篡改位置,并在发生整要素删除时推断被删除要素的几何信息。为了满足上述需求,本研究以零宽度字符为水印序列元素,以信息-摘要算法(message digest algorithm 5,MD5)为数学基础,针对高精地图数据特性提出了一种脆弱水印算法。

1　面向高精地图的脆弱水印算法

1.1　总体思想

面向OpenDRIVE格式高精地图的脆弱水印算法(以下简称高精地图脆弱水印算法)以零宽度字符序列为基础,以XML节点为单位,生成、嵌入、提取和验证水印信息。每一节点对应的水印信息都由树状水印和道路要素水印组合而成。为了不影响数据的正常使用,水印序列的嵌入位置选定在每一个节点对应的起始标签或唯一标签的所在行行末。经检验,此处嵌入水印信息不会影响自动驾驶设备读取高精地图数据。

当数据被分发时,发行方可利用本文算法生成脆弱水印并将水印嵌入到高精地图数据中,然后验证水印是否成功嵌入。需要验证文件完整性时,使用方可利用本文算法提取与并验证水印。如果两者完全一致,则告知文件完整性通过检验,否则,算法根据异常情况确定篡改位置。

1.2　水印序列元素

为了改进传统的基于零宽度字符的文本水印在更换软件浏览后易出现异常显示的问题,本研究设计了不可见性测试,以寻找不可见性更好的零宽度字符。通过测试的零宽度字符将作为组成高精地图脆弱水印序列的零宽度字符元素。表1是查阅统一码官方文档后确定的部分零宽度字符的编号以及含义。所有零宽度字符分别组成只包含单一字符且长度不小于20的序列,再各自嵌入一段文本,并依次在Windows记事本、Visual Studio、Chrome内核的浏览器和Notepad++等多种常用于查看和编辑非格式化文本的软件中打开,观察嵌入前后其显示效果是否出现可见的变化。只有3种零宽度字符U+200B、U+180E和U+FEFF在所有测试到的常用文本编辑器与浏览器中均不会引发任何显示上的异常。因此,将它们选作水印序列元素,为后续表述方便,将其依次对应为三进制数0、1和2。

表1　部分零宽度字符及含义

Table 1　Part of Zero-Width Characters with Meanings

编号	含义	编号	含义
U+180E	蒙文元音分隔符	U+202B	嵌入式从右至左标记
U+200B	零宽度空格	U+202C	文字方向变化结束
U+200D	零宽度连接符	U+202D	强制从左至右标记
U+200E	成对从左至右标记	U+202E	强制从右至左标记
U+200F	成对从右至左标记	U+206A	禁止对称交换
U+202A	嵌入式从左至右标记	U+FEFF	零宽度无间断空格

1.3　水印生成

由于组成水印序列的零宽度字符仅有上文确定的3种,每个零宽度字符所能承载的信息量较小。为了保证在这一前提下生成水印不仅具有序列随机性和解码结果唯一性,而且嵌入后不造成文件大小剧烈变化,本文算法在水印生成环节采取两种措施:(1)截取哈希值低位,根据所保护的内容长度及重要性缩短水印信息长度,以此缩短水印序列长度;(2)使用三进制最优前缀编码映射十六进制哈希值,在保证解码结果唯一性前提下尽可能缩短水印序列。各十六进制数对应的编码如表2所示。这些编码的本质是当前频率下最优三叉树的叶子节点,如图1所示。

为了更好地检测和定位篡改,本文算法嵌入的每一处水印信息都由两部分组成:

1)树状水印。树状水印的长度随节点在树中的深度增加而递减,当节点深度增加至一定层级时,其对应的树状水印长度最终减小为0。

表2 十六进制数字最优前缀编码表

Table 2 Prefix Code of Hexadecimal Digits

数字	编码	数字	编码
0	00	8	101
1	010	9	102
2	011	A	110
3	012	B	111
4	020	C	12
5	021	D	20
6	022	E	21
7	100	F	22

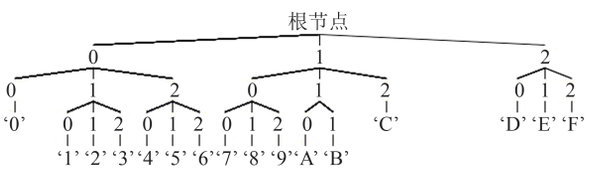


图1 十六进制数字最优三叉树示意图

Fig. 1 Optimal Ternary Tree of Hexadecimal Digits

树状水印能有效避免算法在单处篡改且篡改导致节点局部发生哈希碰撞的极端情况下完全漏检篡改。

根据节点r的外部文本O_r生成对应的哈希值v_r,取其低位h字节记为$v_{r,h}$,编码$v_{r,h}$得到零宽度字符序列$m_{r,h}$。不同节点对应的h取值不同。对于节点r,其树状水印字节数h_r与该节点的深度d_r以及文本长度l_r关系为:

$$h_r=\begin{cases}8, d_r=0\\ 1, 0<d_r\leqslant 2 且 l_r\geqslant 4\,000\\ 0.5, d_r=1 且 l_r<4\,000\\ 0.5, d_r=2 且 500\leqslant l_r<4\,000\\ 0, d_r>2\\ 0, d_r=2 且 l_r<500\end{cases} \tag{1}$$

