Composites: Part A 204 (2026) 109656

Contents lists available at ScienceDirect

Composites Part A

journal homepage: www.elsevier.com/locate/compositesa

FlowCastNet: A CNN-based surrogate model for the rapid prediction of flow filling patterns in VARTM processes

Yuwei Feng [a,b], Bin Yang [a,b], Yunshi Huang [c], Jihui Wang [a,*], Philippe Causse [b,*]

[a] School of Material Science and Engineering, Wuhan University of Technology, 122 Luoshi Road, Hongshan, Wuhan, 430070, Hubei, China
[b] Department of Systems Engineering, École de Technologie Supérieure (ÉTS), 1100, rue Notre-Dame Ouest, Montréal, H3C 1K3, Québec, Canada
[c] Le Laboratoire d'imagerie, de vision et d'intelligence artificielle (LIVIA) , École de Technologie Supérieure (ÉTS), 1100, rue Notre-Dame Ouest, Montréal, H3C 1K3, Québec, Canada

ARTICLE INFO

Keywords:
Vacuum-assisted resin transfer molding
Processing optimization
Convolutional neural network
Flow prediction

ABSTRACT

The manufacturing of fiber-reinforced composites by vacuum-assisted resin transfer molding (VARTM) commonly requires a distribution medium (DM) to facilitate resin impregnation. The DM significantly accelerates the filling process but must be carefully designed to ensure defect-free parts. Traditional physics-based simulations can accurately predict filling patterns, however, inherent computational cost limits their applicability for rapid decision-making in high-dimensional and large design spaces. This paper proposes an alternative surrogate modeling approach that employs a multilayer perceptron (MLP) for initial feature encoding and a convolutional neural network (CNN) as the backbone. The resulting model, FlowCastNet, was specifically developed for fast prediction of filling patterns associated with flat-panel manufacturing. A parametric study was first conducted through numerical simulation, considering several key process parameters such as preform properties, DM placement and race-tracking effects. The resulting synthetic dataset was then used to train the surrogate model and assess its predictive accuracy. Despite the significant size of the input feature domain, the model achieves satisfactory performance when evaluated on an independent dataset ($R^2 = 0.992$ and MSE of approximately 10^{-3}). Remarkably, the model provides extremely fast predictions with an inference time near 1 ms. To highlight the advantage brought by such computational speed, two multi-objective optimization scenarios were analyzed as representative examples. Compared to numerical simulation, FlowCastNet allows exhaustive search of the design space, which enables rapid identification of the Pareto frontier. While small quantitative discrepancies remain, the proposed approach shows promising potential to complement physics-based modeling for multi-criteria optimization in VARTM processes.

1. Introduction

Vacuum Assisted Resin Transfer Molding (VARTM) [1,2] is a popular out-of-autoclave technology for fabricating fiber-reinforced composites, due to its relatively low tooling cost and its capacity to produce large structures. In this process, a dry fibrous preform is placed between a rigid lower mold and a flexible upper vacuum bag, and a reactive resin is infused under vacuum pressure. However, the limited driving force provided solely by vacuum results in relatively low resin flow rates and long impregnation times, which may be impractical for production. To address this limitation, a highly permeable distribution medium (DM) is typically placed above the preform, as originally proposed in the Seemann composite resin infusion molding process (SCRIMP) [3]. The DM facilitates rapid in-plane resin flow along its surface while simultaneously promoting transverse impregnation through the preform thickness. This flow mechanism arises from the pronounced difference in permeability between the DM and the fibrous preform, with the permeability of the DM being 2~3 orders of magnitude higher [4,5]. The use of a DM significantly reduces the overall filling time and mitigates the risk of resin gelation prior to complete mold filling [6,7]. However, it also generates a complex 3D filling pattern with a significant discrepancy between the flow front in the DM and in the preform, which is known as the lead–lag distance (see Fig. 1).

Experimental observations showed a linear decrease of the total filling time when increasing DM dimensions [8]. On this basis, an intuitive approach is to cover the majority of the laminate's top surface with DM. While this strategy performs reasonably well for flat and thin components, it may not be suited for thick laminates [9] or parts with complex geometries [10]. In such cases, an excessive lead–lag distance can cause air entrapment and dry spots creation due to

* Corresponding authors.
E-mail addresses: jhwang@whut.edu.cn (J. Wang), philippe.causse@etsmtl.ca (P. Causse).

https://doi.org/10.1016/j.compositesa.2026.109656
Received 16 December 2025; Received in revised form 29 January 2026; Accepted 12 February 2026
Available online 16 February 2026

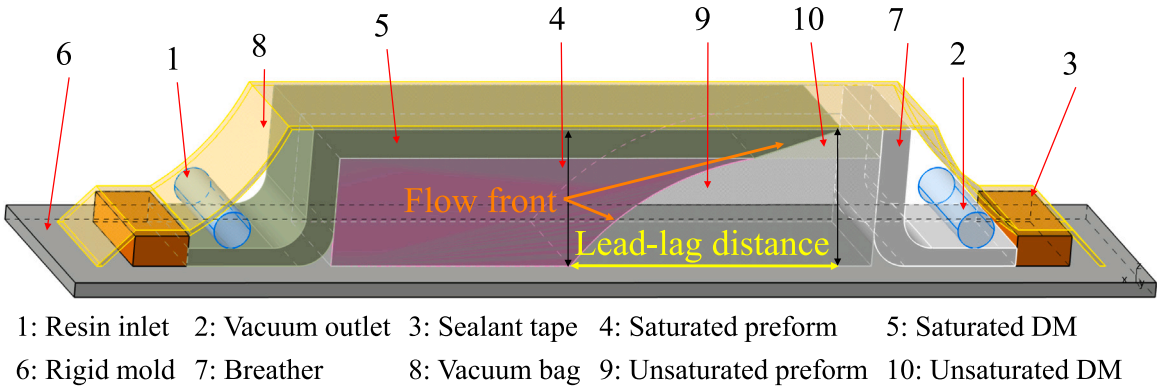

Fig. 1. Schematic of the SCRIMP process incorporating a distribution medium to accelerate resin flow.

irregular through-thickness flow and can also lead to significant resin loss if the resin reaches the outlet prematurely. To mitigate these issues, the DM configuration must be carefully tailored to minimize the total filling time while ensuring complete filling of the preform and limited resin waste.

In industrial practice, DM layouts are typically determined by process engineers based on prior experience. The initial designs are refined through trial and error, a process that is time-consuming, costly, and seldom optimal. To address this issue, several studies have proposed optimization techniques to identify the best DM placement strategy. Hsiao et al. coupled genetic algorithms with filling simulations to simultaneously minimize filling time and void content in the final part [10]. The DM placement was treated as a discrete problem by segmenting the top surface of the laminate into 6 regions that could be covered by 0,1 or 2 plies of DM. When applied to a complex rib structure, this approach converged to a solution that was very close to the global optimum identified by exhaustive search but the procedure required only 15% of the computational effort compared to the brute force technique. Sas et al. further elaborated this concept by adopting a tree search strategy to adapt the number of discrete DM regions needed in the optimization procedure [11]. Their analyses included race-tracking channels caused by inserts or sharp corners in the mold. Applied to a complex geometry, the proposed algorithm was able to identify a robust design of the DM allowing full impregnation in the presence of uncontrolled flow disturbances caused by race-tracking. A similar methodology was used by Seyednourani et al. who added a preliminary optimization step to identify the best locations for gates and vents before optimizing the DM placement [12].

Overall, the aforementioned studies clearly demonstrated the benefits of systematic optimization of the DM placement. However, identifying an optimum solution still requires running numerous numerical simulations, which incurs considerable computational costs. The latter cannot easily be reduced with the SCRIMP process as the combination of planar and through-thickness flow cannot be captured with simplified two-dimensional models and requires 2.5D or 3D approaches [13,14]. More importantly, even minor changes in geometry or process parameters necessitate complete model reconstruction. This includes geometry regeneration, remeshing, re-solving the numerical problem, and complete re-optimization. Recently, Szarski and Chauhan [15] proposed a radically different approach based on Machine Learning (ML) with strong potential to circumvent these limitations. These authors specifically studied the DM placement problem in rectangular laminates with local thickness variations. By interacting with physics-based flow simulations, a deep reinforcement learning agent gradually learned how to place the DM to minimize filling time and avoid dry spot formation. After training on different specific parts, the model was able to instantaneously predict the DM placement on new geometries. Application to a realistic aerospace part showed promising results as the proposed solution outperformed traditional expert design with a filling time reduction of 32%.

Once trained on a representative dataset, ML methods can rapidly predict flow patterns without repeated numerical resolution [16], making them well-suited for parameter optimization. Recent advances in high-performance computing hardware and open-source ML frameworks have further accelerated the integration of ML into composite manufacturing applications [17,18]. For example, several studies used Neural Network (NN) architectures with specific input/output combinations to analyze the impregnation stage of the resin transfer molding (RTM) process. Chai et al. developed a NN capable of predicting the maximum injection pressure and the required number of vents based on the location of the inlet and resin viscosity [19]. Zhu et al. used readings from pressure sensors distributed in the mold and the corresponding time as input to predict the temporal evolution of the flow front [20]. Chen et al. considered typical permeability disturbances encountered in RTM and used a physics-informed NN (PINN) to predict key flow field variables such as pressure and saturation [21]. Overall, the ML model exhibited a satisfactory predictive accuracy when benchmarked against a state-of-the-art numerical solver. While most previous studies have focused on flat rectangular composite panels, the application of NNs to RTM manufacturing has been extended to more realistic parts by Wang et al. [22]. These authors employed a recurrent NN that predicts a projected 2D image of the filling time based on the inlet location. Satisfactory predictions were obtained for both flat panels and typical composite B-pillar of complex geometry. Apart from the work of Szarski and Chauhan [15] mentioned previously, ML models have also been applied to the SCRIMP process by Matzuzaki et al. [23]. These authors proposed an alternative to full 3D modeling by developing a specific NN using 2D flow simulations as input. The model subsequently predicts the though-thickness filling time at each position in the mold. The method was applied to both flat and complex geometries and achieved satisfactory accuracy while reducing computation time by more than two orders of magnitude compared to a full 3D numerical analysis.

