

基于 Web 的管廊可视化信息管理系统设计与实现

陈登峰¹, 张温², 耿建勤², 刘国², 肖海燕²

(1. 西安建筑科技大学 建筑设备科学与工程学院, 西安 710055;

2. 西安建筑科技大学 信息与控制工程学院, 西安 710055)

摘要: 城市化进程促进了城市地下综合管廊的飞速发展, 但目前管廊的巡检等运维管理工作大多以人工为主, 且信息化水平低下, 而且管廊内的管线复杂、人工运维管理效率低且危险隐患高; 文章提出一种基于 Web 的管廊建筑结构与建筑信息轻量化可视化、CO 环境数据可通过浏览器在线训练与预测的系统; 通过 Revit 二次开发技术将建筑信息模型轻量化解耦并通过 WebGL 技术将几何模型加载在浏览器前端页面, 后台服务程序可以通过通信线程池将终端节点采集到的环境数据存入 Mysql 数据库, 并与浏览器前端动态耦合; 通过整合 Tensorflow.js 可以在浏览器前端实现基于 LSTM 算法的 CO 浓度预测模型在线训练与预测; 经过测试, 模型加载平均耗时 114.24 ms, 单次交互平均耗时 20 ms, 预测精确度为 86.3%, 满足系统设计需求。

关键词: 综合管廊; Web; 轻量化可视化; Tensorflow.js

Design and Implementation of Utility Tunnel Visual Information Management System Based on Web

Chen Dengfeng¹, Zhang Wen², Geng Jianqin², Liu Guo², Xiao Haiyan²

(1. School of Building Equipment Science and Engineering, Xi'an University of Architecture and Technology, Xi'an 710055, China;

2. School of Information and Control Engineering, Xi'an University of Architecture and Technology, Xi'an 710055, China)

Abstract: Urbanization promoted the rapid development of utility tunnel. However, there are some problems in the management of utility tunnel currently such as management mean depends on manual, the complicated pipelines in utility tunnel, the low efficiency of manual management and the high risk factors. The article puts forward a web-based utility tunnel visualization information management system based on web to realize the lightweight visualization of building structure and building information, online prediction and model training of carbon monoxide (CO) environmental data in web browser. Management system uses the secondary development of Revit to decouple the BIM model and loads the geometric model in the web page by WebGL. The background services put the environment data which is collected by the terminal nodes to Mysql database through the communication thread pool, and dynamically couple with the browser web page. Integrating Tensorflow.js to implement online training and prediction of CO based on long short-term memory (LSTM) in the browser. According to the test, the loading process of model takes 114.24 ms on average, the average time of a single interaction is 20 ms, and the accuracy of prediction is 86.3%. The management system meets the design requirements.

Keywords: utility tunnel; web; lightweight visualization; Tensorflow.js

0 引言

随着我国经济的快速发展与城市化的高速推进,“马路拉链”、“蛛网线路”等“城市病”不断加剧,城市地下综合管廊(简称管廊)是重要解决手段之一^[1]。管廊是位于城市地下的一个相对密闭的空间,在其运营过程中,会产生如 CH₄、CO 等有害与易燃气体^[2],当 CO 气体吸入过量,轻则使人头晕恶心,重则会导致人死亡,故对管廊舱室 CO 指标实时监测与预测十分重要,目前,国内外综合管廊运营维护中的巡检等工作还以人工为主^[3]。现有的将 BIM(建筑信息建模技术)用于管廊管理的可视化系统大多 C/S

架构,由于模型中所包含数据量较大、数据结构较复杂且专业性较强^[4],导致系统操作难度大、硬件要求高、不可跨平台且缺少安全性预测功能。

文章提出基于 Web 的管廊可视化信息管理系统,管廊运营者可通过移动端、电脑端浏览器以 3D 模型漫游管廊、查看管廊建筑数据、监测实时环境数据,并根据 CO 环境数据预测值、实时环境数据以及管廊轻量化实现危险早期处理。

1 整体设计

1.1 功能分析

管廊运营维护过程中,运维人员需通过不同终端进行管廊信息可视化查阅和管廊环境数据实时监测,当监测或预测到异常需快速定位故障舱室,并辅助给出解决方案。基于需求确定系统采用 B/S 架构^[5],并满足以下功能需求:

