

# CHATTWIN: ENABLING NATURAL LANGUAGE INTERACTIONS WITH INFRASTRUCTURE DIGITAL TWINS

Peihang Luo, Erika Parn, and Ioannis Brilakis  
University of Cambridge, Cambridge, UK

## Abstract

Infrastructure lifecycle management requires interactions with dynamic datasets. Traditional interfaces often hinder users' ability to rapidly locate lifecycle-specific information they need. Our work proposes ChatTwin, a system that employs Large Language Models (LLMs) to enable natural language queries related to various lifecycle stages of infrastructure, with a focus on operations and maintenance. Simulated scenarios were constructed to test the system. The results demonstrate that the system can effectively categorise interactions, fetch relevant information, and produce human-friendly outputs. With this LLM-based approach, we present an improvement in user-centricity in infrastructure lifecycle management, streamlining interactions and decision-making throughout the entire infrastructure lifecycle.

## Introduction

Modern infrastructure systems are becoming increasingly complex and interconnected, posing significant challenges in their lifecycle management (Grafius et al., 2020). Decisions made throughout their lifecycles require consideration of large amounts of data from various sources. Digital twins have emerged as promising tools to overcome these challenges. They are virtual replicas of physical entities or systems, offering integrated data platforms that enable stakeholders to gain an enriched understanding of infrastructure's behaviour, to optimise maintenance and operation activities, and more (Broo et al., 2022; Rudskoy et al., 2021). Digital twins typically consist of three elements: a physical space where the actual object resides, a digital space that hosts the virtual model, and a connection that facilitates synchronisation and real-time feedback between the two spaces (Barricelli et al., 2019). However, the current human-information interaction paradigms with digital twins remain unintuitive, leading to a steep learning curve and significant investment in training. UK employers invest around £42 billion in training their employees each year, equivalent to an average spend of £2,540 per trainee (Winterbotham et al., 2020). However, even after extensive training, employees can still find it difficult and time-consuming to find the information they need, with employees spending approximately 9.3 hours weekly just sourcing information (Chui et al., 2012). Therefore, the need for more accessible, intuitive interaction methods is clear.

## Background

### Human-information interaction with digital twins

The interaction between users and digital twins plays a crucial role in leveraging the full potential of these complex and dynamic digital twins. Despite their potential, the current interaction modes between users and digital twins, particularly the mode of natural language interaction, remain underdeveloped. For example, Pairet et al. (2019) present a natural language interface designed for interactions with a digital twin of an offshore platform, facilitating user training across various human-robot collaboration scenarios, but without any results or implementation details. Dingli and Haddod (2019) propose a human interface of an Intelligent Digital Twin system for a semiconductor manufacturing plant, utilising three different modalities: hand gestures, a VR controller, and a voice interaction system. However, no results have been obtained for the performance of the system. Siyaev et al. (2023) introduce a neuro-symbolic reasoning mechanism to interact with a virtual aircraft maintenance digital twin constructed from structured manuals using natural language, which requires a special symbolic vocabulary and a neuro-symbolic dataset of queries to work. Furthermore, to the best of the authors' knowledge, there is no work on the natural language interactions with infrastructure digital twins.

### Prompting pre-trained large language models

By enabling users to interact with computer systems using their native language, natural language interactions offer a seamless and accessible interaction experience, particularly for users with limited technical expertise or background knowledge in computing (Pazos R. et al., 2013). Recent advancements in natural language processing, especially the emergence of LLMs, have made it possible to develop more sophisticated and context-aware natural language interaction systems (Wei et al., 2022). Unlike traditional language models, which need to be fine-tuned or even re-trained on specific datasets tailored for individual tasks, LLMs have demonstrated strong in-context learning abilities. This means that they can effectively adapt to new scenarios and perform different tasks by only including a few task-specific input-output examples in the input prompt. This innovative technique is known as "few-shot prompting" (Brown et al., 2020). With few-shot prompt-

ing, LLMs can potentially be used for diverse natural language interaction tasks without the need for task-specific models or training datasets. Moreover, LLMs have also demonstrated the ability to perform tasks through “one-shot learning” or even “zero-shot learning” (Kojima et al., 2023). The former involves the LLM accomplishing tasks with just a single input-output example. The latter refers to LLMs handling specific tasks with no examples provided, relying solely on task descriptions in natural language.