Among existing ML techniques, convolutional neural networks (CNNs) are particularly well-suited for processing grid-structured data such as images [24]. In the field of composite manufacturing, CNNs have notably been used to predict permeability from images of fibrous porous media [25–27]. The ability of CNNs to capture spatial hierarchies and visual patterns could also make them interesting candidates for modeling complex fluid flow dynamics in resin injection processes. To date, a few studies adopted such approach to analyze the filling stage of flat panels manufactured by RTM. Stieber et al. used pressure sensors data to predict the evolution of the flow front position and eventually detect the formation of dry spots [28]. CNNs fed with time-series pressure sensor data formatted as a specific type of image have also been employed to detect flow disturbances caused by dissimilar permeability regions [29] or race-tracking channels [30]. Building on this work, Fernández-León et al. designed an encoder–decoder model that takes a 2D permeability field incorporating boundary and initial

conditions, as well as time, as input and predicts the spatiotemporal evolution of saturation and pressure during mold filling [31]. Compared to numerical simulation, the model achieved an impressive execution time reduction of three orders of magnitude. The advantage of such computational speed was illustrated through an inverse analysis of experimental pressure data to quantify the properties of a dissimilar permeability region as well as the magnitude of race-tracking. To the best of the authors' knowledge, no attempt has yet been done to apply a similar method to analyze the filling stage in the SCRIMP process. As a matter of fact, the only publication related to SCRIMP identified in the literature is the work of Zhang et al. [32]. These authors integrated CNNs with recurrent neural networks (RNN) and long short-term memory (LSTM) architectures to predict the spatiotemporal evolution of temperature within the laminate from the knowledge of the imposed temperature cycle. Although not directly related to the impregnation stage, this study highlighted the benefits of very fast CNN prediction compared to classical physics-based simulation in the context of process monitoring and control requiring quick decision tool.

The present study aims to investigate the potential of CNN-based modeling to predict typical filling patterns encountered during SCRIMP manufacturing. To reach that goal, a specific CNN surrogate model was designed to predict the filling time distribution on the top and bottom surfaces of the laminate, based on the processing conditions formatted as scalar tuples. The remainder of this paper is organized as follows. The generation of the synthetic dataset and the design of the surrogate model are presented in Section 2 and Section 3, respectively. Section 4 then reports the training history and evaluates the predictive accuracy of the proposed approach. It also illustrates its potential for rapid multi-objective optimization of the DM configuration. Finally, Section 5 summarizes the main findings of the study and outlines potential directions for future research.

2. Synthetic dataset generation

Successful ML model training requires access to a dataset that is both diverse and representative of the target domain. Gathering the required amount of data through time and cost-intensive physical experiments poses inherent challenges for data-driven model training. To address this limitation, a common practice consists of using validated physics-based simulation tools to gather large training datasets [19,20,22,23,28–34]. A similar approach was adopted in this study, and the PAM-RTM software [35] was used to build a synthetic dataset consisting of 2000 flow simulations.

2.1. Governing equation

In the SCRIMP process, resin flow typically occurs at a low Reynolds number, with negligible inertial forces. In addition, we also neglect gravitational effects, resulting in a simplified balance between pressure forces and viscous resistance. Consequently, the momentum conservation equation can be represented by Darcy's law:

$$\mathbf{v}(\mathbf{c},t) = -\frac{\mathbf{K}}{\varphi\mu}\nabla P(\mathbf{c},t) \tag{1}$$

Here, φ denotes the porosity, μ is the viscosity of the resin and $\mathbf{v}$ is the fluid velocity. $\mathbf{c}, t$ stand for the spatial position vector and time, respectively. P is the fluid pressure, and $\mathbf{K}$ is the permeability tensor.

In this study, the fibrous preform and the DM are considered to be rigid porous media with constant permeability and porosity throughout the filling stage. We also assume isothermal conditions and neglect curing of the resin, which is treated as an incompressible fluid with constant density and viscosity. The transient flow is approximated as a sequence of steady-state processes under isothermal conditions. The governing equation is then given by:

$$\nabla\cdot\left(-\frac{1}{\varphi\mu}\begin{vmatrix} k_{xx} & k_{xy} & k_{xz} \\ k_{yx} & k_{yy} & k_{yz} \\ k_{zx} & k_{zy} & k_{zz}\end{vmatrix}\begin{vmatrix} \frac{\partial P(\mathbf{c})}{\partial x} \\ \frac{\partial P(\mathbf{c})}{\partial y} \\ \frac{\partial P(\mathbf{c})}{\partial z}\end{vmatrix}\right) = 0 \tag{2}$$

where $k_{ij}(i,j = x,y,z)$ are the components of the permeability tensor. The skew terms (i.e., $k_{xz}, k_{yz}, k_{zx}, k_{zy}$) are commonly set to zero due to material symmetry. This second-order partial differential equation can be solved numerically when appropriate boundary conditions are specified.

2.2. Numerical modeling

A simple parametric model was designed to simulate SCRIMP manufacturing under varying process conditions. As depicted in Fig. 2(a), the test case consists of a planar laminate of dimensions $L_f \times W_f \times H_f$. The model includes two channels of width W_{rt} on the longitudinal edges of the part to account for possible race-tracking, a phenomenon frequently encountered in composite manufacturing processes [36]. The laminate is covered by a rectangular DM having one edge on the inlet line and centered in the transverse direction. Under such constraints, the DM configuration is then fully described by its dimensions: length L_{DM}, width W_{DM} and thickness H_{DM}. To exploit the inherent symmetry of the mold setup, only half of the geometry was considered in the finite element model and the geometry was discretized into tetrahedral elements with a mesh size of 1 mm (see Fig. 2(b)) . This mesh size was chosen to balance spatial resolution and computational efficiency. As shown in Fig. 2(c), constant pressure boundary conditions (BCs) were applied on the two edges corresponding to the inlet and the outlet of the mold. Impermeable wall BCs were considered for the other surfaces.

2.3. Investigated process parameters

The material properties and BCs are needed to complete the numerical model introduced above. An orthotropic behavior is considered for the fibrous preform with principal permeabilities (K_{f1}, K_{f2}, K_{f3}) and porosity φ_f. The DM is assumed to be isotropic with permeability K_{DM} and porosity φ_{DM}. Finally, the race-tracking region is modeled as an isotropic porous medium of permeability K_{rt} and porosity φ_{rt}. To account for a variable magnitude of the edge effect, the permeability of the race-tracking region is bounded between two limiting cases: no edge effect ($K_{rt} = K_{f1}$) and full edge effect ($K_{rt} = K_{rt,max} = 10^{-8}$ m^2).

The parametric SCRIMP model requires 18 process parameters to run a flow simulation. To explore a sufficiently large domain but keep the computational efforts reasonable, some process parameters were varied over specific ranges while others were kept constant during the generation of the synthetic dataset. However, note that these parameters could also be varied and incorporated additional input features in future work. The fixed parameters were:

- the pressure at the inlet and at the outlet: $P_{inlet} = 1 \times 10^5$ Pa and $P_{outlet} = 0$;
- the viscosity of the resin: $\mu = 0.1$ Pa · s;
- the planar dimensions of the preform: $L_f = 100$ mm and $W_f = 100$ mm;
- the DM thickness and porosity: $H_{DM} = 1$ mm and $\varphi_{DM} = 0.8$;
- the race tracking region width and porosity: $W_{rt} = 2$ mm and $\varphi_{rt} = 0.9$.

In total, 9 varying parameters were investigated: the preform thickness, permeabilities and porosity; the DM length, width and permeability; and the race-tracking region permeability. To facilitate adaptability and reduce the magnitude difference among parameters, different transformations were adopted:

- The length and width of the DM were normalized by the preform respective dimensions.
- The preform thickness was normalized by its length.
- The DM permeability and the first principal permeability of the preform were log-transformed. The other two principal permeabilities were normalized by K_{f1}.

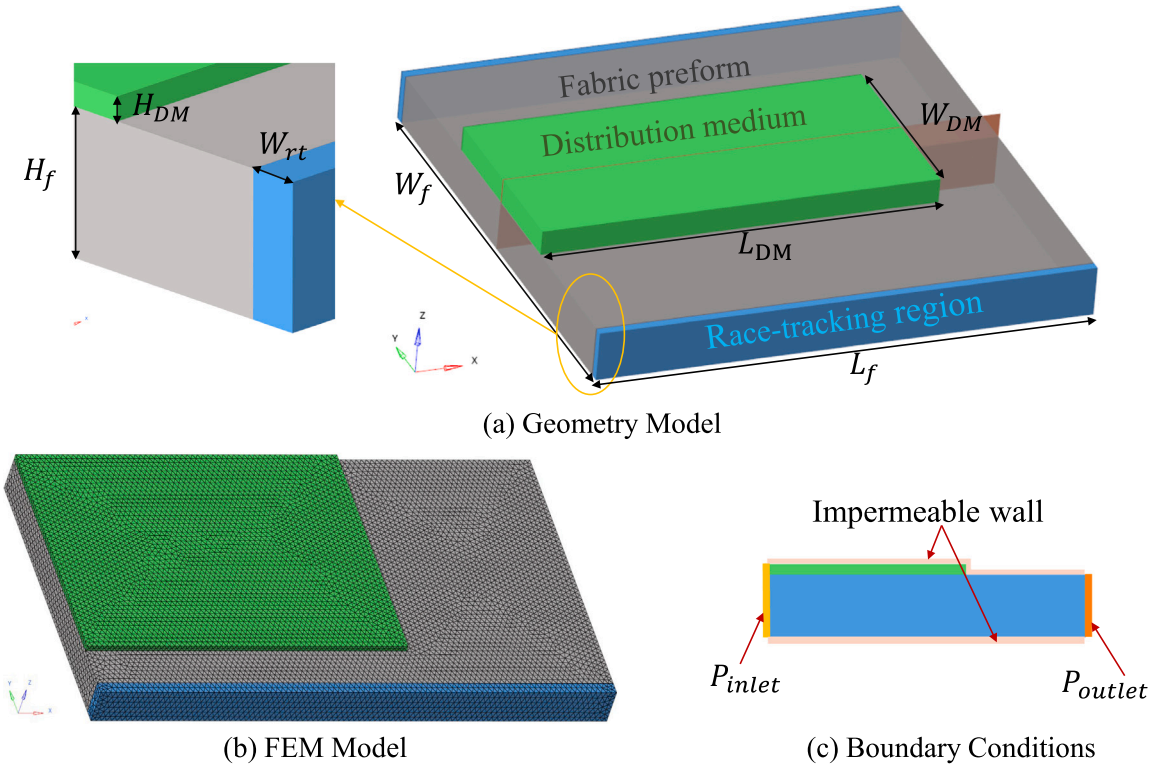

Fig. 2. General description of the test case. (a) Geometric model, (b) discretized finite element model and (c) boundary conditions.

Table 1
Summary of process parameters used in synthetic dataset generation.

Parameters	α_1	α_2	α_3	β_1	β_2	β_3	φ_f	γ_1	γ_2
Range	(0.1,1)	(0.1,1)	(0.01,0.1)	(−12,−10)	$(\frac{1}{3},3)$	(1,10)	(0.4,0.6)	(−9,−8)	(0,1)
Definition	$\frac{L_{DM}}{L_f}$	$\frac{W_{DM}}{W_f}$	$\frac{H_f}{L_f}$	$log(K_{f1})$	$\frac{K_{f1}}{K_{f2}}$	$\frac{K_{f1}}{K_{f3}}$	φ_f	$log(K_{DM})$	$\frac{K_{rt}-K_{f1}}{K_{rt,max}-K_{f1}}$

- The permeability of the race-tracking region was parameterized with a dimensionless input γ_2 and computed with a rule-of-mixture: using $K_{rt} = K_{f1} + \gamma_2 \times (K_{rt,max} - K_{f1})$.