1) BIM 轻量化:将 BIM 几何信息与建筑信息解耦,并将所需建筑信息存入数据库;2) 模型加载:将 BIM 几何

收稿日期:2019-10-16; 修回日期:2019-11-01。

基金项目:国家自然科学基金项目(51705393),陕西省教育厅专项科研项目(14JK1408)。

作者简介:陈登峰(1976-),男,陕西西安人,博士,副教授,主要从事建筑环境监测、工业过程节能控制等领域的教学与科研工作方向的研究。

信息可视化在 Web 页面并与建筑信息以及实时环境数据动态耦合; 3) 环境数据采集: 系统后台服务与终端节点建立通信并将其采集的环境数据存入数据库; 4) 实时数据展示: 将终端节点采集的环境数据以交互形式展示在浏览器前端模型及实时动态曲线; 5) 数据管理: 历史环境数据查询、终端节点属性信息查询与更新; 6) 环境数据预测: 以 CO 历史数据作为训练样本, 通过浏览器在线训练 LSTM 预测模型并对递增 2 小时 CO 数据在线预测并显示和预警。

1.2 设计方案

系统中使用管廊 BIM 模型 Revit 搭建。系统包括终端节点、服务软件和前台软件组成, 如图 1 所示。

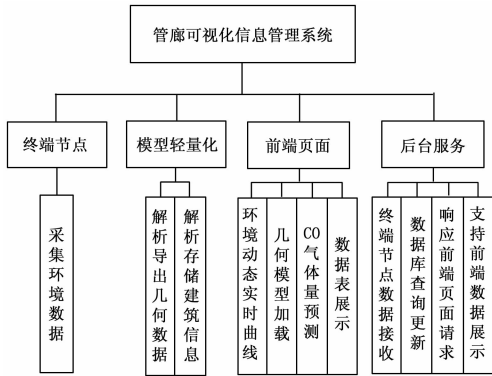


图 1 系统总体功能图

终端节点由传感器、控制器、无线模块组成, 部署在管廊舱室中。终端节点采用 TCP 通信方式, 将采集的环境数据实时发送给服务软件并存入数据库中。

模型轻量化模块将 BIM 模型解耦为几何信息与建筑信息。该部分由 C# 整合 Revit 二次开发实现, 通过 Revit 的建筑模型 Id 建立管廊子模型与建筑信息的对应关系。

数据库模块分为数据视图模块与存储过程模块, 其中数据视图模块用于建立数据库表间逻辑关系, 存储过程模块用于每日定时对数据库实现特定功能。

后台服务模块为该系统提供整体依托, 提供对数据库查询与更新、对终端节点采集的环境数据信息的获取与存储、响应前端页面请求、对前端的数据以及其他服务支撑。

前端页面模块由模型加载模块、动态刷新模块以及其他 Web 元素构成。模型加载模块使用 WebGL 与 Html5 实现^[6-7], 动态刷新模块使用 Ajax 技术与后台服务模块交互获取实时环境数据。气体数据预测模块采用 Tensorflow.js 框架实现回归预测, 结合 Javascript 实现浏览器端对环境数据模型训练与在线预测。通过浏览器在线训练 LSTM (长短期记忆网络) CO (一氧化碳) 预测模型实现对递增时间节点指标数据的在线预测。当监测或预测到某舱室异常, 可向管廊运营者进行危险预警, 给出异常状况辅助解决方案, 以减轻或避免危险的发生。

2 详细设计与实现

2.1 建筑模型轻量化

BIM 在建筑几何模型上附带有施工过程的大量静态属

性信息, 导致存储空间较大。如在 Revit 中绘制 6 面墙, 生成的项目文件大小为 4.47 MB, 而几何数据仅占 12.5 KB。故需将建筑模型加载在前端页面, 轻量化是必不可少的, 系统使用文件为 Obj 文件, 文件结构见表 1。

表 1 Obj 文件结构表

名称	描述
v 行	几何模型顶点坐标
vt 行	模型贴图坐标
vn 行	模型顶点法线
g 行	模型组名称, 数个面为一组
f 行	模型面信息由点序号构成

使用 C# 与 Revit 二次开发 API 实现模型轻量化导出。结合 Revit 二次开发 API 解析建筑模型 Rvt 文件, 将各 Id 对应几何数据以 Obj 文件中 g 标识分割, 通过文件流写入 Obj 文件。