While few-shot prompting, one-shot prompting, and even zero-shot prompting have demonstrated competitive performance on various NLP tasks like question-answering or translation (Brown et al., 2020), it has been observed that for complex reasoning and numerical tasks, LLMs may not always provide accurate answers. A widely adopted approach to address this challenge is “chain-of-thought” prompting. This approach provides chain-of-thought demonstrations in prompts to guide the LLM in generating a series of intermediate reasoning steps for a given task. This technique has proven to significantly improve LLM performance on tasks involving intricate reasoning or numerical skills (Wei et al., 2023).

To further enhance the performance of LLMs, some more advanced prompting techniques have also been proposed. For example, “Program-Aided Language models” (PAL) improve LLM performance in arithmetic and symbolic reasoning tasks (Gao et al., 2023). PAL employs the LLM to read and understand natural language problems, generating programs as intermediate reasoning steps based on the problem description. Unlike other methods, PAL offloads the solution step to a runtime, such as a Python interpreter. This allows the LLM to focus solely on decomposing the problem into executable steps.

### Gaps in knowledge, objectives & research questions

The gaps in knowledge can be summarised as follows:

1. **Domain-Specific Knowledge:** LLMs lack specialised domain knowledge related to infrastructure digital twins and infrastructure lifecycle management. This includes domain-specific processes, the specialised knowledge and contextual understanding required to effectively communicate with and interpret information from digital twins, and more.
2. **Contextual Information Retrieval:** Integrating LLMs with digital twins requires retrieval of relevant information from the digital twin itself. LLMs must be equipped to comprehend the underlying data structure and extract meaningful information and insights from the digital twin’s vast repository of information.
3. **Effective Prompt Structures:** The structuring of prompts plays a pivotal role in guiding LLMs to generate accurate and useful outputs. Understanding how to design prompts that effectively convey the context of a specific query is still an open question to be an-

swered.

4. **Usability and Interpretability:** Ensuring that LLM-driven interactions with digital twins are not only accurate but also interpretable is critical. Gaining insights into how to post-process LLM outputs to extract valuable information in a user-friendly format is another key aspect to explore.

The primary objective of this project is to enhance human-information interaction with infrastructure digital twins by leveraging LLMs for natural language interactions, contributing to the development of more efficient, accurate, and domain-aware LLM-driven interactions with infrastructure digital twins. With this primary objective in mind, the subsequent research questions have been formulated:

1. What specific techniques can be employed to retrieve relevant information from the digital twin?
2. How can prompts be optimised to ensure that LLMs generate outputs that not only align with the specific requirements of the stakeholder but also maintain a consistent level of accuracy? Moreover, how can the retrieved data be presented to the LLMs within the prompts, ensuring that the LLM understands the broader scenario, background, or situation in which the information is embedded, leading to accurate and context-aware responses?
3. What specific procedures or tools can be employed to most effectively present data or insights to generate tangible value for infrastructure management stakeholders?

## Proposed Solution

### Scope

This paper focuses on investigating how LLMs can enhance interactions with infrastructure digital twins, enabling them to offer information and insights, and perform certain actions on behalf of users via natural language interactions. This paper prioritises the following five common interaction tasks, using a road digital twin as a case study. Road digital twins are comprehensive virtual models of road systems that replicate their physical counterparts in real-time. These models integrate various data sources to create a dynamic simulation of the road network. This allows for enhanced real-time monitoring, predictive maintenance, and strategic planning (Marai et al., 2021). Although our solution is demonstrated on road networks, it is designed to be adaptable to other infrastructure systems with minimal modifications.

#### *Task 1: Data visualisation*

One of the primary functions of road digital twins is integrating data from multiple sources, such as time-series data from meteorological and traffic sensors. Our first

goal is to enable users to generate and view visualisations of such time-series data through natural language interactions, enabling them to gain a more intuitive view of the complex information patterns within such data.

*Task 2: Information summarisation*

Navigating through the vast datasets of a road digital twin to extract pertinent information is often cumbersome and time-consuming. Our system aims to simplify this by summarising key data, like road defect details, in response to user queries in natural language.

*Task 3: Performing changes*

A frequent and critical interaction with road digital twins is the implementation of changes to the properties of specific instances. This includes, for example, changing the status of a defect from “scheduled to fix” to “fixed” after a repair with a natural language prompt.

*Task 4: Model visualisation*

Accessing and visualising models, such as point clouds, is another routine task for road digital twin users. This typically involves locating specific files and using dedicated tools. We propose to simplify this process using natural language interaction, enabling users to request visualisations of model sections through simple prompts, enhancing user experience and information access efficiency.