当节点r为道路要素节点时,利用文件头节点外部文本O_{hdr}生成的哈希值v_{hdr}第53~56位生成文档标记m_{hdr},并将其连接在$m_{r,h}$后。若节点r为非道路要素节点,且为同标签名节点中首个被遍历到的节点,设标签名为L_n的所有要素节点的外部文本依遍历序依次连接得到的文本为O'_{L_n},据O'_{L_n}生成的哈希值为v_{L_n},由v_{L_n}低位1字节生成的非道路要素标记为m_{L_n},将m_{L_n}连接在$m_{r,h}$后,由此得到节点r的树状水印m_r。

2)道路要素水印。对于描述道路要素的节点,道路要素水印信息字节数$h'=2$,对于其余节点,其为空白串,即$h'=0$。设计道路要素水印的目的是允许算法帮助用户在道路要素被整要素删除的情况下,也能推断出被删除的道路要素大致位置。具体生成方式为:获取道路要素集合R_i在参考线起点处惯性坐标系的坐标值(x_i, y_i),分别对横坐标x_i和纵坐标y_i由小到大排序,排序结果为x'_i与y'_i。根据这一结果确定对于每一个道路要素对应的节点,以及所需要的文本用于生成该节点对应的道路要素水印。道路要素水印由4个水印信息字节数为0.5的部分组成:由横坐标排序x'_i中的前一节点f确定的$m'_{r,x,f}$、后一节点l确定的$m'_{r,x,l}$、由纵坐标排序y'_i中的前一节点确定的$m'_{r,y,f}$以及后一节点确定的$m'_{r,y,l}$。对于每个道路要素R_i,记其对应的节点为r_i,则R_i对应生成$m'_{r,x,f}$的输入文本$T_{i,x,f}$和$T_{i,x,l}$,有:

$$T_{i,x,f}=\begin{cases}O_{r_i}, I(x'_i)=1\\ O_{r_{(I(x'_i)-1)}}, I(x'_i)\neq 1\end{cases} \tag{2}$$

$$T_{i,x,l}=\begin{cases}O_{r_i}, I(x'_i)=c\\ O_{r_{(I(x'_i)+1)}}, I(x'_i)\neq c\end{cases} \tag{3}$$

式中,$I(x'_i)$为R_i在横坐标排序结果中的位置;$r_{(I(x'_i)+1)}$为起始点横坐标序号为$I(x'_i)+1$的道路要素对应的节点;c为道路要素总数。

$T_{i,x,f}$和$T_{i,x,l}$生成的哈希值分别记为$v'_{r,x,f}$和$v'_{r,x,l}$。对上述哈希值的低位1字节进行编码,取其高位0.5字节,分别得到$m'_{r,x,f}$与$m'_{r,x,l}$。依照同样的规则,可得$T_{i,y,f}$和$T_{i,y,l}$,生成其哈希值$v'_{r,y,f}$与$v'_{r,y,l}$,并编码其低位0.5字节,生成$m'_{r,y,f}$与$m'_{r,y,l}$。依次连接$m'_{r,x,f}$、$m'_{r,x,l}$、$m'_{r,y,f}$和$m'_{r,y,l}$,构成道路要素水印m'_r。将树状水印m_r与道路要素水印m'_r依次连接,即生成最终嵌入到文件中的水印序列m''_r。

1.4 水印嵌入

高精地图脆弱水印算法嵌入水印的操作可分为4个步骤,流程如图2所示。

1)合法性判断和数据预处理。首先尝试读取文档根节点,若无法读取,则判定数据为非高精地图文档,输出提示并拒绝处理文档。在数据合法的前提下,为保证水印内容的稳定性,需要判断文档中是否已存在水印。若已存在水印,则输出提示并拒绝再次处理该文档。

2)获取生成水印所需信息。获取文档根节点并遍历整个文档树,获取生成水印所需全部节点的信息集F_r,r为深度优先遍历过程中节点的访问序号,$r=1, 2, \cdots, c$,c为文档树中节点总数。对于任一节点r,其对应的节点信息定义为(N_r, n_r, d_r, O_r),其中,N_r、n_r、d_r、O_r分别为节点r的起始行号,标签名、深度和外部文本。

