1. This paper is concerned with the development of a hydrological-hydraulic surrogate architecture for urban drainage systems. The paper is within scope of the journal. It also presents some interesting, novel ideas that go beyond previous work. However, these are not at all supported by empirical results. Instead, all results are presented for the new framework as is. We therefore don't know at all if the study actually made progress compared to previous similar works. In addition, the paper is not always concise, and the methodological details are occasionally incomplete and sometimes confusing.

I have provided details below. I don't think that these issues can be addressed in a normal revision, and I therefore suggest to reject and invite the resubmission of a thoroughly revised manuscript that also includes a number of new results supporting the methodological progress.

We thank the reviewer for carefully evaluating our work and for the constructive and detailed feedback. We agree that additional empirical analyses, including ablation studies, component-level comparisons, and computational performance benchmarks, will further strengthen the manuscript by isolating the contribution of individual architectural choices. These analyses will help illustrate methodological progress relative to previous studies, beyond evaluating the unified framework as a whole.

At the same time, we would like to clarify that the manuscript already includes substantial empirical evidence demonstrating that the proposed unified surrogate performs well across a range of hydrological–hydraulic tasks. The present results evaluate predictive accuracy for node states, conduit flows, flood-node detection, and flood-volume estimation across multiple storm events. These findings support the central proof-of-concept contribution: that a single end-to-end GNN can successfully learn the coupled rainfall–runoff and flow-routing process. The additional analyses we will add in response to the reviewer's suggestions are intended to complement this existing evidence by providing finer-grained insight into the role of each model component, rather than to fill an absence of empirical support.

We also acknowledge the reviewer's observation that certain methodological sections can be made more concise and clearer. In the revised manuscript, we will streamline the presentation, improve figure design, and expand explanations where needed to ensure that the methodological pipeline is transparent and easy to follow. Many of the points raised, e.g., clarifying feature definitions, refining figure annotations, and specifying training configurations, are straightforward improvements in presentation, and we will address each of them carefully in the revision.

Detailed comments:

2. l64: I disagree on this point. The main reason why previous studies focused on the hydraulics is because that is where most of the computation time is used. Hydrological models are already fast and can also be integrated in differentiable form into a surrogate framework.

There also aren't any results for what we gain from including the hydrological model in the surrogate architecture, neither in terms of accuracy nor speed.

We agree that hydraulic routing is typically the dominant computational bottleneck in physics-based models . Our intention was not to suggest that hydrological models are computationally slow on their own, but rather the value of learning the coupled hydrological-hydraulic process within a single unified surrogate. In this formulation, the model directly maps rainfall and catchment attributes to node depths, conduit flows, and flooding behavior, instead of relying on a separate runoff-generation step followed by a distinct hydraulic module.

We recognize that hydrological models can be embedded in differentiable form within surrogate frameworks, but such designs generally retain a modular structure in which precipitation is first transformed to runoff through a predefined hydrological component and then routed hydraulically. By contrast, our GNN-based framework replaces the entire simulation pipeline with an end-to-end mapping that learns rainfall–runoff–routing interactions jointly. The inclusion of hydrological parameters (e.g., imperviousness, slope) as node features allows the network to infer how rainfall interacts with local catchment characteristics and how these effects propagate through the drainage graph, rather than assuming a fixed hydrological mapping upstream of the surrogate.

Empirically, Sections 5.2 and 5.3 already show that this unified surrogate reproduces key hydrological–hydraulic behaviors across multiple events, including node depths, conduit flows, flood node detection, and flood volume estimation. This provides the core proof-of-concept for the study, demonstrating that the proposed model *can* learn the complete coupled rainfall–runoff–routing process, which is a non-trivial advancement beyond modular or hydraulics-only surrogate designs.

To further clarify the benefits of this formulation, we will include a new subsection in the Results section reporting the model's computational performance. Because the goal of our approach is to replace the entire physics-based pipeline, the most meaningful metric is the end-to-end speedup relative to the full SWMM workflow rather than a component-wise hydrology comparison, which would not reflect the different problem formulations. The new subsection will therefore quantify the overall pipeline acceleration achieved by the unified GNN surrogate and clarify its computational advantages.