*Task 5: Work schedule enquiry*

Another commonly performed task for infrastructure digital twins is generating and visualising work schedules. Our work aims to automate these tasks, traditionally a manual and laborious task, through natural language interactions. This automation will facilitate more frequent and accurate schedule adjustments and updates, improving productivity and reducing unnecessary costs associated with outdated schedules.

However, due to the data availability of the demonstrative digital twin, only summarisations of defect information are included in the scope of *Task 2*, and the scope of *Task 5* is only limited to inquiries concerning work schedules of defect rectifications. In addition, this work limits its scope of *Task 3* only to property modifications due to their regularity and natural language compatibility. More complex modifications, such as geometrical alterations or adding new instances, often require assistance from specialised equipment like laser scanners and are thus excluded.

**Details**

This section outlines the step-by-step process of the proposed solution that enables the user to interact with the

infrastructure digital twin using natural language prompts.

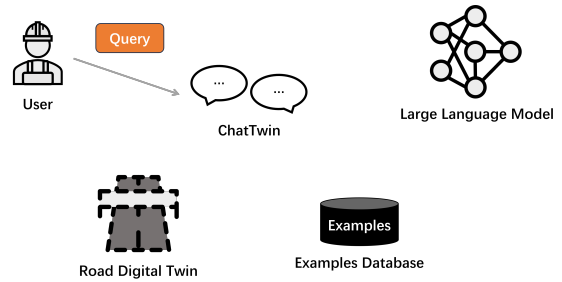


Figure 1: Step 1 - User initiating request

In step 1, as illustrated in Figure 1 above, the user initiates the interaction process by providing a natural language prompt that describes their query or request. Such prompts could either instruct the system to perform specific actions on the digital twin, or request specific data visualisations or summaries.

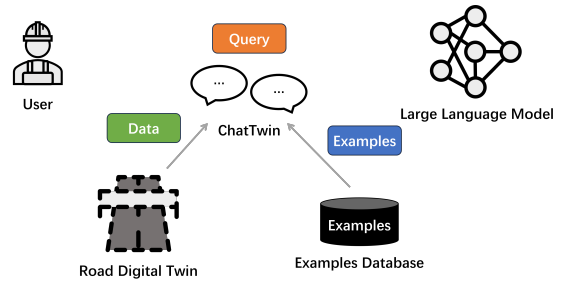


Figure 2: Step 2 - Requesting and receiving relevant data from the Road Digital Twin & Examples Database

In step 2, upon receiving the user’s input, the system determines which task category this user prompt belongs to. This categorisation result will determine the subsequent actions of the system. This is done by sending a pre-structured prompt to the LLM. The structure of this prompt can be found in appendix “Prompt for categorising tasks”. Depending on the task category, the system fetches relevant data from the digital twin and the examples database (as shown in Figure 2 above). Each task category dictates a unique retrieval process. Further details on the data retrieval process can be found in the “Results & findings summary” section. This aims to solve the first research question, so that relevant data can be correctly retrieved from the digital twin for each of the task categories.

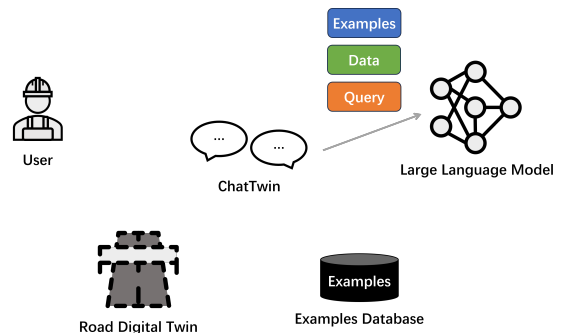


Figure 3: Step 3 - Pre-processing and sending modified prompt to LLM

In step 3 (as shown in Figure 3 above), the relevant data and examples retrieved from the digital twin and the examples database are combined with the user prompt to generate a modified prompt, which is then sent to the LLM. The modified prompt is generated by collating the user’s original prompt, the relevant data from the digital twin (if required), and the relevant few-shot training examples from the examples database into a structured prompt. An illustrative example is provided in Figure 8 in the appendix. The prompt starts with one or a few few-shot training example(s) retrieved from the examples database. Each example consists of the relevant data retrieved from the digital twin (if applicable), the user prompt, and the expected task-specific output from the LLM. Additional examples are appended to the end of the previous ones. The second part of the prompt is the relevant data retrieved from the digital twin (if required), and the final part of the prompt is the user’s original prompt. The modified prompt is then sent as input into the LLM. This aims to solve the second research question. The examples from the examples database provide the LLM with essential information about the data it works with. This includes but is not limited to the nature of the data, its structure, and the methods to access it. By illustrating this with sample outputs for typical user query inputs, the LLM gains a better understanding of both the data and its context, enabling it to produce more accurate results. In addition, the examples also act as guides for the LLM to ensure that its responses meet the specific needs and standards of different users and situations. This helps in maintaining the relevance and accuracy of the LLM’s outputs.