将属性信息通过动态链接库 MySQL.Data.dll 以 Id 为索引写入 Mysql 数据库。目的是实现将建筑几何数据与建筑属性信息解耦以达到轻量化的效果。其中 g 中包含 Id 信息, g 表示以下的几何数据信息为该 Id 对应的建筑模型几何数据。

2.2 环境数据预测模块

由于不同舱室气体环境有异, 且舱室环境数目较多, 为提高预测数据精度减轻服务端压力, 设计一种基于浏览器的分布式 CO 预测模块。

该模块将传统基于 PC 服务端的模型训练与预测移植到 Web 客户端, 采用 google 近期推出的 tensorflow.js 框架^[8], 训练与数据预测的功能由浏览器支持实现而无需后台支持, 也不需要复杂的环境部署与库文件调用过程, 只需使用移动端或电脑端浏览器即可实现对环境数据的训练与预测, 将训练和预测的流程由服务端移植到客户端。不同管廊舱室可针对每个舱室的历史数据给出个性化的预测结果, 管廊内部环境密闭, 其中 CO 含量实际是等间隔连续的时间序列, LSTM (长短期记忆神经网络) 神经网络对时间序列数据预测准确性较高^[9], 本模块采用 LSTM 对 CO 数据进行预测。在线模型训练流程如图 2 所示。

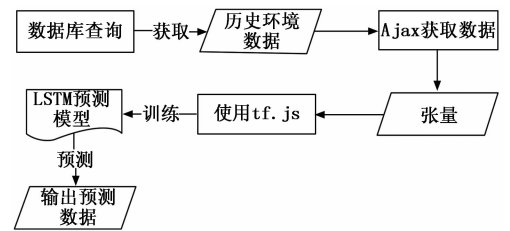


图 2 在线模型训练流程图

基于浏览器的分布式 CO 预测模块运行流程描述如下。

步骤一: 登录系统选择某终端节点历史表信息点击查询, 查询结果会作为样本数据以 Json 形式存储在公共变量中;

步骤二: 将样本数据:

$$X = \{x_1, x_2, \dots, x_n\}$$

整合并做标准化操作, 计算数据顺序涨幅百分比:

$$X' = \left\{ \frac{x_2 - x_1}{x_1}, \frac{x_3 - x_2}{x_2}, \dots, \frac{x_n - x_{n-1}}{x_{n-1}} \right\}$$

步骤三: 使用滑动窗口法初始化数据, 构建训练数据, 设置输入维度, 滑动步数, 以张量存储; 其中张量 Tensor 是一维或多维数组的结构, 是 Tensorflow.js 的数据中心单位。

步骤四: 设置少量训练数据来作为测试数据对已训练模型进行验证, 设置过程如下所示:

```
await model.fit(xs, ys, {validationSplit: 0.2});
```

设置参数为 0.2, 即 80% 数据用来训练模型, 剩余 20% 数据用来对模型进行测试验证。

步骤五: 优化算法选择 RMSProp, 损失函数使用 MSE (均方误差)。设当前收敛过程中迭代次数为 t , RMSProp 优化算法描述如下:

$$s_{dw} = \beta s_{dw} + (1 - \beta) dW^2$$

$$s_{db} = \beta s_{db} + (1 - \beta) db^2$$

$$W = W - \alpha \frac{dW}{\sqrt{s_{dw} + \epsilon}}$$

$$b = b - \alpha \frac{dW}{\sqrt{s_{db} + \epsilon}}$$

其中: s_{dw} 、 s_{db} 为上一轮迭代过程梯度动量, β 为梯度累计指数, W 权重值, b 为偏置值, ϵ 取值 10^{-8} 防止分母为 0, 发生异常。损失函数描述如下:

$$MSE = \frac{1}{M} \sum_{m=1}^M (y_m - \hat{y}_m)^2$$

其中: M 为测试样本总数, \hat{y}_m 为预测值, 该式表达真实值与预测值关系, 其值越大, 预测效果越差。系统对二者定义部分代码如下:

```
model.compile({optimizer: rmsprop,
loss: tf.losses.meanSquaredError});
```