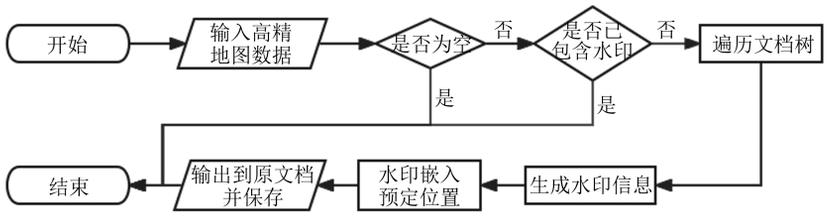


图 2　水印嵌入流程图

Fig. 2　Watermark Embedding Flowchart

3)水印序列生成。利用 F_r 生成水印序列集合 m_r。

4)嵌入水印并保存。将 m_r 中的水印序列嵌入到预定位置,即节点 r 对应的起始或唯一标签所在行的行末,并将修改后的内容保存至原文档中。

1.5　水印验证

高精地图脆弱水印算法的水印验证操作的主要流程如图 3 所示。

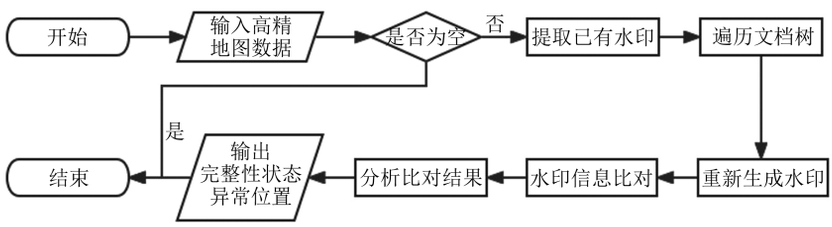


图 3　水印验证流程图

Fig. 3　Watermark Validating Flowchart

1)预验证操作。如果不能读取到文档根节点,则判定数据非高精地图文档,输出提示。并拒绝处理文档。在数据合法的前提下,验证开始前,先提取已有水印并生成除去水印的副本。提取到的已有水印信息为:

$$M_s=(N_s,m_s) \tag{4}$$

式中,s 为提取水印的信息序号,$s=1,2,\cdots,L$,其中 L 表示提取水印信息总数;N_s 为提取水印序列所在行的行号;m_s 为提取水印序列所在行水印序列内容。

2)根据文档非水印内容重新生成参考水印信息。打开无水印副本,获取文档根节点并遍历整个文档树,获取生成水印所需全部节点的信息 F_r,根据 F_r 生成水印序列集合 m_r,过程与§2.2 方法一致。完成水印序列生成后,删除无水印副本。将 m_r 中所有水印与其嵌入位置关联,得到参考水印信息:

$$M_s'=(N_s',m_s') \tag{5}$$

式中,s 为参考水印信息的序号,$s=1,2,\cdots,L'$,其中 L' 为参考水印信息总数;N_s' 和 m_s' 分别为参考水印序列应嵌入位置行号和参考水印序列内容。

3)水印信息比对。逐条比对提取水印信息 M_s 与参考水印信息 M_s'。如果某一行对应水印信息不一致,即证明发现文件异常。输出不一致的水印对应的 XML 节点标签名、起始行行号与异常类型,并输出异常处的要素类型与 ID 号。

4)分析比对结果。在当前节点在起始点位置排序中,其前一节点与后一节点的树状水印和要素水印均存在的前提下,对异常类型进行进一步分析:(1)文件中新增节点。根据新增节点的来源不同,其水印异常特征有所差异。若新增节点由其他含水印文档截取得到,则该节点的树状水印中文档标记部分出现异常。若该节点由篡改者自行构造,则其树状水印缺失,同时其所有亲代节点的树状水印均出现异常。当新增节点为道路要素节点时,其要素水印出现异常,且在起始点坐标排序中,前一节点的要素水印中与后一节点相关的部分异常,后一节点的要素水印中与前一节点相关的部分异常。当新增节点为道路要素节点的后代节点时,其所属要素节点的树状水印出现异常,同时该所属要素节点在起始点坐标排序中,前一节点要素水印中与后一节点相关的部分异常,后一节点要素水印中与前一节点相关的部分异常;(2)文件中删除节点。当被删除节点为道路要素节点时,在起始点坐标排序中,其原前一节点的要素水印中与后一节点相关的部分出现异常,原后一节点的要素水印中与前一节点相关的部分出现异常。当被删除节点为道路要素节点的后代节点时,其所属要素节点的树状水印出现异常,同时该所属要素节点在起始点坐标排序中,其前一节点要素水印中与后一节

点相关的部分异常,后一节点要素水印中与前一节点相关的部分异常。当被删除节点为非道路要素节点时,需根据该节点是否为遍历过程中第一个被访问到的同标签名要素节点进行判断,该类非道路要素节点表现为节点标记缺失或内容异常。当被删除节点为非道路要素节点的后代节点时,其所属要素节点的树状水印出现异常;(3)节点内容发生变动。当节点内容发生变动时,该节点及其所有包含水印的亲代节点的树状水印均出现异常。若该节点位于道路要素节点中,则其所属要素节点在起始点坐标排序中,其前一节点的要素水印中与后一节点相关的部分出现异常,而后一节点的要素水印中与前一节点相关的部分出现异常。当该节点的水印信息总长度为0,或其在起始点坐标排序中不存在前一节点或后一节点时,则不产生对应的异常信息。