The investigated process parameters, along with their ranges and definitions, are summarized in Table 1. These parameters were selected to provide a reasonably broad range representative of real-world scenarios [37–40]. In summary, each flow case can be characterized by 9 dimensionless parameters:

$$\chi_o = (\alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3, \varphi_f, \gamma_1, \gamma_2) \tag{3}$$

Defining an appropriate parameter space is essential for generating a sufficiently diverse dataset that supports model training and enhances generalization to unseen cases. Latin hypercube sampling (LHS) [41,42], a stochastic random method, was used to generate 2000 instances of χ_o. In LHS, the variable range is divided into multiple intervals and one sample per interval is selected, ensuring uniform coverage of the parameter space. All variables were assumed to follow uniform probability distributions. The input features χ_o were then preprocessed using standard scaling to ensure consistency across variables with different units and magnitudes. All numerical simulations were performed on an Intel(R) Core(TM) i7-8700 CPU @ 3.2 GHz and required approximately 5 min per case.

2.4. Output data preprocessing

Upon completion of the simulations, nodal information and corresponding spatial coordinates were collected and formatted for further preprocessing. The original nodal data results are shown in Fig. 3(a). Due to the nonuniform mesh discretization, the nodes were distributed in a sparse and irregular manner. All nodes on the lowest plane were selected as the bottom surface (Y_{bot}). Nodes on the top surface of the DM and the uppermost nodes of the fabric (excluding the DM) were combined to form the top surface (Y_{top}). Since CNNs require structured, grid-like format data, the simulation data from PAM-RTM must be transformed into a compatible representation. As illustrated in Fig. 3(b), interpolation was applied to resample the data onto a regular grid via standard computational techniques [43], yielding a uniform representation. Gaps in the interpolated field, typically caused by sparse or uneven meshing, were filled by using a k-nearest neighbor imputation method. This approach leverages local spatial information to estimate missing values while preserving continuity and contextual consistency. The resulting output, $Y \in \mathbb{R}^{2\times50\times100}$, consists of two 2D grids, each with a resolution of 50 pixels in height and 100 pixels in width. One grid represents the top surface, and the other represents the bottom surface. These outputs were then normalized using min–max scaling to ensure consistent data ranges and to facilitate stable and efficient CNN training.

3. Surrogate modeling

A specific CNN-based surrogate model, named FlowCastNet, was developed to predict the filling time distribution on top and bottom surfaces ($Y \in \mathbb{R}^{2\times50\times100}$), based on the dimensionless process parameters ($\chi_o \in \mathbb{R}^{1\times9}$). This section outlines the overall architecture of FlowCastNet and details the loss function and hyperparameters used during model training.

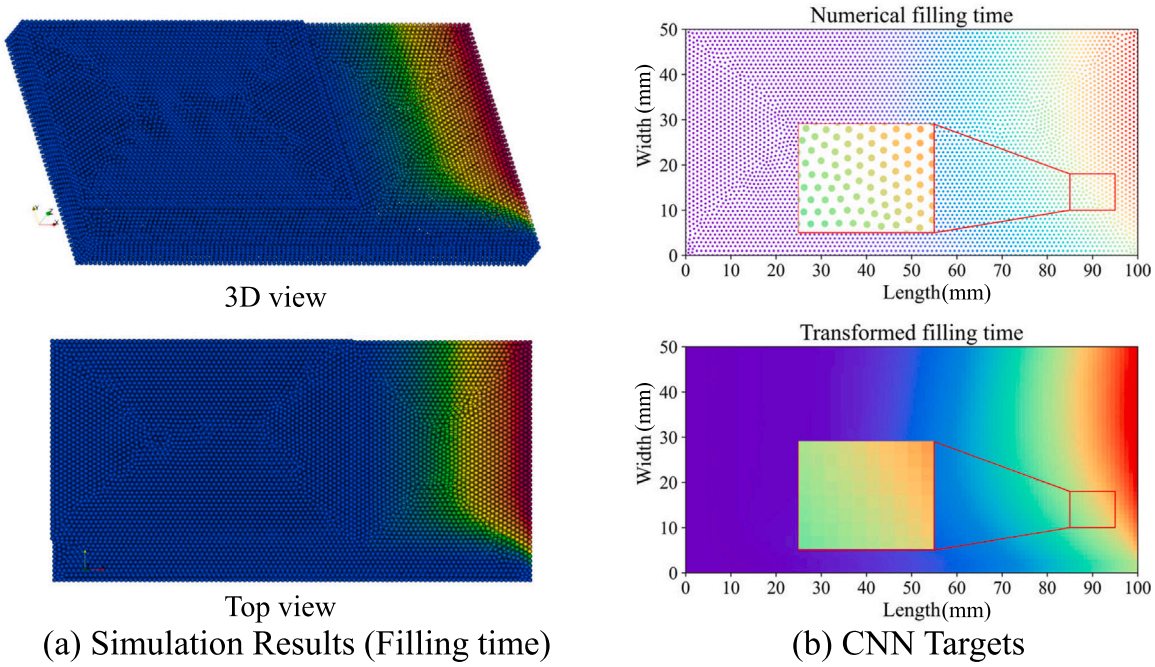

Fig. 3. (a) Original nodal simulation results. (b) Transformation of simulation results into a standardized image format. The upper panel displays the original finite element node distribution and corresponding filling results. The lower panel presents the interpolated grid representation.

3.1. Architecture design

As presented in Fig. 4, FlowCastNet is divided into two main modules: the input features encoding module and the CNN backbone. The original input, χ_o, does not naturally conform to a standard image-like structure, which limits its compatibility with convolutional architectures. To address this, a learnable embedding module was employed to project the input into a structured latent feature map. Specifically, the input vector is first flattened and passed through a two-layer multilayer perceptron (MLP) that encodes it into a latent embedding space. This MLP expands the input into an intermediate embedding with a dimensionality of 64, capturing nonlinear feature interactions. The activation function $ReLU$ [44] introduces nonlinearity into the MLP, helps regulate the output, and prevents issues such as the vanishing gradient problem. Subsequently, the latent embedding was reshaped into a compact feature tensor of shape $\mathbb{R}^{64\times1\times1}$ and then passed through a series of transposed convolutional layers to generate a structured spatial representation. This transformation effectively embeds the non-spatial input into a latent 2D manifold, allowing subsequent convolutional modules to exploit local spatial correlations while preserving the global information encoded by the MLP. Through this feature encoding module, the FlowCastNet framework is rendered agnostic to the dimensionality of the original input features. This design enhances the model's adaptability to evolving requirements as additional input variables can be implemented without changing the architecture of FlowCastNet. It also facilitates efficient retraining or transfer learning without requiring structural modifications.

Following the input embedding stage, FlowCastNet adopts a hierarchical convolutional backbone that incorporates residual learning, attention mechanisms, and feature-wise conditioning. A typical CNN architecture is composed of multiple building blocks, such as convolutional and pooling layers, and is designed to automatically and adaptively learn spatial hierarchies of features through backpropagation. The early layers capture local features such as edges and textures, whereas the deeper layers detect more abstract patterns, culminating in high-level feature representations. Each stage in the CNN backbone consists of a transposed convolutional layer, followed by a Residual Convolutional Block with Convolutional Block Attention Modules (ResidualCBAM) [45] and a Feature-wise Linear Modulation (FiLM) layer [46], which jointly perform feature upsampling and attention-based refinement. The CBAM mechanism combines spatial attention and channel attention to adaptively emphasize informative features and suppress irrelevant responses. Specifically, channel attention utilizes global average pooling and max pooling to capture complementary statistics across channels, while spatial attention leverages pooled spatial descriptors to localize salient regions. Together, these two attention branches enable the model to dynamically recalibrate intermediate feature maps, thereby enhancing interpretability and robustness against noise. Moreover, to support adaptive conditioning on external scalar information, each residual stage can optionally incorporate a FiLM layer. The FiLM module generates instance-specific scaling and shifting parameters (γ, β) through a lightweight MLP, thus allowing the network to dynamically adjust its internal feature distributions in response to contextual or conditional inputs.

Throughout the development of FlowCastNet, multiple architectural components were evaluated to balance expressive power with network complexity. Each convolutional layer was tailored with specific channel configurations and kernel sizes to extract a wide range of features from the input. Additionally, transposed convolutional layers were incorporated in FlowCastNet, as they were found to yield superior predictions compared to upsampling layers, which is consistent with previous findings [47].

After passing through several residual-attention stages, the model employs a final pooling operation to consolidate multi-scale features. As shown in Fig. 4, the pooling layers were incorporated into the final convolutional blocks. Although the activation functions helps mitigate the vanishing gradient problem, it often creates regions with very small values, which are known as "nearly dead spaces". These low-activation areas can hinder the network's ability to effectively learn from the data. This issue is particularly critical when predicting filling time distributions, which inherently contain near-zero values corresponding to the initial stage of the flow. In this context, pooling layers can be beneficial because they aggregate and propagate near-zero activations, allowing more pixels to contribute meaningfully to the network computations. Additionally, pooling layers serve as fundamental components in CNNs by reducing the spatial dimensions of

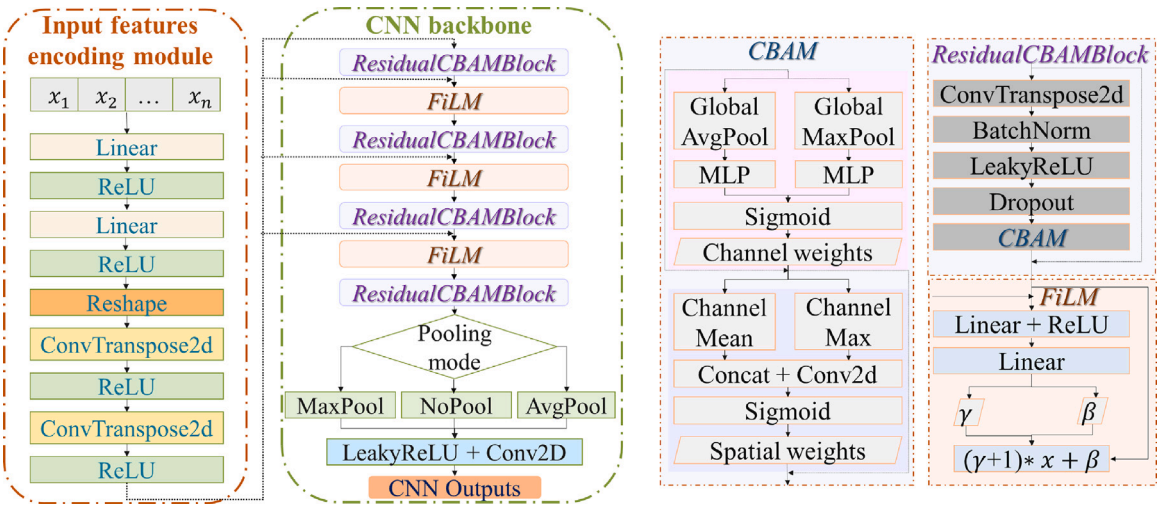

Fig. 4. Diagram of the FlowCastNet architecture designed to forecast filling time distribution based on scalar processing variables.

feature maps. This dimensionality reduction decreases the number of trainable parameters and computational cost, mitigates overfitting, and preserves the hierarchical structure of spatial features. To assess the impact of pooling strategies, average pooling (AvgPool), max pooling (MaxPool), and no pooling (NoPool) were systematically compared in terms of predictive performance.