One exception is that for model visualisation tasks, a distinct prompt structure is used to determine whether the requested point cloud data is available within the digital twin, and retrieve the requested point cloud data file if it is available. This prompt structure can be found in appendix “Prompt to retrieve point cloud data”.

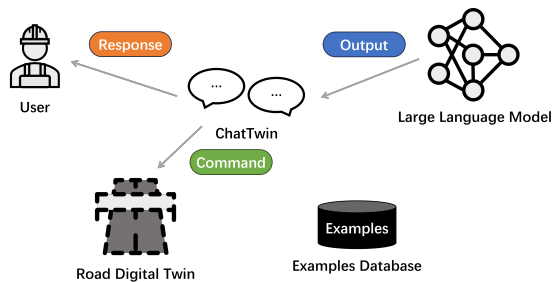


Figure 4: Step 4 - Receiving and post-processing output from LLM

In step 4, as shown in Figure 4 above, an output is received from the LLM, which contains either actionable commands for the digital twin, machine-readable information on the requested information, or a natural language summary of the information requested by the user.

The commands are generated in the form of Python scripts with embedded comments that contain chain-of-thought

information. The inclusion of comments helps improve the quality and readability of the generated commands. For tasks where the system is asked to perform specific changes to the digital twin, the LLM will also generate natural language feedback to the user based on the result of the execution of the Python script. For example, if the Python script is successfully executed without errors, the LLM will then provide the user with a “success” message, confirming that the requested actions have been successfully applied to the digital twin.

Otherwise, for example, for information summarisation tasks, the summary generated by the LLM is directly sent to the user, or for model visualisation tasks, the output from the LLM is first analysed - if the output is “Not available”, the system will output a message indicating that the requested data is not available within the digital twin, or if the output is the name of the requested data file, the name of the file will be passed into a pre-structured Python script to visualise the requested data. This pre-structured script can be found in appendix “Python script to visualise point cloud data”.

This step aims to solve the third research question, so that the data retrieved from the digital twin can be presented to the user in an appropriate form using different tools and procedures, enabling users to quickly gain more value and insights from the data.

## Research Methodology

Our work constructs a demonstrative road digital twin to evaluate the proposed interaction pipeline. This twin is populated with four data types: geometric data in point cloud format, time-series sensor data, defect information, and recommended maintenance actions.

### Data collection and preparation

The geometric point cloud data, representing a specific road junction in Tallinn, is sourced from the Tallinn City Digital Twin. Time-series sensor data, simulating the input from infrastructure sensors, is obtained from the UK Environment Agency’s Real Time flood-monitoring API. This API provides real-time and historical water level and flow measurement data at each of the monitoring stations across the UK. This data is indicative of the real-time sensor data one would expect in a fully operational digital twin. These sensor data have been integrated with the point cloud data, maintaining their properties such as names, IDs, and typical ranges, but changing their location-related properties to correspond with the geospatial location of the point cloud data, so that the sensors are essentially “embedded” into the point cloud at certain locations.

The defect information and suggested actions are stored in text format in a .txt file. The information stored within

the digital twin about each registered defect includes defect type (e.g. pothole, crack, etc.), defect location (shown as a kilometre number along a specified road), severity level (either severe, moderate, or low), and suggested action based on the severity level (immediate fix for severe defects, schedule fix for moderate severity defects, and monitor progress for low severity defects).

Another component of the system is the examples database. The examples database contains few-shot training examples for each task category. However, not every example contains all three parts (relevant data, user query, and expected task-specific output from LLM). In some cases, no relevant data is retrieved from the digital twin. Instead, time-series data or geometry data are retrieved from the digital twin via Python scripts. Such an example can be found in appendix “An example for *Task 1: Data visualisation* from the examples database”, where the system is asked to retrieve and visualise time-series data from a specified monitoring station.