5)输出结果。若无异常,输出完整性通过验证的信息。若有异常,输出发生异常的要素信息,如有需要,可将具体的水印异常情况一并输出以辅助研判。如遭遇整要素删除,则输出推断的被删除要素大致范围。

2 实验及分析

为了验证由高精地图脆弱水印法生成的高精地图脆弱水印是否达到预定设计要求,从不可见性、对篡改敏感性以及含水印文件大小变化三个角度进行分析。实验数据为描述某交通环岛的高精地图文档RoundAbout-3Arms.xodr,可视化效果如图4所示。

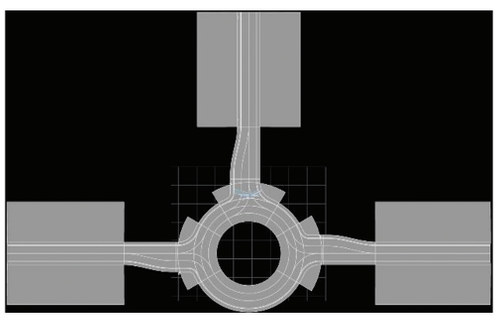
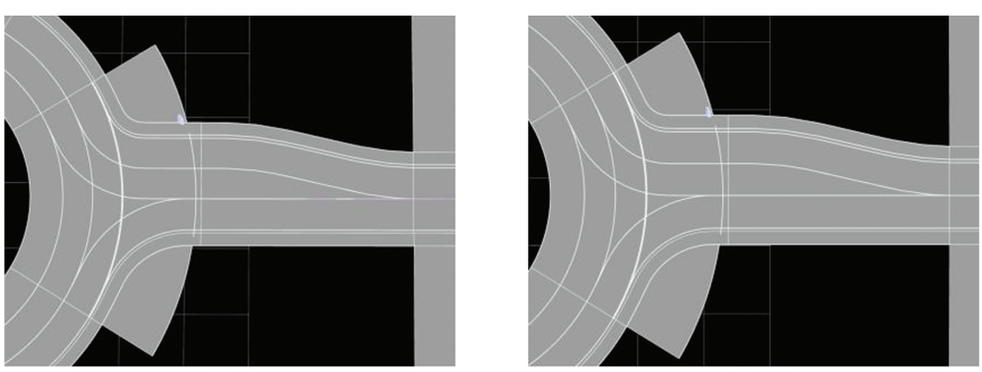


图4 原始高精地图概览图

Fig. 4 Overview Image of Original High Precision Map

2.1 不可见性分析

不可见性是指在水印嵌入文本后,数据在视觉层面上无肉眼可见变化的性质。首先对水印嵌入前后的数据可视化效果进行比较,图5(a)是原始数据局部放大效果,图5(b)是嵌入水印后对应区域的可视化效果。由图5可见,水印未改变高精地图数据的可视化效果,且本文算法不修改属性值,根据属性值所包含的几何信息与属性信息生成的可视化效果理论上与无水印文档一致。因此,在数据可视化效果方面,水印的不可见性良好。



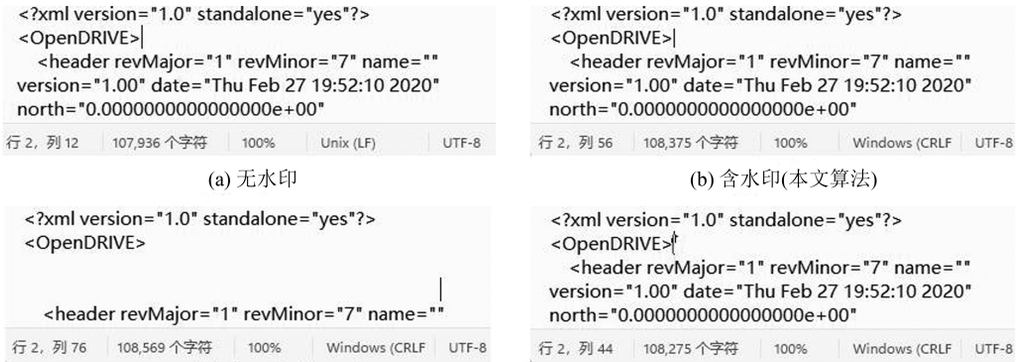
(a) 无水印 (b) 含水印

图5 局部可视化效果对比

Fig. 5 Comparison of Partial Visualization Effect

然后,对文本显示效果进行对比。受到篇幅的限制,以对控制字符最敏感的Windows记事本为例,使用本文算法嵌入水印后效果如图6(b)所示,使用文献[17-18]水印序列元素嵌入内容相同的水印后效果见图6(c)和图6(d)。图6中黑色实心竖线为行末光标位置。由图6可见,高精地图脆弱水印虽然改变了行长度,但未改变行末位置,亦未产生额外的显示效果。因此,在文本显示效果方面,水印的不可见性良好。