The NoPool approach omits pooling layers entirely and thereby preserves the original spatial dimensions of feature maps throughout the network. Max pooling, denoted as Y_{maxpool}, extracts the maximum value from each localized pooling region and emphasizes the most dominant features in the feature maps. This method is particularly adept at identifying distinct characteristics such as edges, sharp transitions, and textures, which are crucial in image-based tasks. In contrast, average pooling, Y_{avgpool}, involves calculation of the mean value within each pooling region, producing a smoothed and downsampled feature map. This approach introduces a modest degree of translation invariance while maintaining the overall feature distribution, making it less aggressive in eliminating subtle spatial details. The pooling strategies are summarized as follows:

$$Y_{\text{maxpool}} = \max\left\{\zeta_i\right\}_{i=0}^{N_{pool}} \tag{4}$$

$$Y_{\text{avgpool}} = \frac{1}{N_{pool}} \sum_{i=1}^{N_{pool}} \zeta_i \tag{5}$$

where ζ_i represents the pixel values within the local pooling region, and N_{pool} represents the total number of pixels in the region.

3.2. Loss functions

The primary objective of training a machine learning model is to determine the optimal set of weights (θ_w) and biases (θ_b) that minimize prediction error. During training, the computed loss value $\mathcal{L}(\theta)$ guides the iterative adjustment of weights and biases via backpropagation:

$$\theta_{opt} = arg\min_{\theta\in\Theta} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(\theta) \tag{6}$$

Here, n denotes the total number of training samples, and Θ represents the parameter space encompassing all feasible values of weights and biases. Although θ appears on both sides of Eq. (6), the left-hand side refers to the optimized parameters obtained by minimizing the loss function on the right-hand side.

The choice of an appropriate loss function $\mathcal{L}$ is critical, as it directly influences the learning performance and prediction accuracy. To jointly preserve structural similarity and value fidelity in the predicted fields, a hybrid loss function was employed, combining the Structural Similarity Index (SSIM) [48] $\mathcal{L}_{SSIM}$ with a weighted L1 loss $\mathcal{L}_{wL1}$:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{SSIM} + \lambda_2 \mathcal{L}_{wL1} \tag{7}$$

where λ_1 and λ_2 are scalar weighting coefficients controlling the relative importance of structural and pixel-level terms, respectively. $\mathcal{L}_{SSIM}$ is incorporated as a perceptual similarity metric that complements pixel-wise loss functions. Unlike purely numerical metrics, SSIM mimics human visual perception by evaluating structural information, including luminance, contrast, and structural similarity. Its mathematical formulation is provided by:

$$\mathcal{L}_{SSIM} = 1 - \frac{(2\mu_{y_i}\mu_{\hat{y}_i} + c_1)(2\sigma_{y_i\hat{y}_i} + c_2)}{(\mu_{y_i}^2 + \mu_{\hat{y}_i}^2 + c_1)(\sigma_{y_i}^2 + \sigma_{\hat{y}_i}^2 + c_2)} \tag{8}$$

where y_i and $\hat{y}_i$ denote the ground truth and predicted values at the ith pixel. μ_{y_i} and $\mu_{\hat{y}_i}$ are the average pixel values, $\sigma_{y_i}^2$ and $\sigma_{\hat{y}_i}^2$ the variances, and $\sigma_{y_i\hat{y}_i}$ denotes the covariance between y_i and $\hat{y}_i$. Constants $c_1 = 1 \times 10^{-4}$ and $c_2 = 9 \times 10^{-4}$ are used to stabilize the division operation. Consequently, the SSIM loss ranges from 0 to 1, with lower values indicating greater similarity between images.

Due to the existence of the race-tracking region and the DM, most pixel values in the dataset are concentrated in low-value regions. This leads to a long-tailed distribution, in which rare but physically important pixels are statistically underrepresented. If a conventional L1 or L2 loss is applied, all pixels contribute equally to the training objective. To emphasize high-magnitude or rare-value regions in the target field, a piecewise weighting strategy is adopted in the L1 component:

$$\mathcal{L}_{wL1} = \frac{1}{N_{pixel}} \sum_{i=1}^{N_{pixel}} w(y_i) \left|\hat{y}_i - y_i\right|, \quad w(y_i) = \begin{cases} w_{\text{low}}, y_i \le y_{\text{threshold}}, \\ w_{\text{high}}, y_i > y_{\text{threshold}}, \end{cases} \tag{9}$$

where N_{pixel} represents the total number of pixels and $w(y_i)$ is a value-dependent weight. In this work, the weight parameters are empirically set as $y_{\text{threshold}} = 4.0$, $w_{\text{low}} = 1.5$ and $w_{\text{high}} = 4.0$. Overall, such design increases the gradient contribution of high-value pixels during backpropagation, enabling the network to focus on rare but critical regions without excessively distorting low-value areas.

To assess the influence of the loss function on the predictive accuracy of the model, different combinations of weighting coefficients were considered with $\lambda_1 \in [0.3, 1]$ and $\lambda_2 \in [0.5, 7]$. For comparison, simple loss functions $\mathcal{L}_{SSIM}$ and $\mathcal{L}_{wL1}$ were also evaluated. Combined with the three possible pooling strategies, this resulted in 66 different variants of FlowCastNet. All models were trained under similar conditions as described in the following subsection.

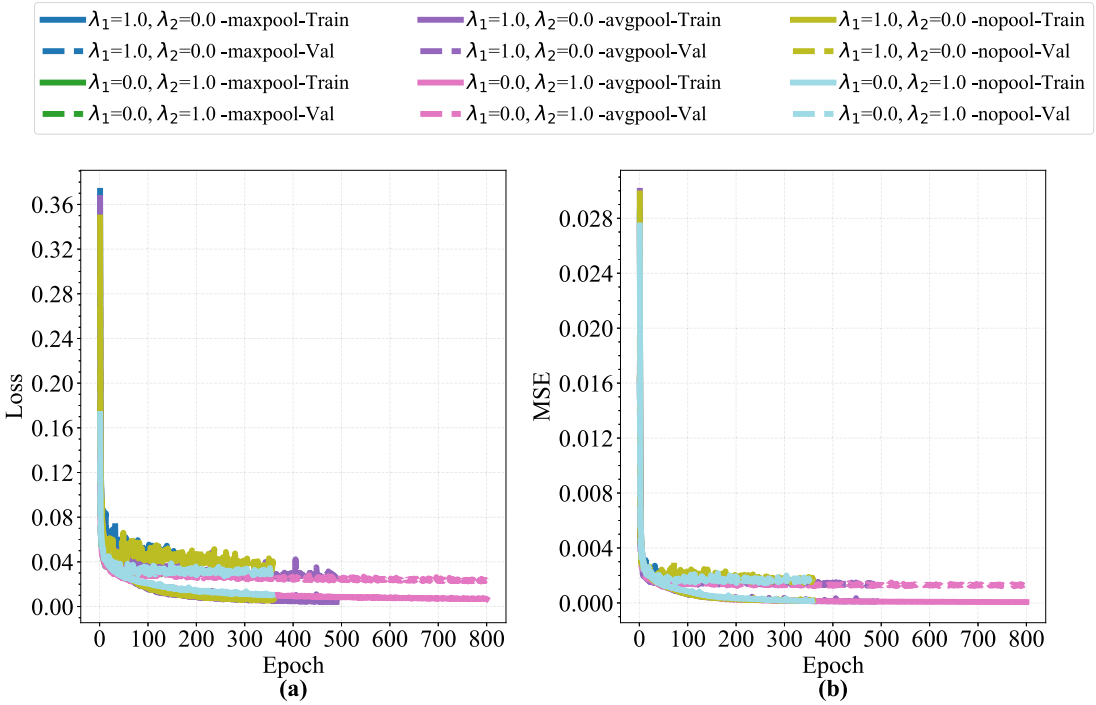

Fig. 5. Training progress of selected FlowCastNet variants with different pooling strategies and loss function weight factors.

3.3. Training setting

The synthetic dataset was randomly divided into three subsets using a fixed global random seed to ensure reproducibility and eliminate selection bias across different FlowCastNet variants. Specifically, 70% of the data were allocated for training, 20% for validation to prevent overfitting and optimize hyperparameters, and the remaining 10% for testing to assess generalizability and predictive accuracy. The FlowCastNet architecture, including variants with different pooling layers, comprises approximately 3.7 million trainable parameters.

Training was conducted in a Python 3.11 environment on a server equipped with an NVIDIA H100 80 GB HBM3 GPU. FlowCastNet was implemented in *PyTorch* [49] and trained for 800 epochs. Early stopping was applied if no improvement was observed after 50 epochs to prevent overfitting. Each training session took approximately 10 to 20 min. The Adaptive Moment Estimation (Adam) optimizer was employed to fine-tune the network weights and biases. A learning rate of 1.5×10^{-4} and a batch size of 16 were selected through empirical tuning.

4. Results and discussion

4.1. Training history

Fig. 5(a) shows the hybrid loss progression during training using three different pooling strategies and two weight factors combinations. Only six FlowCastNet variants are shown here, as the learning curves of all other models exhibited similar trends. It should be noted that the two components of the loss function operate on distinct numerical scales, which complicates the direct comparison between models with different weight factors. Consequently, the mean squared error (MSE) was used as a unified performance metric to enable consistent evaluation of different variants. MSE evolution curves are reported in Fig. 5(b) for the six selected FlowCastNet variants.

Both loss and MSE decreased sharply during the initial epochs and gradually stabilized after approximately 100 epochs. This trend reflects a steady improvement of the model, with predictions progressively aligning more closely with the ground truth over the epochs. As expected, the validation loss remained consistently higher than the training loss, as FlowCastNet became increasingly specialized in fitting the training data while naturally performing slightly worse on unseen validation samples. Both training and validation losses declined steadily, without widening gap or late-stage degradation in validation performance. This consistent behavior, reinforced by early stopping, emphasizes the absence of overfitting and indicates that the model possesses some generalization capability.