步骤六: 训练 LSTM 模型并应用模型进行预测, 预测部分代码如下:

```
await model.predict(tf.tensor([input])).data();
```

步骤七: 整合使用 tfvis 可视化框架对样本数据、训练过程以及预测信息可视化, 如图 3 所示。

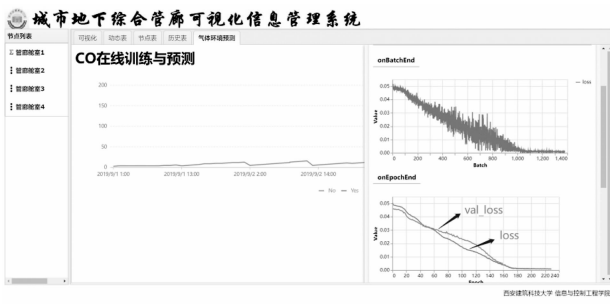


图 3 训练过程

其中 loss 指训练集损失, val_loss 指测试集损失。

2.3 数据库设计

设计的数据库包括用户信息表、终端节点信息表、环境数据表、历史信息表。

用户信息表用于支持用户登陆系统校验, 其列信息包括用户 id、用户名、密码、用户角色。用户在前端页面输入用户名、密码以及角色后, 该信息会与数据库中存储的用户信息进行比较, 如信息匹配, 则用户登陆成功, 页面展示对应角色的内容, 否则, 提示登录失败。

终端节点信息表用于对终端节点信息查询与更新, 其列信息包括终端节点 Id、终端节点名称、终端节点在实际综合管廊中所处的舱室信息、节点持续使用时长。管理员角色登陆系统后, 可对终端节点信息进行更新与查询, 其中更新操作包括增、删、改; 舱室位置为语言描述, 描述内容为该节点所处管廊的舱室位置; 使用时长为自动生成, 自终端节点投入使用后算起。

环境数据表用于存储终端节点采集的环境数据, 其列信息包括数据 Id、时间戳、传感器节点 Id、温度、湿度以及 O_2 、 CO 、 CH_4 、 H_2S 浓度, 结构如表 2 所示。

表 2 环境数据表结构

列名	类型	备注
Id	Int(11)	Id
SenId	Int(11)	节点 Id
TimeStamp	TimeStamp(12)	时间戳
Temp	Varchar(255)	温度
Humi	varchar(255)	湿度
O_2 浓度	Varchar(255)	氧气
CO 浓度	varchar(255)	一氧化碳
CH_4 浓度	varchar(255)	甲烷
H_2S 浓度	varchar(255)	硫化氢

SenId 与终端节点信息表的 Id 对应, 后台服务端接收终端节点采集到的环境数据存入该表中, 前端页面中实时动态曲线通过 Ajax 定时从数据库查询获取最新的数据。

历史信息表用于存储历史环境数据, 该数据表为气体环境预测模块提供训练样本, 同时可供系统使用者查看, 其列信息包括 Id、终端节点 Id、当日日期 (精确到小时)、平均温度、平均湿度、平均 O_2 含量、平均 CO 含量、平均 CH_4 含量、平均 H_2S 含量, 所有平均值均为每日每 1 小时的平均值。其内容更新由已编译的存储过程在每日零点定时进行。

2.4 系统前端

系统前端使用 JavaScript、CSS、Highcharts、EasyUI 与 Sim.js 设计与开发, 可实现建筑信息模型轻量可视化、环境数据实时曲线、终端节点管理与历史数据查询的功能。实时数据展示中实时动态曲线使用 Highcharts 结合 Ajax 局部刷新技术实现, 每次单位时间查询获取数据库中该终端节点采集的最新环境数据, 整体采用 setInterval 方法实现定时发送 Ajax 请求。在请求的过程中使用轻量级的 Json 作为数据传输的格式, 模拟面向对象的过程。系统使用 WebGL 绘图协议的开源框架 Three.js 整合 Sim.js 将三维模型 Obj 文件加载在前端页面中, 模型加载过程部分代码如下:

```
//创建 Obj 加载对象
var loader=new THREE.OBJMTLLoader();
loader.load('./model/pipe.obj', './model/pipe.mtl', function
(object){
//遍历 Obj 模型文件结构
object.traverse(function(obj){
//按照 Id 依次将模型对象加入场景中
that.createModel(obj,this.boxcount);});
this.focus();
this.count = 0; //模型计数器
this.createPlane();//创建地面}
```