最后,从算法是否对高精地图所承载的地理信息造成改动的角度分析。本文提出的高精地图脆弱水印法只在特定标签末尾添加零宽度字符序列,该过程不涉及对高精地图文档中任何几何坐标、拓扑关系或属性字段数值的修改,也不会引起数据结构的重排或精度损失,所以含水印高精地图文档中包含的所有几何信息以及属性



(a) 无水印　(b) 含水印(本文算法)

(c) 含水印(文献[17])　(d) 含水印(文献[18])

图 6　Windows 记事本内文本局部显示效果对比

Fig. 6　Comparison of Partial Text Display Effects in Windows Notepad

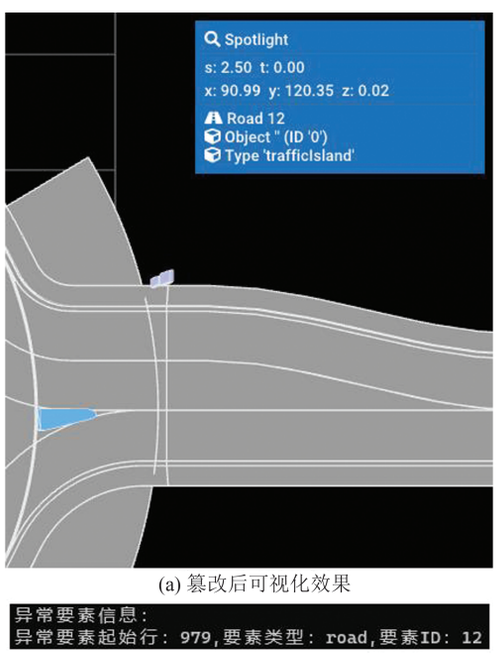
信息都与原始高精地图文档完全一致。因此,在是否改变所载信息角度方面,水印的不可见性亦良好。

2.2　对篡改敏感性分析

攻击者可能在高精地图文档在从分发方将数据传输到用户终端的过程中对含水印的高精地图数据进行篡改。错误路网数据的高精地图可能扰乱自动驾驶车辆对交通环境的感知,造成危害性后果。本文对模拟攻击者可能的攻击手段,验证高精地图脆弱水印算法是否如既定设想检出并定位篡改。

2.2.1　针对道路要素内部结构的攻击

1)道路要素内部增加节点。向 ID 为 12 的道路要素内部增加一个包含障碍物信息的子节点,该子节点为原地图文档中存在的交通岛障碍物节点。篡改后可视化效果与脆弱水印验证结果如图 7 所示。对比验证结果和实际发生的篡改可知,本文算法准确定位了被篡改的要素。



(a) 篡改后可视化效果

(b) 检测结果

图 7　增加节点攻击

Fig. 7　Node Addition Attack

2)道路要素内部删除节点。将 ID 为 1 的道路要素中 ID 为 −2 的车道删除。篡改后可视化效果与脆弱水印验证结果如图 8 所示。对比分析结果和实际发生的篡改可知,本文算法准确定位了被篡改的要素。

3)改动节点内容。篡改 ID 为 1 的道路要素中心线参数。篡改后可视化效果与脆弱水印验证结果如图 9 所示。对比验证结果和实际发生的篡改可知,本文算法准确定位了被篡改的要素。

2.2.2　针对道路要素的攻击

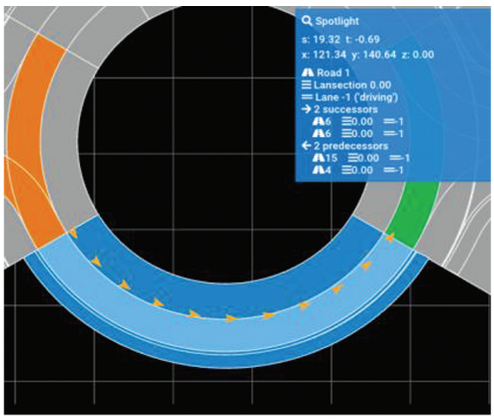
1)删除道路要素。将 ID 为 3 的道路要素删除,篡改后原位可视化效果如图 10(a)所示,推断结果如图 10(b)所示,推断被删除要素位置在地图上的位置如图 10(a)中红色矩形框所示。对比验证结果与实际发生的篡改可知,本文算法检测到整要素删除,且基本准确地推断了被删除要素的位置。

2)添加道路要素。将描述某立体交叉路口的含水印高精地图文档 map4.xodr 中 ID 为 18 的道路要素添加到 RoundAbout-3Arms.xodr 文档。篡改后地图全览效果如图 11(a)所示,蓝色高亮位置为添加的道路要素,检测结果如图 11(b)所示。对比验证结果与实际发生的篡改可知,本文算法准确定位到了不属于原文档的要素。