4.2. Performance evaluation

The predictive performance of the model was evaluated on the independent test dataset consisting of 200 simulations that remained untouched during training. To ensure an unbiased evaluation, the MSE obtained on the entire test dataset was first compared for the different variants to identify the best model. As shown in Fig. 6, the MSE values range from 0.001376 to 0.00237,indicating no substantial performance difference between the variants. Moreover, the results do not reveal an optimal pooling strategies and no clear trend is visible on the impact of weight factors (λ_1, λ_2). Nevertheless, it can be observed that a single SSIM or L1 loss (i.e., configurations where one of the λ is zero) consistently failed to yield optimal results for each pooling strategy. Importantly, Fig. 5 further demonstrates that, even for extreme weighting settings such as (λ_1, λ_2) = (1,0) and (0,1), the loss magnitudes during training converge to comparable numerical ranges, confirming that no severe scale imbalance exists between the SSIM and weighted L1 components. Hence, the relatively flat performance landscape of Fig. 6 is not an artifact caused by scale mismatch between the loss function components. This supports the hypothesis that a weighted combination of loss terms more effectively captures both pixel-wise and structural information in flow prediction. Overall, the lowest MSE were achieved by the AvgPool variant with $\lambda_1 = 0.3$ and $\lambda_2 = 3.0$. This empirically optimal configuration was therefore selected as a representative model for all subsequent analyses presented in this paper.

In Fig. 7(a), the predictions of the selected model are compared with ground truth values (i.e., flow simulation results) for all pixels included in the entire test dataset. As can be seen, some systematic positive of negative deviation from the reference line ($\hat{y} = y$) exist. However, this

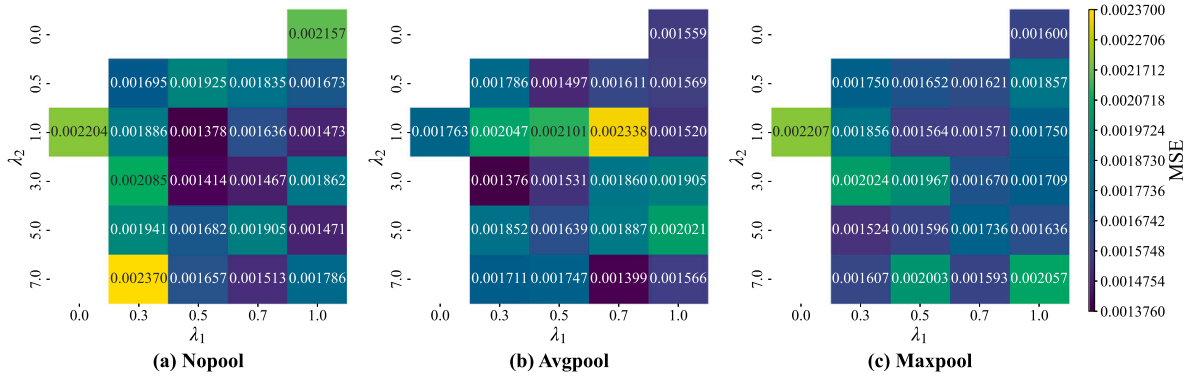

Fig. 6. Evaluation of FlowCastNet performance on the test dataset for different combinations of loss functions and pooling strategies.

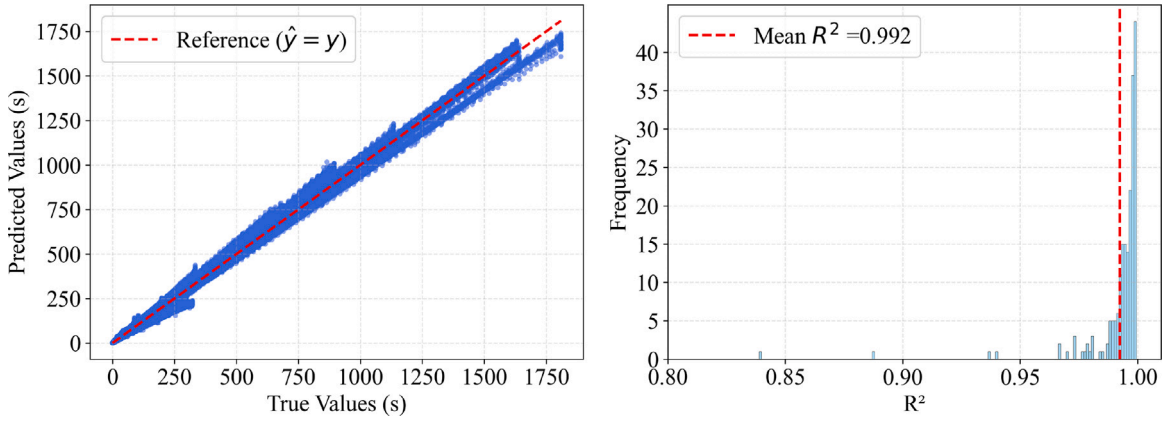

Fig. 7. Predicted versus actual values over the entire test dataset and distribution of per-sample R^2 for the optimal FlowCastNet variant (λ_1 = 0.3, λ_2 = 3.0, AvgPool).

effect is rather limited and it can be concluded that FlowCastNet maintains a satisfactory accuracy over the entire dataset. It is interesting to note that the test dataset covers a significant domain with some process parameters varying by more than one order of magnitude. Despite this, the model exhibits a consistent behavior with no pronounced bias for either short or long filling times. To further evaluate the predictive performance, the R-squared (R^2) metric was computed for each simulation of the dataset. The resulting distribution reported in Fig. 7(b) shows that the majority of test instances are predicted with low residual errors, as most samples are clustered between the mean value of 0.992 and 1. Such narrow spread indicates good generalization and low variance among samples, implying that the selected model configuration effectively balances reconstruction and regularization terms. It also shows that the predictive accuracy was much lower in few cases, which yielded R^2 values below 0.95. To further investigate the predictive capability and limitations of FlowCastNet, the best- and worst-performing test cases were analyzed in more details.

Fig. 8 presents the filling time distributions from both FlowCastNet and PAM-RTM, along with the corresponding relative error for the best-performing case of the test dataset. Almost no difference can be visually detected in the filling time distribution for both top and bottom surfaces. The model adequately captures the initial filling of the DM-covered region and the race-tracking channel as well as the delayed impregnation of the uncovered region away from the edge effect. Analysis of the error map reveals no significant trend except in a localized zone near the outlet and close to the race-tracking channel. In that region, a steep filling time gradient exists, which seems to cause a transition from an underestimation to an overestimation of the filling time. Apart from that local deviation, the relative error remains below 5% everywhere else.

In contrast, the lowest R^2 case exhibits noticeable discrepancies between the prediction and ground truth, with relative error exceeding 20% in some significant regions (see Fig. 9). These deviations are likely caused by insufficient representation of such rare flow regimes in the training data or by artifacts introduced by the deconvolutional operations [50]. Nevertheless, even in this case, the global flow pattern and overall filling-front geometry are well preserved by the FlowCastNet's prediction.

To investigate further the potential dominance of individual parameters, a sensitivity analysis was conducted using the full dataset. Fig. 10(a) shows the mean R^2 values computed over discretized intervals of each input parameter, while Fig. 10(b) reports the corresponding Spearman rank correlation coefficients. The mean-trend R^2 analysis shows that performance degradation near parameter boundaries occurs gradually and only for certain parameters($\alpha_1, \alpha_3, \beta_1$ and γ_1), while others show no clear boundary-related trends. The most influential parameters regarding model accuracy are: α_1 (normalized DM length) ($\rho = -0.384$) and preform permeability $\beta_1 = logK_{f1}$ ($\rho = 0.295$). This indicates that geometric coverage of the DM and preform permeability play a significant role in influencing model performance. Larger DM coverage leads to sharper spatial transitions at the interface between the DM and the preform, which are inherently more challenging for convolutional kernels to reconstruct accurately.

Despite these localized challenges, the mean R^2 remains consistently high across the entire parameter space, exceeding 0.985 for all parameter intervals. Poor-performing cases, such as the one presented in Fig. 9 appear to come from rare parameter combinations rather than systematic model failure. The limited performance degradation near parameter boundaries could be further mitigated in future work by enriching the training dataset with additional samples targeting extreme parameter combinations.

$$\chi_o = [0.300, 0.654, 0.030, -10.756, 1.704, 3.011, 0.501, -8.465, 0.277]$$

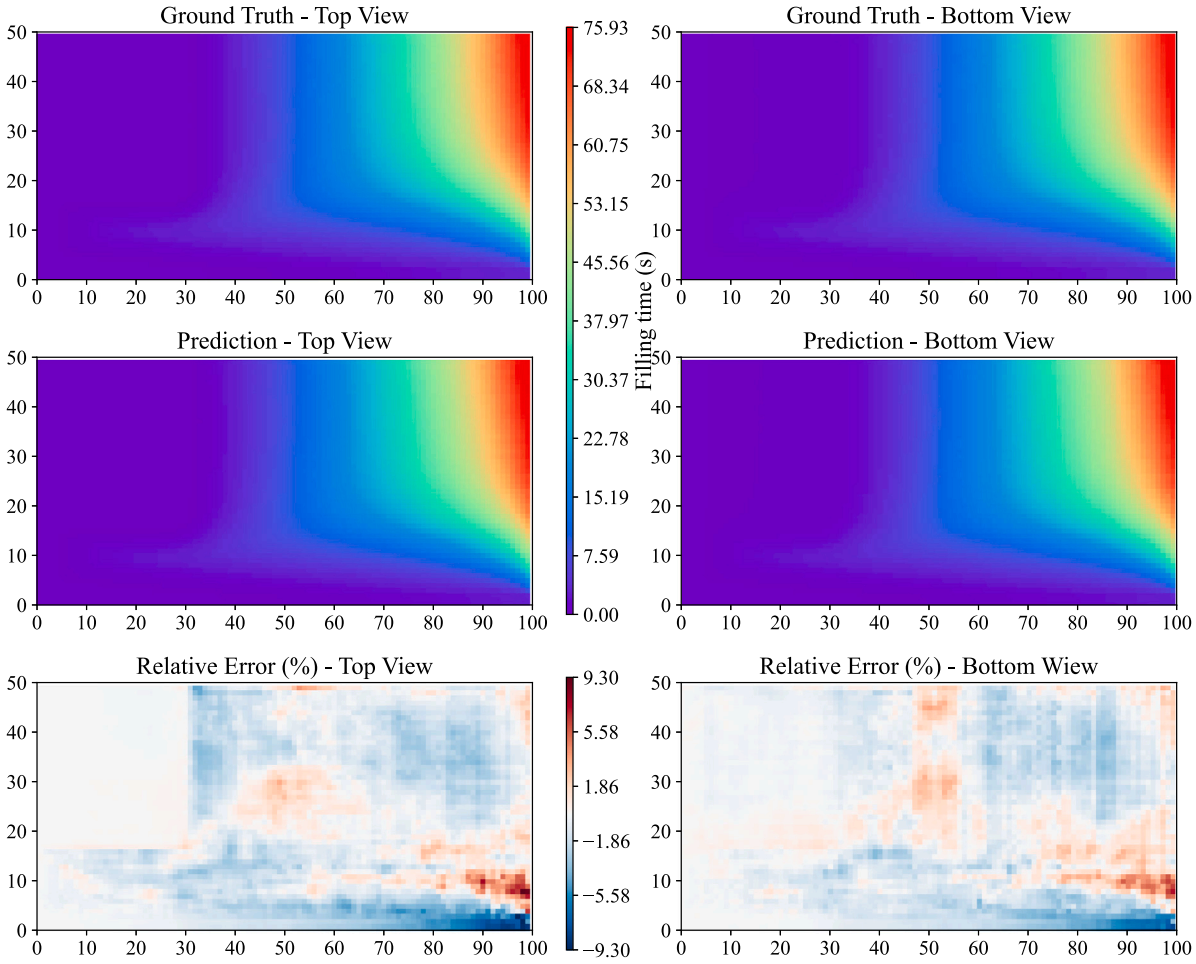

Fig. 8. Predicted and ground truth filling-time fields, along with the relative error map, for the best-performing test case (highest $R^2 = 0.9995$).