其加载过程具体描述如下:

步骤一: 使用 ObjLoader.js 中的 Load 方法对 Obj 文件进行解析;

步骤二: 结合 traverse 方法, 对每组几何数据进行解析;

步骤三: 结合面向对象的思想, 将模型 Id 以及 Mesh 信息构建几何对象;

步骤四: 遍历上述对象, 将对象加载在 Scene 中, 最终通过 render 渲染在前端页面。

文件加载采用面向对象方法的目的是方便为每个子模型附加交互函数。

2.5 后台设计

后台服务为前端提供服务支持, 采用 Java 开发, 使用 SSM (Spring、SpringMVC 与 MyBatis) 整合框架, 应用 MVC 框架 (Model、View、Controller), 复杂逻辑实现于 Controller 控制器部分, View 指前端界面, Model 指数据的封装部分^[10]。该框架将前后台程序分块解耦, 增加了代码扩展性与复用性。

系统中, 后台服务模块需要具备以下功能: 1) 实时数据采集: 获取终端节点的实时环境数据并存入数据库; 2) 实时数据展示: 数据展示的方式可分为用户与管廊模型交互展示和实时动态曲线; 3) 数据查询: 可对历史数据以及终端节点数据进行查询; 4) 终端节点管理: 当更新管廊终端节点后, 同步更新传感器节点数据; 5) 角色控制: 控制不同的角色登陆系统并展示不同功能, 角色分为管理员与非管理员。

后台服务模块数据持久层采取 MyBatis, 它支持个性化 Sql 与存储过程, 该框架将 JDBC 细节掩盖, 将数据库配置集成于 xml 文件。

出于安全性与数据完整性的考虑, 系统中的后台服务与终端节点采用 TCP/IP 通信协议交互。终端节点为通信客户端。由于管廊中终端节点数量众多, 故在后台构建 TCP 服务端线程池, 可以节省大量资源。线程池与终端节点交互并将终端节点 Id 以及采集到的环境数据存入环境数据表。

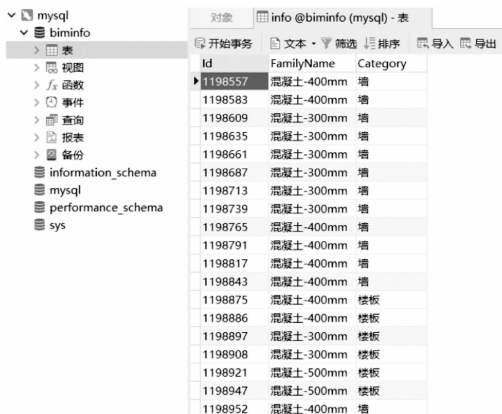
3 实现效果

管廊可视化信息管理系统功能主要包括管廊场景漫游与交互、数据库表维护以及基于 Web 的在线预测。

系统运行环境为 Windows10、Cpu i5-7500、显卡 GTX-1060、内存 8 G, 系统开发语言为 Java, Ide 为 Myeclipse10, 数据库为 Mysql, 数据库可视化工具选择 Navicat Premium 12, 后台服务使用 SSM 整合框架, 前端使用 jquery.js、tensorflow.js、hightcharts.js、three.js 以及 sim.js 等开发实现, 项目依托 tomcat7 服务器部署。

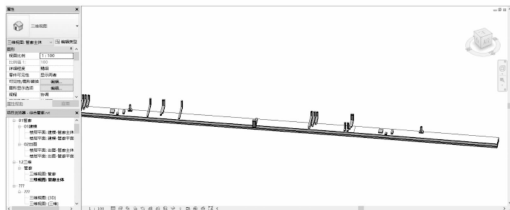
3.1 轻量可视化

管廊 BIM 模型轻量化解耦以及可视化页面如图 4 所示, 其中 (a) 为二次开发导出并存储在 mysql 中的建筑信息, (b) 为 revit 中绘制的模型, (c) 为轻量化后加载在前端页面的效果图。