### Testing and validation

In order to test and validate the solution proposed in this work, test cases have been constructed for each type of task introduced in section “Scope”. One test case has been proposed for each type of task introduced, and the proposed test cases are summarised below:

1. What is the historical water level at the station “Bourton Dickler” for the last 5 days?
2. Are there currently any defects on the road network?
3. Could you please change the severity level of the defect located at M25 28km to “severe”?
4. Could you please visualise the junction of Vabaduse väljak and Pärnu maantee for me?
5. Could you please generate a work schedule based on the current status of the road network?

### Implementation details

These test cases are first categorised using a structured prompt (shown in appendix “Prompt for categorising tasks”). Then, they are processed according to their categorisation results. More details can be found in the next section.

The LLM used in this project to generate the results in the next section is the gpt-3.5-turbo model developed by OpenAI, and the model is accessed via the OpenAI API with the openai Python library.

## Discussion

### Results & findings summary

This section presents a comprehensive overview of the outcomes and discoveries obtained from the implementation

and evaluation of our proposed natural language interaction system with infrastructure digital twin.

First, the task categorisation results are presented in Table 1 below.

Table 1: Expected categories vs actual categorisation results for the test cases

Test Case	Expected Category	Categorisation Result
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5

It can be seen from Table 1 above that the LLM is able to categorise all the test cases correctly.

Then, task-specific results are generated for each of the test cases. For test case 1, the test prompt, together with the retrieved example, are compiled according to the prompt structure introduced in step 3 in section “Details” into an input into the LLM. Then, the output from the LLM, which is a piece of Python script with chain-of-thought comments, is executed in a Python interpreter to produce the visualisation result shown below in Figure 5.

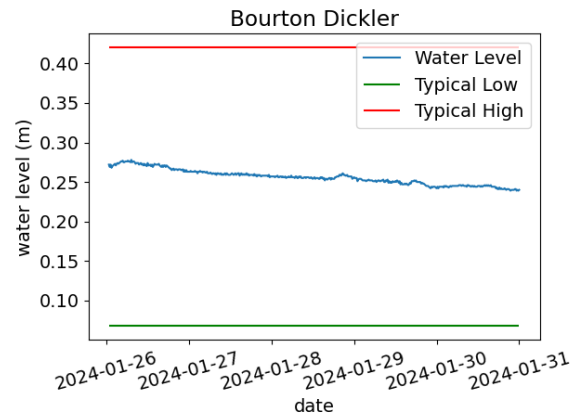


Figure 5: Visualisation result generated by the system

It can be seen from Figure 5 above that a visualisation of the requested time-series data has been successfully and correctly generated.

Then, for test case 2, the test prompt, together with the retrieved example and the defect data retrieved from the digital twin, are concatenated according to the prompt structure introduced in step 3 in section “Details” into an input into the LLM. Then, the output from the LLM is presented below:

Yes, I have found a total of 8 defects

on the road network. Here is a summary of the defects:

High Severity Defects:

1. Pothole on M25 at 112km

...

3. Crack on A14 at 21km

It is suggested that you prioritize fixing the high severity defects immediately, schedule the moderate severity defects for the next round of fixing, and monitor the low severity defects for progress.

The full result is available in appendix “Full result for test case 2”. It can be seen that the system can retrieve the relevant data from the digital twin and generate a natural language summary. Upon cross-checking with the actual data stored within the digital twin, it was confirmed that the system retrieved the correct information. This verification process involved a comparison between the output generated by the system and the actual defect records within the digital twin. This consistency validates the reliability of the data retrieval process implemented by the system.

Next, for test case 3, the test prompt, and the retrieved example, are combined according to the prompt structure detailed in step 3 in section “Details” into an input into the LLM. The output from the LLM is then executed within a Python interpreter to apply the changes to the digital twin. The system returned a “success” message, and it was verified that the requested change had been successfully applied to the digital twin.

Then, for test case 4, the test prompt, and the point cloud file(s) retrieved from the digital twin, are collated according to the prompt structure in step 3 in section “Details” into an input into the LLM. Then, the output from the LLM, which is the name of the requested point cloud file, is passed into a pre-structured Python script (shown in appendix “Python script to visualise point cloud data”) to generate the visualisation shown in Figure 6 below.