4.3. DM layout optimization

Compared to physics-based simulation, FlowCastNet can provide extremely fast predictions with an inference time of approximately 1 ms. In this section, we investigate the feasibility of using the model as a quick optimization tool to help configure the DM layout under specific manufacturing conditions.

4.3.1. Objective functions

To illustrate the use of FlowCastNet in the context of multi-criteria process optimization, two conflicting scalar objectives were derived from the model predictions. The first objective is the total filling time (t_{total}), which is simply defined as the maximum filling time considering both top and bottom surfaces. In practice, the total filling time should be minimized to avoid premature gelation of the resin and improve production efficiency. The second objective is the time-averaged lead–lag distance, denoted $\overline{LL}$, which intents to capture the uniformity of the flow filling pattern. To compute it, we first consider the raw predictions of the model, which gives the filling times at each position of the two surfaces: $\text{top}[y, x]$ and $\text{bot}[y, x]$. At a given time t_i, for each row index y, the longitudinal positions of the flow-front on both surfaces are first determined as follows:

$$\begin{aligned} x^{top}_{front} &= max\left\{x \mid top[y, x] \le t_i\right\} \\ x^{bot}_{front} &= max\left\{x \mid bot[y, x] \le t_i\right\} \end{aligned} \tag{10}$$

The lead–lag distance at each row y_n is then defined as the difference between the top and bottom front positions (see Fig. 11):

$$\Delta x_{y_n} = x^{top}_{front} - x^{bot}_{front} \tag{11}$$

For each time t_i, the instantaneous lead–lag distance $LL(t_i)$ is subsequently obtained by averaging across all N_y rows:

$$LL(t_i) = \frac{1}{N_y}\sum_{n=1}^{N_y} \Delta x_{y_n} \tag{12}$$

Finally, the intensity of the lead–lag phenomenon is reduced to a single scalar parameter by computing the time-averaged distance:

$$\overline{LL} = \frac{1}{M}\sum_{i=1}^{M} |LL(t_i)| \tag{13}$$

where M denotes the number of discrete time samples, which is set to 100 in this work.

Fig. 12 illustrates the lead–lag analysis for a specific case, randomly selected in the synthetic dataset. At the initial stage, the presence of the DM layer on the top surface causes the resin to flow significantly faster than in the bottom layer, resulting in a large lead–lag value. As the flow progresses and velocity decreases, the difference between the two layers diminishes, causing $LL(t_i)$ to gradually decrease and stabilize close to zero during the second half of the filling process. As mentioned previously, this study considers the time-averaged lead–lag distance $\overline{LL}$ as an objective to minimized (see next two subsections). However, note that this is only an illustrative example and other design criteria could be considered from the field values predicted by the model. For

$$\chi_o = [0.702, 0.559, 0.080, -11.201, 1.443, 3.094, 0.516, -8.008, 0.239]$$

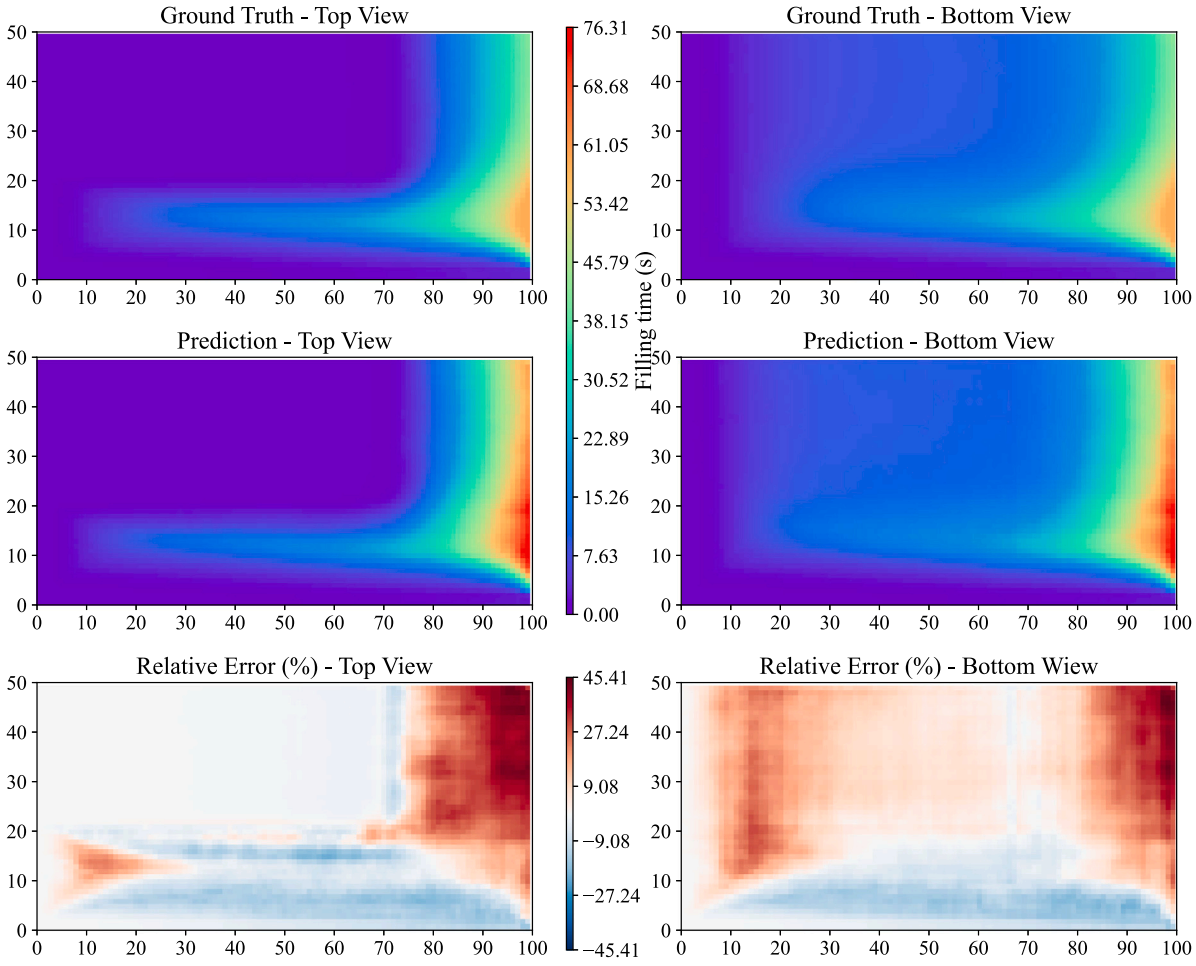

Fig. 9. Predicted and ground truth filling-time fields, along with the relative error map, for the worst-performing test case (lowest $R^2 = 0.8387$).

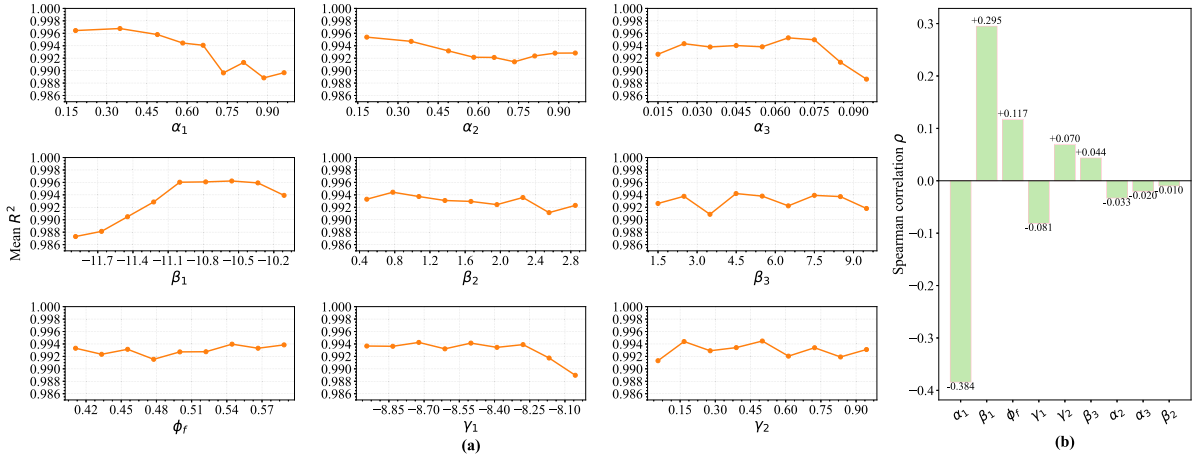

Fig. 10. Sensitivity and importance of input parameters with respect to model performance. (a) Mean coefficient of determination (R^2) computed within quantile-based bins of each input parameter. (b) Spearman rank correlation coefficients (ρ) between individual parameters and R^2, ranked by their absolute values to highlight relative parameter importance.

example, the risk of dry spot creation could also be assessed by the maximum instantaneous lead–lag distance or filling time gradients (in both longitudinal and transverse directions). An estimation of the resin waste could also be obtained from the time distribution at the outlet edge.

4.3.2. Scenario 1

In this first example, we consider the bi-objective optimization problem for a specific manufacturing case with the following fixed process parameters:

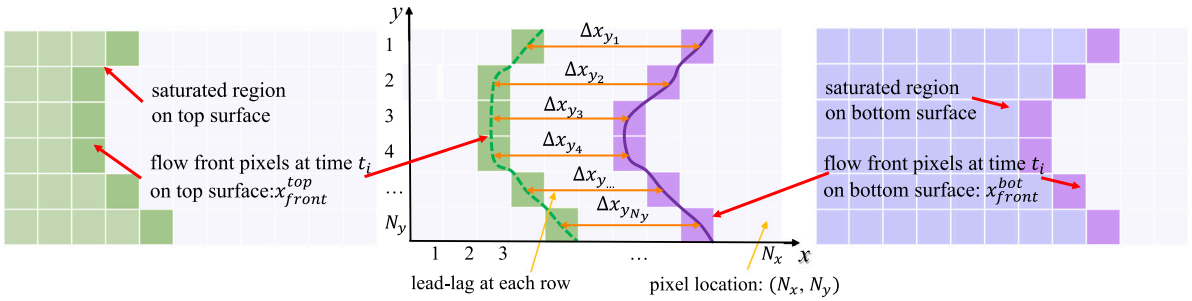

Fig. 11. Definition of the lead–lag distance between the top and bottom surfaces, based on the flow-front pixel positions.