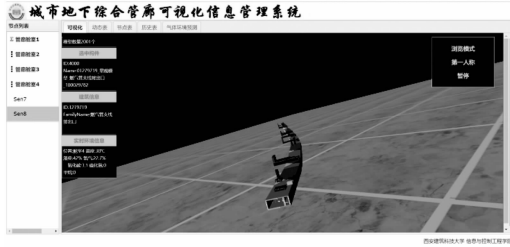


Id	FamilyName	Category
1198557	混凝土-400mm	墙
1198583	混凝土-400mm	墙
1198609	混凝土-300mm	墙
1198635	混凝土-300mm	墙
1198661	混凝土-300mm	墙
1198687	混凝土-300mm	墙
1198713	混凝土-300mm	墙
1198739	混凝土-300mm	墙
1198765	混凝土-400mm	墙
1198791	混凝土-400mm	墙
1198817	混凝土-400mm	墙
1198843	混凝土-400mm	墙
1198875	混凝土-400mm	楼板
1198886	混凝土-400mm	楼板
1198897	混凝土-300mm	楼板
1198908	混凝土-300mm	楼板
1198921	混凝土-500mm	楼板
1198947	混凝土-500mm	楼板
1198952	混凝土-400mm	墙

(a) 导入 mysql 的建筑信息



(b) Revit 绘制的 BIM 模型



(c) 前端加载效果图

图 4 轻量可视化

前端页面加载轻量化后的管廊场景, 可对场景漫游。当点击左侧菜单中的终端节点后会将相机视角调整至模型中对应终端节点的位置, 主页面左侧会将节点附近的环境信息以及建筑属性信息显示出来。用户也可点击右上角菜单选择场景浏览模式, 可进行鼠标查看场景或第一人称漫游场景, 用户与场景中的建筑构件交互, 会将模型构建信息展示在左侧。

3.2 环境数据曲线

环境数据实时动态曲线如图 5 所示。该页面主要功能

为将终端节点采集的环境数据以动态曲线展示，根据左侧菜单选择终端节点，主页面将展示该终端节点监测到的实时数据，数据每十秒刷新一次。

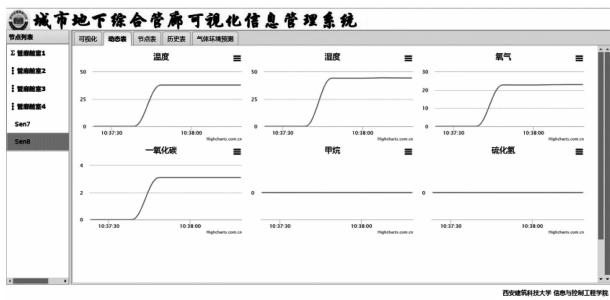


图 5 环境数据实时动态曲线

3.3 数据库维护

数据库维护包含对终端节点表、历史信息表的查询与更新。

终端节点表对建筑中的终端节点进行管理，如在实际建筑中更新（增、删、改）节点，则管理员角色对该表进行更新，同时左侧菜单栏也会相应更新。

历史信息表存储每日的历史信息，可供管廊运维人员查看，同时提供数据用于 Tensorflow.js 模型训练。每日零点，预先编译的存储过程会定时将每日平均两小时环境数据进行均值处理，处理结果存入历史信息表中，由于历史信息表数据量不会很大，故该表仅提供查询功能，可根据终端节点、起止日期对表进行查询。