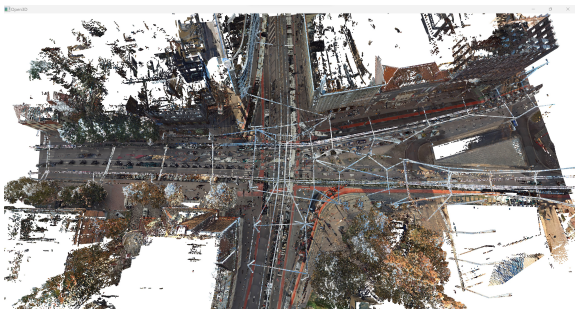


Figure 6: Model visualisation generated by the system

It can be seen from Figure 6 that the system is able to correctly retrieve and visualise the relevant point cloud data requested by the user.

Finally, for test case 5, the test prompt, together with the suggested actions retrieved from the digital twin and the relevant examples, are combined according to the prompt structure introduced in step 3 of section “Details” into an input into the LLM. The output from the LLM is then executed within a Python interpreter to generate the visualisation of the work schedule as a Gantt chart, which is shown in Figure 7 below.

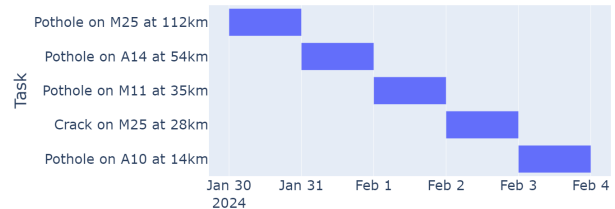


Figure 7: Work schedule visualisation generated by the system

The assumptions and criteria used here were:

1. Only “severe” and “moderate” defects are included in the schedule as the suggested action for low-severity defects is “monitor progress”.
2. Prioritise “severe” defects over “moderate” defects.
3. Within “severe” and “moderate” defects, prioritise “M” roads over “A” roads.
4. Within “M” roads and “A” roads, prioritise roads with a smaller road number assigned to them.
5. Within each road, schedule the tasks according to the location of the defects, and work along each road from the defect with the smallest km number to the defect with the largest km number.
6. The time for fixing each defect is one day.

These parameters and criteria can be changed easily within the digital twin and ChatTwin to reflect the real situation. The assumptions are made purely for demonstration and to validate the results generated by the system, and do not reflect any real circumstances in the real world. It can be seen from Figure 7 above that the system is able to generate a correct work schedule according to the pre-set criteria and visualise it as a Gantt chart.

## Contributions

Our work presents a novel approach to enhance the human-information interaction with infrastructure digital twins by leveraging the capabilities of LLMs, filling gaps in both the fields of digital twin research and human-computer interaction (HCI), as detailed in the following two paragraphs.

First, this work fills the gap in digital twin research by proposing a systematic approach to structure prompts that effectively leverage LLMs for generating informative and contextually relevant outputs with infrastructure digital

twins. This enables the LLMs to perform domain-specific tasks without the need for extensive fine-tuning. Our work contributes to the growing field of prompting pre-trained LLMs, providing some structured templates for structuring prompts that effectively guide the responses of LLMs within the context of infrastructure digital twins. This work also presents an empirical evaluation of our LLM-based interaction system through scenario-based simulations, demonstrating its strengths, limitations, and potential areas of improvement, offering valuable insights for researchers and practitioners interested in advancing natural language interaction with digital twins in the domain of infrastructure management.

In addition, current infrastructure digital twin research often emphasises technical aspects. This work primarily enhances the interaction layer, bridging the gap between intricate domain-specific data in digital twins and straightforward, user-friendly interactions, adding a user-centred design perspective, and aligning the infrastructure management domain with foundational HCI principles by facilitating more intuitive communication with digital twin systems, further enhancing the capabilities of conventional digital twin systems. This paves the way for stakeholders to more effectively engage with digital twin systems for infrastructure monitoring, control, and decision-making.

## Conclusions

The integration of LLMs with infrastructure digital twins has far-reaching implications for the infrastructure management domain and beyond. By providing a more accessible, intuitive and user-friendly interface, LLM-driven interactions reduce the learning curve required to interact with intricate infrastructure digital twin systems, empowering decision-makers at different levels, from engineers to policymakers, with timely and contextually relevant insights. This improved decision-making capability, supported by real-time data analysis and predictions provided by the capabilities of digital twins, can lead to more informed choices regarding infrastructure management, maintenance, and resource allocation.

## Recommendations for future

While our research provides valuable insights, some limitations remain. This section lists some potential areas for further investigation.