We also note that the unified, end-to-end formulation enables modeling capabilities that modular or decoupled surrogates cannot easily support. By learning the full rainfall–runoff–routing process, the model can act as a fast 1D surrogate for coupled 1D/2D flood simulation frameworks, where its predicted nodal flood volumes can serve as source terms for a 2D inundation surrogate. Additionally, because the model learns how subcatchment properties (e.g., imperviousness, slope) affect hydraulic response, it can support decision-relevant scenario analyses such as evaluating the hydraulic impact of various LID strategies. We will add a brief clarification of these potential applications in the Discussion section to make these broader implications explicit.

3. l95: bad reference

Thank you. We will fix this in the revised manuscript.

4. Section 2.1: do we need to describe SWMM? maybe this can be moved to an appendix

In the revised version, we will move the description of SWMM to an Appendix.

5. Section 2.2.1: This section can also be shortened. Do mention that this is a graph convolution? Why was it selected over GineConv and GATConv that seem to have performed better in other studies?

Thank you for the constructive feedback. We agree that this section can be streamlined, and we will revise it accordingly. We will also make it explicit that the "GN block" used in our processor is a form of graph convolution based on the Graph Network (GN) framework and implements a message-passing structure.

Regarding the choice of this architecture over GATConv or GINEConv: our primary requirement is the ability to jointly learn dynamic edge states (conduit flows) and dynamic node states (junction

depths) within the same model. This requires an operator that explicitly updates edge embeddings as part of the iterative message-passing process. The standard GATConv does not natively incorporate edge features and is designed primarily for node-level tasks, requiring additional custom fusion layers to embed edge information (Veličković et al., 2018, Zhang et al. 2024). GINEConv does incorporate edge features but does so through a more restrictive additive formulation and does not update edge states iteratively during message passing (Garzón et al. (2024a, Garzón et al., 2024b).

In contrast, the GN Block we use, which is based on architectures explicitly designed for physics simulation (Sanchez-Gonzalez et al., 2018; Pfaff et al., 2021), performs a two-stage update that first computes new edge embeddings via a learned non-linear function of the edge and its two incident nodes, and then updates node states using these dynamically updated edges. This architecture is specifically designed for domains where relational (edge) dynamics are as important as object (node) dynamics, making it well suited for stormwater hydraulics.

We will revise Section 2.2.1 to shorten the description and clearly justify this architectural choice so that the rationale is transparent to the reader.

6. Figure 1: This figure is both overloaded and can at the same time not be understood from the figure and the caption alone. The lower part illustrating the message passing process is mostly confusing, and can safely be removed, because this process is already well documented in the text as well as the literature.

In the upper part, it is not clear what node types a and b are, what the indices of x1, x4, x14 and x13 indicate, if four different encoder MLPs are used or if the same MLP is used, that we start working on edges in the processor and symbolise inputs for three different edges - each composed of upstream and downstream node properties as well as edge properties, they values are being aggregated over edges connected to each node, that we need to apply another MLP to decode, and how we finally arrive at predictions for t+2 and t+3. The figure just repeats the message passing step, so it may well be a consideration to drop it.

Thank you for the constructive feedback. We will completely redesign the figure in the revised version. The new version will be a simplified schematic that directly addresses the issues raised by the reviewer: we will remove the redundant lower panel illustrating message passing, clearly define 'Node type a' and 'Node type b' , replace the confusing indices (e.g., x1, x14) with a general data flow, clarify the encoder/decoder steps, and be corrected to clearly illustrate a single-step autoregressive prediction ($t \rightarrow t + 1$) to resolve the inconsistency the reviewer noted.

7. Figure 2: Text in the top part of the figure is too small. Information in this subfigure and the table is repeated, which is confusing. It is not clear to me why went nice coordinates would be necessary inputs to the model. The network is fully defined from adjacency and edge length (and slope, which is missing). dx and dy are not explained.

Thank you for the constructive feedback. We will revise this figure to address the issues raised by the reviewer.

Regarding the feature inputs, we acknowledge the inconsistency in listing “slope” as a conduit feature and will correct this in the revision. In the revised version, we will also clarify the role of the spatial features. Absolute node coordinates (x, y) are provided as static node attributes, and dx and dy are derived edge features representing the relative horizontal displacement between connected nodes. Together with node elevations, these features allow the model to infer conduit

orientation and slope implicitly, while also providing spatial context that improves learning of location-dependent hydraulic behavior. We will clearly define dx and dy and revise both the figure and the accompanying text to ensure this rationale is transparent.

8. Eq. 6: Why do you use difference in depth and not hydraulic head, which is the actual driving force behind the flow through edges?