2.2.3　针对水印序列的攻击

删去水印序列部分内容。将 ID 为 10 的道路要素节点对应的水印序列删去 3 个零宽度字符,验证结果如图 12(a)所示。

1)移除水印序列。移除 ID 为 42 的交叉路口要



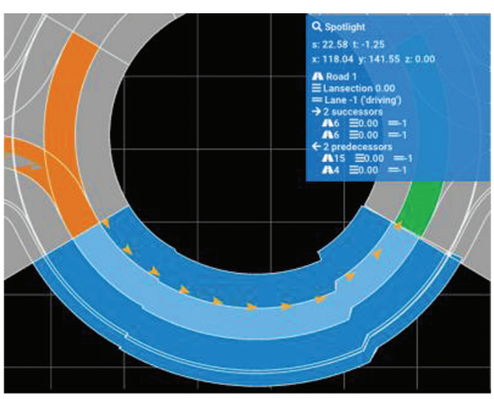
(a) 篡改后可视化效果

```
异常要素信息:
异常要素起始行: 5,要素类型: road,要素ID: 1
```

(b) 检测结果

图8 删除节点攻击

Fig. 8 Node Deletion Attack



(a) 篡改后可视化效果

```
异常要素信息:
异常要素起始行: 5,要素类型: road,要素ID: 1
```

(b) 检测结果

图9 改动节点内容攻击

Fig. 9 Node Contents Altering Attack

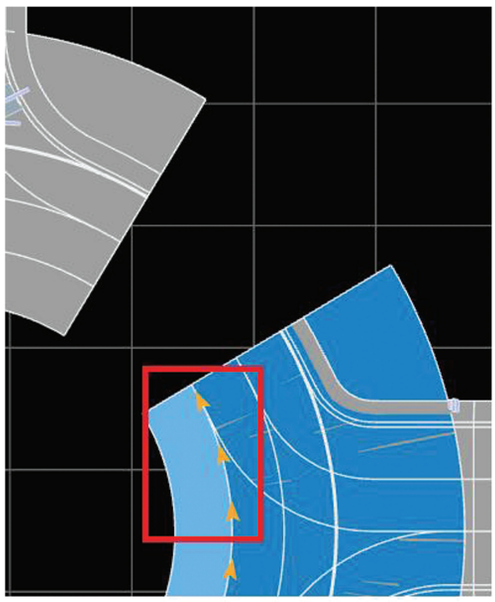
素节点对应的水印序列,验证结果如图12(b)所示。

2)交换水印序列位置。将ID为42的交叉路口要素和ID为43的交叉路口要素的水印序列交换位置,脆弱水印验证结果如图12(c)所示。

3)对比分析结果和实际发生的篡改可知,本文算法准确定位了针对水印序列的攻击发生的位置,并报出了相应的异常详情。

2.3 水印导致高精地图文件大小变化分析

为了解使用本文算法嵌入水印后高精地图文件大小产生的变化,向多份高精地图文档示例数据中嵌入了水印,对比嵌入前后的文件大小,部分结果如表3所示。针对多份来源、规模及要素复杂度不同的高精地图数据,嵌入水印造成的增量在原始文件大小的0.69%~4.10%之间。描述每个要素的文本的平均长度越大,文件大小变



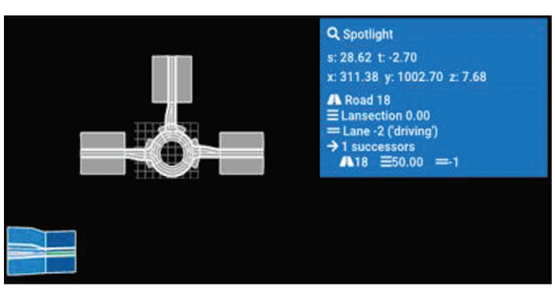
(a) 篡改后可视化效果和推断篡改位置

```
异常要素信息:
推断异常要素大致位置在以下四点连线范围内:
(97.7249,109.7036)-(97.7249,120.0000)
-(104.8540,120.0000)-(104.8540,109.7036)
```

(b) 检测结果

图10 整要素删除攻击

Fig. 10 Feature Deletion Attack



(a) 篡改后可视化效果

```
异常要素信息:
异常要素起始行: 1436,要素类型: road,要素ID: 18
推断异常要素大致位置在以下四点连线范围内:
(156.0132,1000.0000)-(156.0132,1458.4657)
-(468.4638,1458.4657)-(468.4638,1000.0000)
```

(b) 检测结果

图11 添加要素攻击

Fig. 11 Feature Appending Attack

```
解码失败! 检测到非法编码!
水印值异常, 异常行: 1544, 原因: 水印无法被正确解码。
异常要素信息:
异常要素起始行: 1544,要素类型: road,要素ID: 10
```

(a) 检测结果:一处水印序列不完整

```
水印值异常, 异常行: 1627, 原因: 水印值缺失。
异常要素信息:
异常要素起始行: 1627,要素类型: junction,要素ID: 42
```