### *Enhancing interaction feedback*

While the current system can successfully generate and execute Python scripts for requested actions on the digital twin, future improvements can focus on providing more comprehensive feedback to users throughout the process to enhance user engagement and make the interface more informative. This can involve monitoring the digital twin's

status during and post-execution, enabling the system to confirm the successful implementation of user requests or flag potential issues, thus ensuring the accuracy of performed actions. Users might receive a visual cue for successful operations, or a notification detailing any encountered challenges. This could establish a closed feedback loop and eliminate ambiguity about action outcomes. Potential research questions could include: How can real-time feedback during task execution enhance user trust and system transparency for users of different expertise levels? What impact does visual or textual feedback have on user comprehension and satisfaction?

### *Expanding task categories*

While this project addresses the five most common tasks associated with infrastructure digital twins, there exists substantial potential to broaden the range of tasks covered by this approach. Expanding beyond these fundamental tasks can further improve system applicability. Furthermore, with the ReAct (Yao et al., 2023) approach, there is also potential to develop a more generally applicable natural language interaction system. Such a system could leverage the advanced reasoning capabilities of LLMs to autonomously deduce steps for various user-initiated actions beyond pre-defined task categories.

### *Multi-turn interactions and task clarification*

The current focus of the project on single-turn interactions provides a solid foundation for future research exploring more complex multi-turn conversations. This expansion will enable the system to engage in more intricate and dynamic interactions with users, improving the adaptability of the system. This extension is particularly crucial for addressing scenarios that involve ambiguity and might require additional clarification. In such cases, the system can proactively seek clarification from users when faced with potential misunderstandings to help disambiguate potentially confusing queries. Moreover, multi-turn interactions could also be leveraged to handle situations where users require assistance in accurately conveying their intent. With multi-turn interactions, the system's ability to interact, interpret, and execute user intents can be significantly enhanced, ultimately contributing to a more effective and user-centric infrastructure management system. Potential research questions could include: How do multi-turn interactions influence user experience and system accuracy in complex query scenarios? What methodologies can effectively disambiguate user queries?

### *User-centric design evaluation*

Future enhancements can be rigorously user-tested in simulated and controlled real-world scenarios to validate their effectiveness and user-centricity. This approach will ensure that proposed improvements align with actual user

needs and preferences, thereby enhancing the practical applicability of our findings.

### *Evaluating performances with different LLMs*

OpenAI's gpt-3.5-turbo model has been used in this work. In the future, tests can be conducted on other LLMs (e.g. PaLM, BERT, and Llama) to evaluate and compare the effectiveness of the system with different LLMs.

## **Acknowledgements**

The authors gratefully acknowledge the EPSRC, Bentley Systems and the University of Cambridge for funding this research through the EPSRC Centre for Doctoral Training in Future Infrastructure and Built Environment: Resilience in a Changing World (EPSRC grant reference number EP/S02302X/1); and Christopher-Robin Raitviir and Andres Maremäe from Tallinn City Council for generously providing data from the Tallinn City Digital Twin. This work also uses Environment Agency flood and river level data from the real-time data API (Beta). For the purpose of open access, the authors have applied a Creative Commons Attribution (CC BY) licence to any Author Accepted Manuscript version arising from this submission.