We agree that the hydraulic head gradient (ΔH), not just the water depth difference (Δy), is the true physical driver of flow. In our formulation, the model receives both components needed to infer ΔH: the water depth difference (Δy) as the Diff-depth feature, and the elevation difference (Δz) is available implicitly because node elevations ($z_i \; and \; z_j$) are included in the node features used during message passing (Eq. 3). Therefore, the GNN has access to $\Delta H = \Delta z + \Delta y = (z_i - z_j) + (y_i - y_j)$, and can learn the corresponding hydraulic head gradient along each edge.

We will revise Section 3.1 to make this justification clearer and to explicitly state how the model reconstructs the hydraulic head gradient from the supplied features.

9. l200: I don't understand this. In Figure 2 you illustrate that you predict three steps at the time. Here and in Figure 3 it looks like you move one step at the time. Please clarify and make sure that the description is consistent throughout.

Thank you for pointing out this inconsistency. Our model is one-step-ahead predictor that is applied recursively (autoregressively) to generate multi-step forecasts. Figure 3 correctly illustrates this autoregressive structure. We agree that this was not explained clearly enough in the manuscript, and we will revise the methodology to explicitly state that the model predicts one step at a time rather than multiple steps in a single forward pass.

The confusion in Figure 2 arises from how the pushforward training strategy was visualized. The three-step window shown there reflects the training configuration used for the pushforward trick, not the model's inherent prediction mechanism. We will revise the figures, their captions, and the corresponding text so both figures consistently represent the autoregressive nature of the model and avoid ambiguity.

10. Section 3.2: The works of Palmitessa, Garzon and Bentivoglio (for 2D flooding) all condition the model on its own predictions during training. What you do different is that you include an additional loss term where the model is conditioned on the ground truth. You need to show that this improves training stability compared to the situation where we condition on model predictions only. This may actually be the case, because conditioning on the model predictions essentially corresponds to a recurrent network, which are well known for vanishing gradient issues. In any case, the argument is currently formulated the wrong way round, and results for this aspect are missing. Figure 4 is again overcomplicating things. I'm able to understand the concepts from the text, but not from the figure, which is overloaded with information that is not relevant in this context.

Thank you for this helpful observation. We agree that additional empirical evidence isolating the effect of our stability loss would strengthen the manuscript. We will include an ablation study comparing training with and without the stability component to demonstrate its contribution to stability and error recovery.

We also appreciate the reviewer's comparison to prior work. Our approach differs from standard autoregressive conditioning or multi-step losses used in previous 2D flood surrogates. Specifically,

our contribution is more than simply adding a loss term conditioned on the ground truth (L_{real}). The novelty lies in the specific implementation of our stability loss ($L_{stability}$), which uses a gradient-blocking pushforward strategy. During training, the model's own predictions are reintroduced into the input sequence, but the gradient from the initial prediction is cut. This differs fundamentally from standard autoregressive or multi-step losses, where gradients propagate through the entire predicted trajectory. By blocking the gradient pathway, $L_{stability}$ forces the network to learn robustness to distribution shift (i.e., to recover from its own imperfect predictions) rather than optimizing only short-term accuracy. We extend this mechanism by applying the stability loss over a three-step window (n = 3), which we found provides stronger regularization for stormwater applications. This gradient-blocking, multi-step stability design is distinct from the training formulations in previous studies, and we will revise the manuscript to describe this implementation more clearly.

To avoid confusion, we agree that Figure 4 currently contains more information than needed. In the revision, we will simplify the figure and revise the explanation of the training strategy so that the distinction between our stability loss and conventional autoregressive conditioning is clear.

11. Section 3.3: These loss functions are interesting and potentially valuable, but you don't illustrate that you actually achieve better training performance than previous studies that have just used L_real. How do we know that these modifications actually make a difference?

Thank you for this helpful comment. We agree that the manuscript should explicitly demonstrate the contribution of the physics-guided loss components. In the revised version, we will include a new ablation study comparing

- a baseline model trained with L_{base} only
- variants trained with $L_{base} + L_{penalty}$ and $L_{base} + L_{diff}$, and
- the full model trained with all loss terms

This ablation will quantify the effect of each component on training stability and predictive accuracy. We will also revise Section 3.3 to clarify the motivation for these loss terms and how they complement L_{real} in guiding physically consistent predictions.

12. Section 5.1: It seems all the information in the table is also described here. Please remove either text or table. How many forecast steps did you use for training? How much GPU memory did you have available?