3.4 基于 Web 的在线预测

在线预测功能将历史信息表 4 天内每小时 CO 数据的均值作为训练样本，以实验数据为例，最终预测结果如图 6 所示。

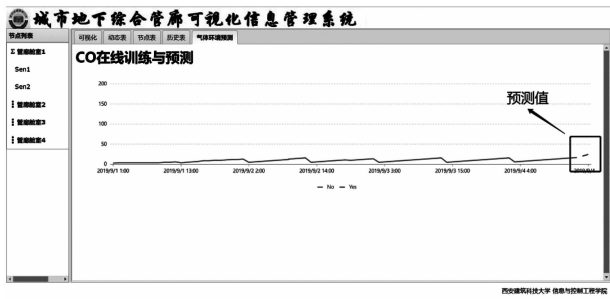


图 6 预测结果图

4 结论

文章设计实现了一种用于管廊运营维护的信息化系统，具备以下特点与功能：

1) 结合 Revit 二次开发将 BIM 模型几何与属性信息解耦，轻量化加载在 Web 页面中，结合 Spring、SpringMVC、MyBatis，实现管廊轻量化展示、管廊交互漫游、环境数据动态更新以及建筑信息管理，基于 B/S 模式开发，具有较强的跨平台性，可以使用计算机或移动端浏览器。

2) 环境数据实时显示，通过交互的方式展示在前端加载的模型中，也可动态实时展示在曲线中；

3) CO 数据预测性，基于 LSTM 对 CO 数据在线预测，不同舱室运营者可分布式在客户端浏览训练并预测不同舱室 CO 数据指标，提高了预测的精度，减轻了服务端的压力，将模型训练与预测功能由服务端分布实现在客户端。同时也很大程度上提高了系统的实用性，安全性预警可以减少不必要的人员与设备损坏，提高了管廊运维的安全性与稳定性。

参考文献：

[1] 向春玲. 中国城镇化进程中的“城市病”及其治理 [J]. 新疆师范大学学报(哲学社会科学版), 2014, 35 (2): 45-53.

[2] 陈登峰, 赵 婷, 肖海燕. 综合管廊环境安全性模糊综合评价研究 [J]. 地下空间与工程学报, 2018, 14 (S2): 906-912.

[3] 李 芊, 许高强, 韦海民. 基于 BIM 的综合管廊运维管理系统研究 [J]. 地下空间与工程学报, 2018, 14 (2): 287-292.

[4] 刘 佳, 张庆彬, 梁秋丽. BIM 模型的轻量化展示平台研究 [J]. 建筑技术, 2019, 50 (7): 791-793.

[5] 吴晓珊, 曹旭东, 王森, 魏文龙. 基于 B/S 架构的管理系统软件开发 [J]. 计算机测量与控制, 2019, 27 (2): 123-128.

[6] 宁 静, 卜乐平, 冯 源. 基于 WebGL 的舰船模拟训练虚拟三维技术应用 [J]. 计算机测量与控制, 2016, 24 (9): 251-253.

[7] 马昊好, 李 平, 周 启, 等. WebGL 在线动态地图服务框架设计 [J]. 测绘通报, 2019 (1): 118-122.

[8] Huang C Y, Liu L, Chen Y L. An Online Integrated Fingerprint Image System [J]. International Journal of Machine Learning and Computing, 2019, 9 (1): 51-56.

[9] 段大高, 赵振东, 梁少虎, 等. 基于 LSTM 的 PM2.5 浓度预测模型 [J]. 计算机测量与控制, 2019, 27 (3): 215-219.

[10] 张 时, 王向东, 李树江. 在线油烟实时监测系统的设计与实现 [J]. 计算机测量与控制, 2019, 27 (8): 35-39.

(上接第 169 页)

[8] 裴家正, 黄 勇, 董云龙, 等. 杂波背景下基于概率假设密度的辅助粒子滤波检测前跟踪改进算法 [J]. 雷达学报, 2019, 5 (3): 355-365.

[9] 孙 云, 王国宏, 谭顺成, 等. 基于辅助粒子滤波的机动弱目标 TBD 算法 [J]. 电光与控制, 2013, 30 (7): 28-31, 92.

[10] 李海君, 赵国荣. 基于快速高斯变换的辅助边缘粒子滤波算法 [J]. 数据采集与处理, 2014, 29 (6): 998-1002.

[11] 王爱民. 制造系统工程 [M]. 北京: 北京理工大学出版社, 2017.

[12] 周卫琪, 齐 翔, 陈 龙, 等. 基于无迹卡尔曼滤波与遗传算法相结合的车辆状态估计 [J]. 汽车工程, 2019, 41 (2): 188-190, 205.

[13] 张 民, 贾海涛, 沈 震. 基于遗传算法改进的粒子滤波重采样模型 [J]. 电子科技大学学报, 2015, 44 (3): 344-349.