## **References**

- Barricelli, B. R., Casiraghi, E., and Fogli, D. (2019). A survey on digital twin: Definitions, characteristics, applications, and design implications. *IEEE Access*, 7:167653–167671.
- Broo, D. G., Bravo-Haro, M., and Schooling, J. (2022). Design and implementation of a smart infrastructure digital twin. *Automation in construction*, 136:104171.
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners.
- Chui, M., Manyika, J., Bughin, J., Dobbs, R., Roxburgh, C., Sarrazin, H., Sands, G., and Westergren, M. (2012). The social economy: Unlocking value and productivity through social technologies | McKinsey. <https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/the-social-economy>.
- Dingli, A. and Haddod, F. (2019). Interacting with intelligent digital twins. In Marcus, A. and Wang, W., editors, *Design, User Experience, and Usability. User Experience in Advanced Technological Environments*, pages 3–15, Cham. Springer International Publishing.
- Gao, L., Madaan, A., Zhou, S., Alon, U., Liu, P., Yang, Y., Callan, J., and Neubig, G. (2023). PAL: Program-aided language models.
- Grafius, D. R., Varga, L., and Jude, S. (2020). Infrastructure interdependencies: Opportunities from complexity. *Journal of Infrastructure Systems*, 26(4):04020036.
- Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. (2023). Large language models are zero-shot reasoners.
- Marai, O. E., Taleb, T., and Song, J. (2021). Roads infrastructure digital twin: A step toward smarter cities realization. *IEEE Network*, 35(2):136–143.
- Pairat, È., Ardón, P., Liu, X., Lopes, J., Hastie, H., and Lohan, K. S. (2019). A digital twin for human-robot interaction. In 2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pages 372–372.
- Pazos R., R. A., González B., J. J., Aguirre L., M. A., Martínez F., J. A., and Fraire H., H. J. (2013). *Natural Language Interfaces to Databases: An Analysis of the State of the Art*, pages 463–480. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Rudskoy, A., Ilin, I., and Prokhorov, A. (2021). Digital twins in the intelligent transport systems. *Transportation Research Procedia*, 54:927–935. International Scientific Siberian Transport Forum - TransSiberia 2020.
- Siyayev, A., Valiev, D., and Jo, G.-S. (2023). Interaction with industrial digital twin using neuro-symbolic reasoning. *Sensors*, 23(3).
- Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., and Fedus, W. (2022). Emergent abilities of large language models.
- Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. (2023). Chain-of-thought prompting elicits reasoning in large language models.
- Winterbotham, M., Kik, G., Selner, S., Menys, R., Stroud, S., and Whittaker, S. (2020). Employer skills survey 2019: Training and workforce development. [https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment\\_data/file/936487/ESS\\_2019\\_Training\\_and\\_Workforce\\_Development\\_Report\\_Nov20.pdf](https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/936487/ESS_2019_Training_and_Workforce_Development_Report_Nov20.pdf).
- Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. (2023). ReAct: Synergizing reasoning and acting in language models.

# Appendices

## Prompt for categorising tasks

Please determine which task category this user prompt belongs to: [user prompt]

-----

Candidate task categories:

Task category 1: Data visualisation

Task description: Tasks where the user asks the system to visualise specific time-series data

Example user prompt: What is the water level data at the station "Cam" over the past 2 days?

Task category 2: Information summarisation

Task description: Tasks where the user asks the system to find and summarise certain data or information

Example user prompt: Are there currently any defects on the road network?

Task category 3: Performing changes to digital twin

Task description: Tasks where the user asks the system to change the property value of a specific instance within the digital twin

Example user prompt: Can you change the status of the defect at M2 112km to fixed?

Task category 4: Model visualisation

Task description: Tasks where the user asks the system to visualise a certain section of the digital twin model

Example user prompt: Can you show me a visualisation of Regent Street between 5km and 8km?

Task category 5: Work schedule enquiry

Task description: Tasks where the user asks the system to generate/visualise a work schedule

Example user prompt: Could you generate a work schedule for defect rectification?

-----

Please only respond with the task category number in Arabic number.

## Prompt to retrieve point cloud data

The available point cloud data files are:

[the available point cloud data retrieved from the digital twin]

The user's query is: [user query]

Is the point cloud data requested by the user available? If it is available, please only respond with the name of the corresponding point cloud data file. If it is not available, please respond with "Not available".



## An example for *Task 1: Data visualisation* from the examples database

Q: What is the water level data at the station "Cam" over the past 2 days?

A:

```
# To get water level data, we first need to find the list of monitoring stations using
the function build_station_list()
stations = build_station_list()
# Then we need to find station "Cam" from the list and store it in the variable
station_cam
station_name = "Cam"
for station in stations:
    if station.name == station_name:
        station_cam = station
        break
# We then need to fetch the water level data over the past 2 days using the function
fetch_measure_levels
dt = 2
dates, levels = fetch_measure_levels(station_cam.measure_id,
dt=datetime.timedelta(days=dt))
# We can then finally plot the water level data using the function plot_water_levels
plot_water_levels(station, dates, levels)
```

## Full result for test case 2

Yes, I have found a total of 8 defects on the road network. Here is a summary of the defects:

High Severity Defects:

1. Pothole on M25 at 112km
2. Pothole on A14 at 54km

Moderate Severity Defects:

1. Crack on M25 at 28km
2. Pothole on M11 at 35km
3. Pothole on A10 at 14km

Low Severity Defects:

1. Pothole on M11 at 87km
2. Pothole on M11 at 134km
3. Crack on A14 at 21km

It is suggested that you prioritize fixing the high severity defects immediately, schedule the moderate severity defects for the next round of fixing, and monitor the low severity defects for progress.