Thank you for the helpful suggestion. We will remove the redundant text in Section 5.1 and refer directly to Table 1 to streamline the presentation. For clarity, the training forecast horizon used in the pushforward strategy was three steps, and the model was trained on an NVIDIA RTX 4090 GPU with 24 GB of VRAM. We will include these details explicitly in the revised manuscript.

13. Figure 12: Do the boxplots illustrate variations of average NSE value across events, or do they combine variations of NSE on node and event level, i.e. is 0.89 the lowest average NSE for all events, or is it the lowest value observed for any node in any event?

Thank you for the question. The boxplots do not represent variations of average NSE *across events*. Instead, each boxplot summarizes the per-entity performance (i.e., per-node or per-conduit), where the NSE values are computed for each entity over all test events combined. Thus, values such as 0.89 represent the lowest per-entity NSE observed across all events, not the lowest average event-

level NSE. We will revise the caption and accompanying text for Figure 12 to clearly describe this aggregation method.

14. Section 5.3: I think this assessment should be done as an average across events with surcharges. Considering only a single event is very unreliable.

The flood performance analysis focused on a single, high-intensity storm, which we selected as a representative event to rigorously test the model's flood detection capabilities under challenging conditions. We note, however, that results for multiple rainfall events are already included in other sections of the manuscript. For the sake of brevity, we initially presented only one scenario here, but in the revised version we can add complementary results summarizing flood detection metrics (CSI, precision, and recall) across several additional storm events from our test set to demonstrate that the model's strong performance is consistent.

15. Figure 14: axis labels miss units

Thank you for raising this issue. We will fix this in the revised manuscript.

16. Results for computational speed up are missing entirely

Thank you for pointing this out. We agree that including computational performance is important. In the revised manuscript, we will add a dedicated subsection in the Results section reporting the end-to-end computational speed of the unified surrogate relative to the full SWMM workflow. This subsection will quantify runtime and overall pipeline speedup, including descriptive statistics comparing CPU-based and GPU-based runs, to clearly demonstrate the computational benefits of the proposed model.

17. Section 5.3.2: The behaviour in Fig. 15 is already clear from earlier results. This figure can therefore be moved to an appendix. It is also not common practice to include figures in the discussion.

Thank you for the suggestion. Figure 15 was included in Section 5.3.2 to visually support the discussion on model limitations, but we agree that it can be moved to the appendix. In the revised manuscript, we will move Figure 15 to an appendix and adjust the text in Section 5.3.2 accordingly.

18. In addition, this section is missing an important limitation, namely that the trained model remains system specific.

Thank you for pointing out this important limitation. We agree that the trained model in its current form is system-specific to the Haven Creek Watershed and does not generalizable to a new domain without retraining. We will add this explicitly to the limitation section.

At the same time, we would like to clarify that the proposed framework itself is not inherently system-specific. The unified, end-to-end surrogate formulation is general and can be trained on other stormwater systems or extended to additional domains. This paper serves as a proof-of-concept for the methodology, and we will revise the manuscript to clearly distinguish between the generality of the framework and the system-specific nature of the trained model used in this study.

19. Section 6: this section does not belong into a scientific paper

We will remove Section 6 from the manuscript as suggested.

References

- Garzón, A., Kapelan, Z., Langeveld, J., & Taormina, R. (2024a). Transferable and data efficient metamodeling of storm water system nodal depths using auto-regressive graph neural networks. *Water Research, 266*, 122396.
- Garzón, A., Kapelan, Z., Langeveld, J., & Taormina, R. (2024b). Accelerating Urban Drainage Simulations: A Data-Efficient GNN Metamodel for SWMM Flowrates. *Engineering Proceedings, 69*(1), 137.
- Pfaff, T., Fortunato, M., Sanchez-Gonzalez, A., & Battaglia, P. W. (2021). Learning mesh-based simulation with graph networks. *International Conference on Learning Representations (ICLR)*.
- Sanchez-Gonzalez, A., Heess, N., Springenberg, J. T., Merel, J., Riedmiller, M., Hadsell, R., & Battaglia,P. (2018). Graph networks as learnable physics engines for inference and control. *International Conference on Machine Learning (ICML)*.
- Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph attention networks. *International Conference on Learning Representations (ICLR)*.
- Zhang, Z., Tian, W., Lu, C., Liao, Z., & Yuan, Z. (2024). Graph neural network-based surrogate modelling for real-time hydraulic prediction of urban drainage networks. *Water Research, 263*, 122142.