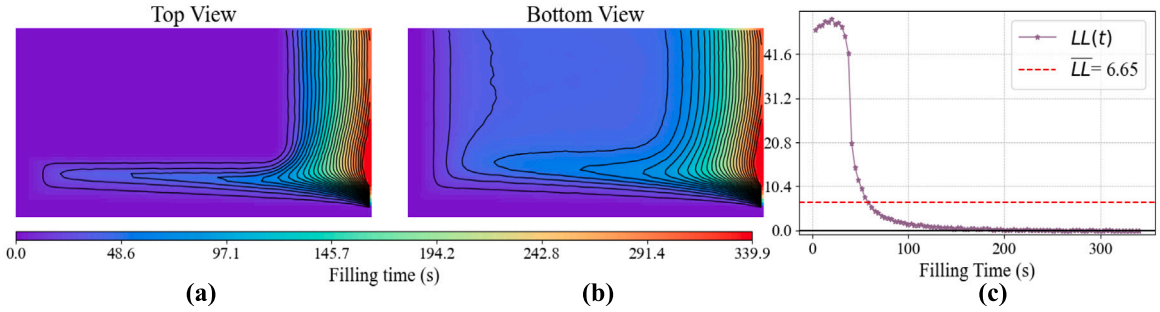

Fig. 12. Illustration of the lead–lag analysis for a representative test case: Filling-time distributions on the (a) top and (b) bottom surfaces. (c) Corresponding temporal evolution and time-averaged value of the lead–lag distance during the filling process.

- Preform permeability and porosity: $K_{f1} = K_{f2} = 4 \times 10^{-11}$ m^2, $K_{f3} = 4 \times 10^{-12}$ m^2, $\phi_f = 0.5$
- Part thickness: $H_f = 9$ mm
- DM permeability: $K_{DM} = 8 \times 10^{-9}$ m^2
- Magnitude of the race-tracking phenomenon: $\gamma_2 = 20\%$ corresponding to $K_{rt} = 2.032 \times 10^{-9}$ m^2

The influence of the DM configuration is investigated for parameters α_1 and α_2 within the range [0.5, 0.9]. This interval was selected to provide significant variations of the objective functions while avoiding insufficient or excessive DM coverage, which could lead to extreme values in one of the criteria.

A first exploration of the design space was conducted with a small full factorial (FF) plan with increments of 0.1 for each dimensionless parameters. The resulting filling time distributions obtained with PAM-RTM and FlowCastNet are reported Fig. 13 and Fig. 14, respectively. As shown, the filling pattern predicted by the surrogate model is in good agreement with the flow simulations for all 25 cases considered. As expected, faster filling occurs in the DM and in the race-tracking region. In the top surface, the shape of the flow front is significantly affected by the DM configuration. With a narrow DM, increasing its length leads to a sharp V-shaped profile in the filling pattern. When the width of the DM increases, the effect is gradually shifted towards the edge and it completely disappears for the largest coverage considered ($\alpha_1 = \alpha_2 = 0.9$). In the latter case, filling of the DM dominates the race-tracking phenomenon, resulting in delayed filling at the edge of the part. On the bottom surface, the edge of the laminate influenced by race-tracking always fills first. Filling times decrease steadily when the DM size increases while the overall pattern remains rather similar. The main effect arises from the DM length with a reduction of the total filling time by a factor of two when α_1 increases from 0.5 to 0.9. Increasing the DM width also speeds-up the impregnation process, though to a much lesser extent.

All cases form the 25 FF plan were then analyzed in terms of lead–lag following the procedure described in the previous sub-section. Results reported in Fig. 15 confirm that the DM length has the greatest influence. For short DM configurations, the instantaneous lead–lag distance gradually decreases from an initial maximum value, and filling is completed in quasi-steady state with minimal lead–lag. As the length of the DM increases, the peak in the lead–lag becomes more pronounced, and the final plateau region is shortened. Consequently, larger α_1 values lead to reduced total filling time but increased time-averaged lead–lag. Fig. 15 also shows that FlowCastNet provides good predictions of the temporal evolution although slight deviations from physic-based simulation can be observed. In most cases, FlowCastNet tends to over-predict filling times in the very late stage of the process. The surrogate model also overestimates the lead–lag distance, particularly for larger DM coverage. Despite these quantitative discrepancies, the model appears to capture the overall influence of the design parameters α_1 and α_2, on both t_{total} and $\overline{LL}$, suggesting some potential to address the optimization problem.

Subsequently, a Pareto analysis was conducted on the 25 FF dataset generated with PAM-RTM, which resulted in 15 non-dominated solutions (i.e., approximately 60% of the initial dataset). As shown in Fig. 16, this only provides a sparse description of the Pareto front with an associated Pareto set covering a large portion of the design space. To deepen the analysis, an extended FF plan, comprising 6400 cases uniformly sampled within the variables space, was investigated using FlowCastNet. Note that such an exhaustive search would be computationally prohibitive using numerical simulation. Excluding geometry generation and mesh discretization, the simulation process alone would require approximately 22 days, assuming an average runtime of 5 min per case. In contrast, FlowCastNet completed the predictions for the entire dataset in approximately 2.2 s.

As reported in Fig. 16, the 6400 FF plan led to dense representation of the Pareto front with 554 non-dominated solutions (8.7% of the initial dataset). The associated Pareto set is localized in a small region of the design space forming a narrow band around $\alpha_2 = 0.85$. This observation appears consistent with the previously discussed influences of the design variables on the objective functions. As already mentioned, the surrogate model predicts correctly the overall filling pattern but makes small quantitative errors in the optimization objectives. Consequently, the predicted Pareto front does not perfectly superimpose on the optimal solutions identified through numerical simulation. However, this does not necessarily mean that optimal design variables are not adequately captured. To test this hypothesis, a k-means clustering algorithm was used to extract 90 optimal solutions

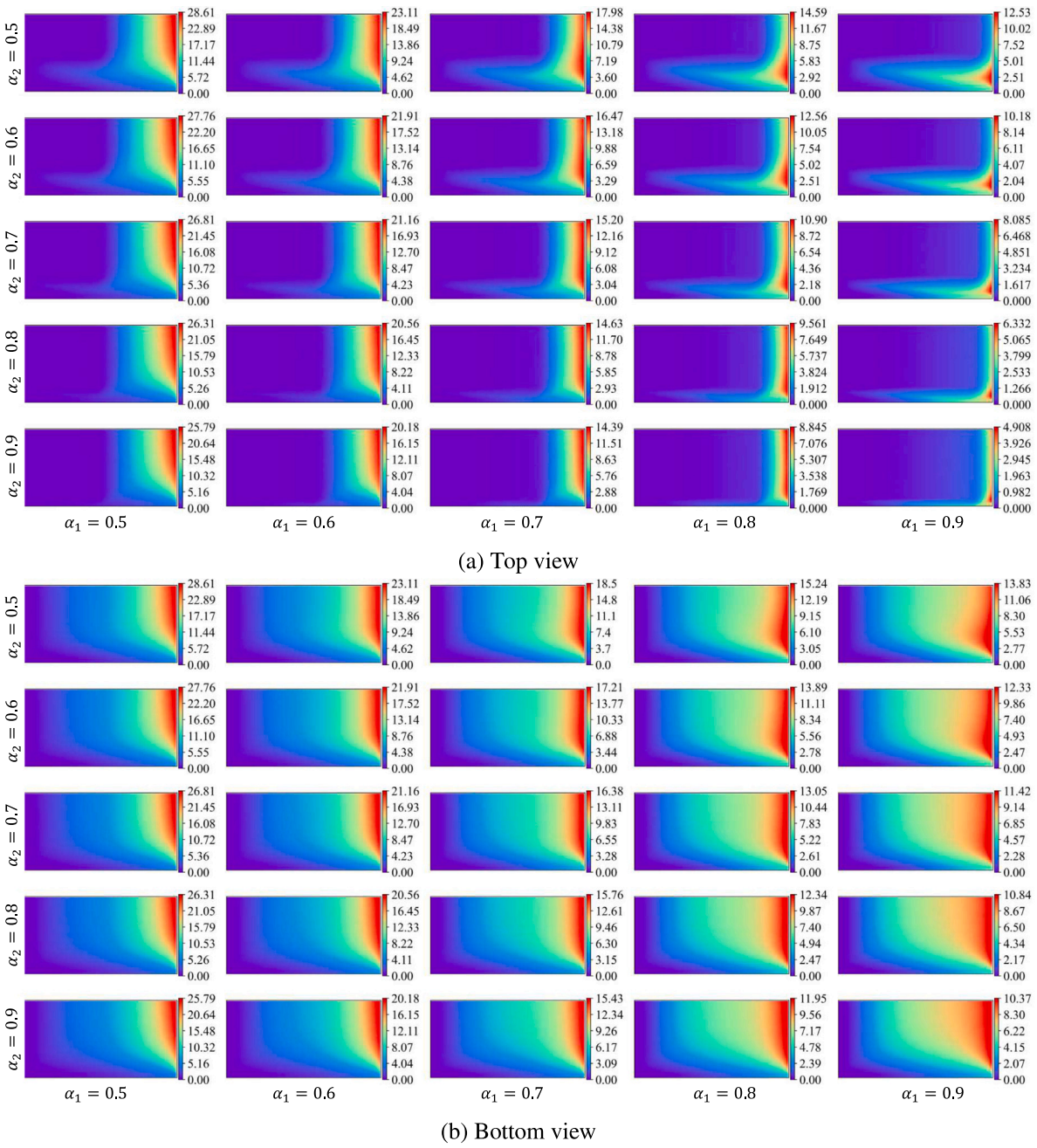

Fig. 13. Filling time distributions obtained from PAM-RTM on: (a) top view, and (b) bottom view across the 25-case full-factorial design.

from the FlowCastNet Pareto front, which were then simulated using PAM-RTM. Results are reported as magenta diamonds in Fig. 16. As can be seen, a new Pareto frontier is reconstructed that outperforms most of the previous solutions derived from the initial 25 FF dataset. Therefore, although prediction errors remain, FlowCastNet can be used as a complementary tool for rapid optimization of process parameters. Once trained, the model is capable of exhaustively exploring the design space and generating an almost unlimited number of solutions, which would not be achievable through numerical simulation.