(b) 检测结果:一处水印序列缺失

```
水印值异常, 异常行: 1627, 原因: 树状水印内容异常。
水印值异常, 异常行: 1645, 原因: 树状水印内容异常。
异常要素信息:
异常要素起始行: 1627,要素类型: junction,要素ID: 42
异常要素起始行: 1645,要素类型: junction,要素ID: 43
```

(c) 检测结果:两处水印序列异常

图12 水印序列攻击验证结果

Fig. 12 Validation Results of Watermark Sequence Attacks

表3　嵌入水印后文件大小变化

Table 3　File Size Expansion After Embedding Watermark

文件名	要素数目	嵌入水印前大小/Bytes	嵌入水印后大小/Bytes	要素平均长度/Bytes	增量与原始大小的比值/%
UC_ParamPoly3	11	37 051	38 363	3 368.3	3.54
map4	52	325 224	333 439	6 254.3	2.53
新元高速_20221013	314	939 851	978 380	2 993.2	4.10
changsha_part1_1014	434	2 370 133	2 375 950	5 461.1	2.45
Germany_2018	179	12 594 829	12 682 053	70 362.2	0.69

化的程度越轻。通常情况下,描述每个要素的文本的平均长度不会低于3 000字节,此时增量与原始大小的比值可以控制在5%以内。因此,利用本文算法嵌入水印不会导致高精地图文件大小剧烈变化。

3　结　语

本文针对OpenDRIVE格式的高精地图数据的完整性保护问题,选用不可见性良好的查阅统一码零宽度字符作为载体,利用前缀编码与信息-摘要算法,结合文档自身结构提出了一种适用的脆弱水印算法。结果表明,本文算法达到以下设计要求:

1)向高精地图文档中嵌入水印后,不会造成肉眼可见的异常显示现象。

2)嵌入水印后不会导致高精地图文件大小剧烈变化,增量与原始大小的比值可以控制在5%以内。

3)面对针对节点以及针对水印的攻击具有高度的敏感性,且能够准确定位篡改发生的位置。发生整要素删除时,能推断被删除要素的几何信息。

对于其它结构相似的数据,如超文本标记语言文档和层叠样式表文档,只要取消部分预处理要求、重新确定合适的水印嵌入位置并重新明确要素的定义以及要素间的排序规则,本文算法可以用于这些数据的完整性保护工作。

参　考　文　献

[1] 中华人民共和国交通运输部. 数字交通“十四五”发展规划[EB/OL]. (2021-10-25) https://www.mot.gov.cn/zhuanti/shisiwujtysfzgh/202201/P020220112576470472593.pdf
Ministry of Transport of the People's Republic of China. The 14th Five Year Plan for the Development of Digital Transportation[EB/OL]. (2021-10-25) https://www.mot.gov.cn/zhuanti/shisiwujtysfzgh/202201/P020220112576470472593.pdf

[2] 李必军, 郭圆, 周剑, 等. 智能驾驶高精地图发展与展望[J]. 武汉大学学报(信息科学版), 2024, 49(4): 491-505.
LI Bijun, GUO Yuan, ZHOU Jian, et al. Development and Prospects of High Definition Map for Intelligent Vehicle[J]. *Geomatics and Information Science of Wuhan University*, 2024, 49(4): 491-505.

[3] 齐如煜, 尹章才, 顾江岩, 等. 高精地图的知识图谱表达[J]. 武汉大学学报(信息科学版), 2024, 49(4): 651-661.
QI Ruyü, YIN Zhangcai, GU Jiangyan, et al. Knowledge Graph Expression of High Definition Map[J]. *Geomatics and Information Science of Wuhan University*, 2024, 49(4): 651-661.

[4] 王小云, 于红波. 密码杂凑算法综述[J]. 信息安全研究, 2015, 1(1): 19-30.
WANG Xiaoyun, YU Hongbo. Survey of Hash Functions[J]. *Journal of Information Security Research*, 2015, 1(1): 19-30.

[5] 朱长青, 任娜, 徐鼎捷. 地理信息安全技术研究进展与展望[J]. 测绘学报, 2022, 51(6): 1017-1028.
ZHU Changqing, REN Na, XU Dingjie. Geo-Information Security Technology: Progress and Prospects[J]. *Acta Geodaetica et Cartographica Sinica*, 2022, 51(6): 1017-1028.

[6] 侯翔, 闵连权, 唐立文. 定位篡改实体组的矢量地图脆弱水印算法[J]. 武汉大学学报(信息科学版), 2020, 45(2): 309-316.
HOU Xiang, MIN Lianquan, TANG Liwen. Fragile Watermarking Algorithm for Locating Tampered Entity Groups in Vector Map Data[J]. *Geomatics and Information Science of Wuhan University*, 2020, 45(2): 309-316.

[7] 詹骄, 郭迟, 雷婷婷, 等. 自动驾驶地图的数据标准比较研究[J]. 中国图象图形学报, 2021, 26(1): 36-48.
ZHAN Jiao, GUO Chi, LEI Tingting, et al. Comparative Study on Data Standards of Autonomous Driving Map[J]. *Journal of Image and Graphics*,

2021, 26(1): 36-48.

[8] 赵卫娟, 关虎, 黄樱, 等. 文本水印技术研究综述[J]. 中国传媒大学学报(自然科学版), 2020, 27(6): 55-62.
ZHAO Weijuan, GUAN Hu, HUANG Ying, et al. A Survey of Text Watermarking[J]. *Journal of Communication University of China* (*Science and Technology*), 2020, 27(6): 55-62.

[9] 王炳锡,陈琦,邓峰森. 数字水印技术[M]. 西安: 西安电子科技大学出版社, 2003.
Wang Bingxi, Chen Qi, Deng Fengsen. Technology of Digital Watermarking[M]. Xi'an: Xidian University Press, 2003.

[10] LI Q C, ZHANG J, ZHANG Z H, et al. A Chinese Text Watermarking Based on Statistic of Phrase Frequency[C]//2008 International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Washington, USA, 2008.

[11] MIR N, KHAN M A U. Copyright Protection for Online Text Information: Using Watermarking and Cryptography[C]//The 3rd International Conference on Computer Applications & Information Security (ICCAIS), Riyadh, Saudi Arabia, 2020.

[12] 蔡毅, 李峰, 周亮, 等. 一种基于XML的数字水印技术[J]. 现代计算机(专业版), 2005, 11(6): 66-68.
CAI Yi, LI Feng, ZHOU Liang, et al. An Digital Watermark Technology Based on XML[J]. *Modern Computer*, 2005, 11(6): 66-68.

[13] 姚荣华, 赵启军, 卢宏涛. 基于水印的XML文档完整性保护算法[J]. 计算机应用与软件, 2008, 25(6): 30-32.
YAO Ronghua, ZHAO Qijun, LU Hongtao. Algorithm for Integrity Protection for XML Documents Based on Watermark [J]. *Computer Applications and Software*, 2008, 25(6): 30-32.

[14] CHEN L, HE W, SHU H, et al. Research on the Method of Text Information Hiding Based on XML [J]. *Applied Mechanics and Materials*, 2013, 385/386: 1665-1668.

[15] 斯琴, 张力, 廉德亮. 基于文本特征的文本水印算法[J]. 计算机应用, 2009, 29(9): 2348-2350.
SI Qin, ZHANG Li, LIAN Deliang. Text Watermarking Based on Text Feature[J]. *Journal of Computer Applications*, 2009, 29(9): 2348-2350.

[16] AL-WESABI F N, ALROWAIS F, MOHAMED H G, et al. Heuristic Optimization Algorithm Based Watermarking on Content Authentication and Tampering Detection for English Text[J]. *IEEE Access*, 2023, 11: 86104-86111.

[17] 李兆璨, 王利明, 葛思江, 等. 基于正交编码的大数据纯文本水印方法[J]. 计算机科学, 2019, 46(12): 148-154.
LI Zhaocan, WANG Liming, GE Sijiang, et al. Big Data Plain Text Watermarking Based on Orthogonal Coding[J]. *Computer Science*, 2019, 46(12): 148-154.

[18] 陈旖旎, 李千目, 吕超贤, 等. 不可见字符的文本安全隐藏算法研究[J]. 网络空间安全, 2019, 10(5): 88-96.
CHEN Yini, LI Qianmu, LV Chaoxian, et al. Research on Text Security Hiding Algorithm for Invisible Characters[J]. *Cyberspace Security*, 2019, 10(5): 88-96.

[19] 彭登. 基于控制符编码的网页链接篡改定位算法[D]. 成都: 西南交通大学, 2015.
PENG Deng. Webpage Link Tamper Localization Algorithm Based on Control Character Encoding [D]. Chengdu: Southwest Jiaotong University, 2015.

[20] XIAO C, ZHANG C, ZHENG C X. FontCode: Embedding Information in Text Documents Using Glyph Perturbation [J]. *ACM Transactions on Graphics*, 2018, 37(2): 1-16.

[21] QI W F, GUO W, ZHANG T, et al. Robust Authentication for Paper-Based Text Documents Based on Text Watermarking Technology[J]. *Mathematical Biosciences and Engineering*, 2019, 16(4): 2233-2249.

[22] 孙杉, 张卫明, 方涵, 等. 中文水印字库的自动生成方法[J]. 中国图象图形学报, 2022, 27(1): 262-276.
SUN Shan, ZHANG Weiming, FANG Han, et al. Automatic Generation of Chinese Document Watermarking Fonts[J]. *Journal of Image and Graphics*, 2022, 27(1): 262-276.

[23] 姚晔, 刘书辉, 王慧, 等. 基于字符扰动变形和字库替换的鲁棒中文文本水印[J]. 密码学报, 2023, 10(4): 769-785.
YAO Ye, LIU Shuhui, WANG Hui, et al. Robust Chinese Text Watermarking Method Based on Chinese Character Glyph Perturbation and Font Replacing [J]. *Journal of Cryptologic Research*, 2023, 10(4): 769-785.