4.3.3. Scenario 2

The previous example showed how the computational speed of FlowCastNet can be leveraged to thoroughly investigate a specific manufacturing case. Since the model has been trained with process parameters spanning significant ranges, it can also provide rapid analysis at any point within the training domain. To illustrate this capability, we now consider a change of processing conditions compared to the previous scenario:

- Preform in-plane anisotropy is introduced by increasing the transverse permeability to $K_{f2} = 8 \times 10^{-11}$ m^2
- The part thickness is reduced to $H_f = 3$ mm
- The magnitude of the race-tracking phenomenon is increased to $\gamma_2 = 50\%$ corresponding to $K_{rt} = 5.02 \times 10^{-9}$ m^2

Other process parameters are kept similar to scenario 1.

Following a similar procedure, the bi-objective Pareto front is identified from a 6400 FF search of the design space. The results, reported in Fig. 17, are compared with the analyses conducted for scenario 1. As illustrated, FlowCastNet predicts a clear difference between the two manufacturing cases. Compared to scenario 1, the Pareto front is shifted towards smaller values in both objectives, primarily due to the reduced

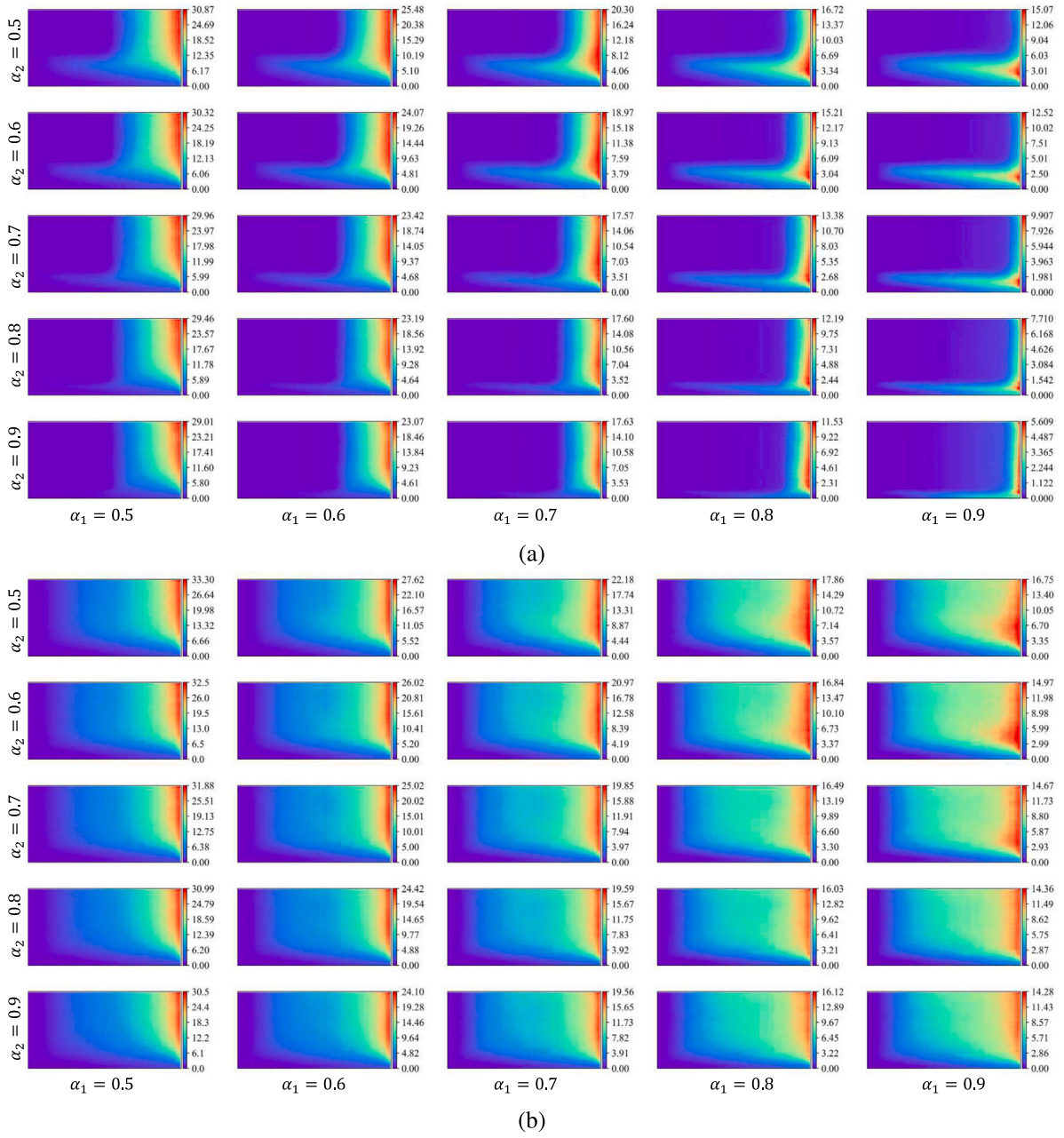

Fig. 14. Filling time distributions predicted by FlowCastNet on: (a) top view, and (b) bottom view across the 25-case full-factorial design.

preform thickness, which shortens the overall filling path. Another significant difference arises from the combined effects of increased transverse permeability and race-tracking magnitude, which promote transverse flow and alter the V-shaped region of the flow front. As a result, the Pareto set of scenario 2 is associated with lower α_2 values, particularly for longer DM.

Knowledge of the entire Pareto frontier can subsequently be used to assist mold design based on any desired trade-off between impregnation speed and flow uniformity. As an illustrative example, we consider a specific design objective, which consists of minimizing the total filling time while constraining the lead–lag below a maximum value: $\overline{LL} < 15$ mm. The corresponding optimal solutions are shown with star symbols in Fig. 17. For scenario 1, FlowCastNet predicts an optimal DM configuration with $(\alpha_1, \alpha_2) = (0.713, 0.809)$. When the processing conditions are changed to scenario 2, the DM dimensions must be adapted to account for the reduced thickness and increased transverse flow near the race-tracking channel. Consequently, the model recommends lengthening and narrowing the DM, yielding a new optimal solution $(\alpha_1, \alpha_2) = (0.875, 0.733)$.

The above example demonstrates that FlowCastNet can serve as a rapid decision-making tool by providing near-instantaneous predictions of the complete Pareto frontier across multiple manufacturing scenarios. While conventional gradient-based or heuristic optimization methods coupled with finite element (FE) solvers could be more efficient when studying a specific manufacturing case, their computational overhead scales approximately linearly with the number of cases considered. Changing a single manufacturing parameter, such as fabric permeability, typically requires complete re-optimization. This limitation becomes increasingly restrictive when exploring large and high-dimensional VARTM design spaces. In contrast, the cost associated with data generation and model training for FlowCastNet represents a one-time investment. Once trained, FlowCastNet can analyze almost any point within the training domain, enabling exhaustive searches

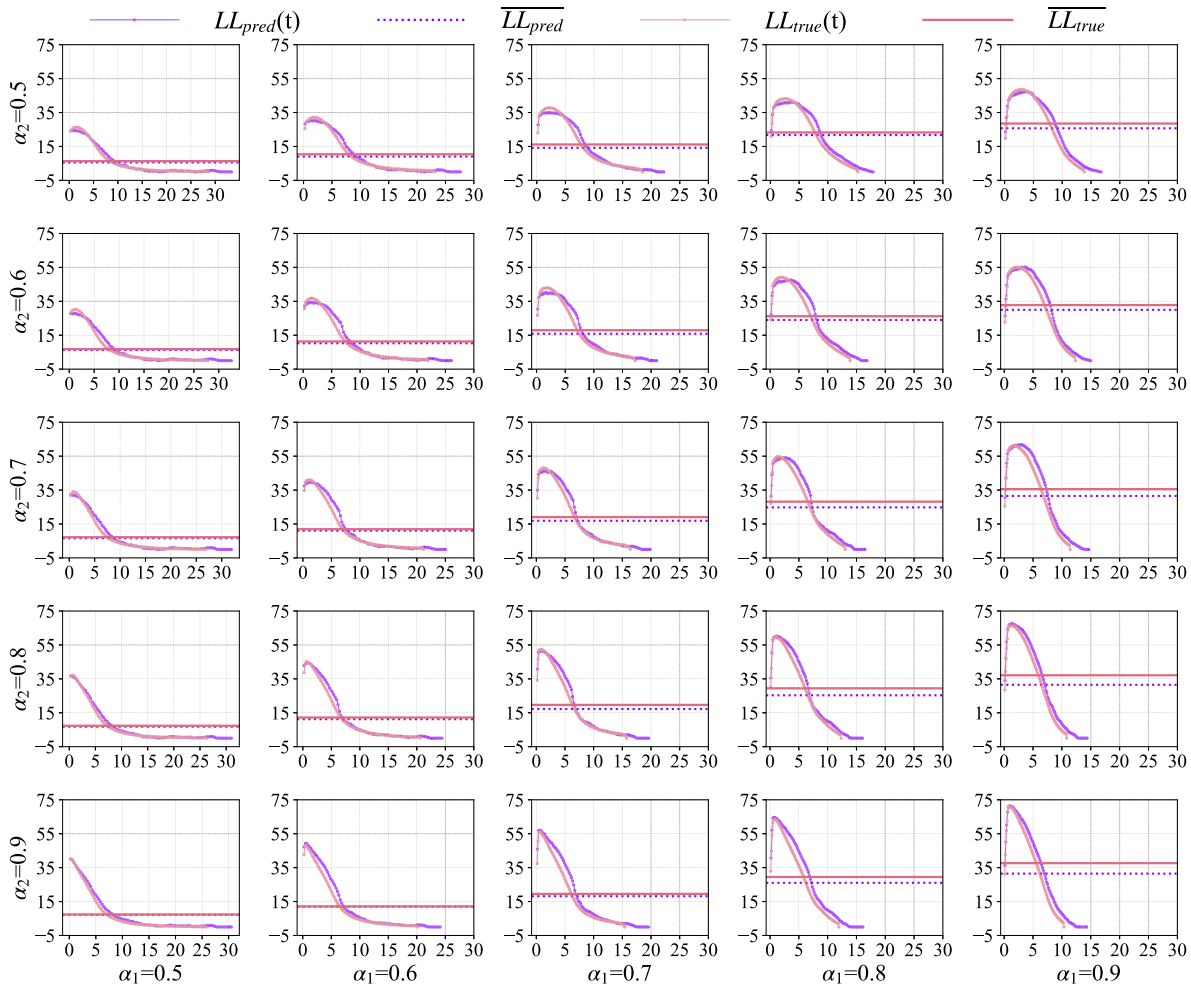

Fig. 15. Temporal evolution of the lead–lag metric during the resin filling process across the 25-case full-factorial design from both FlowCastNet (predicted) and PAM-RTM (true value).

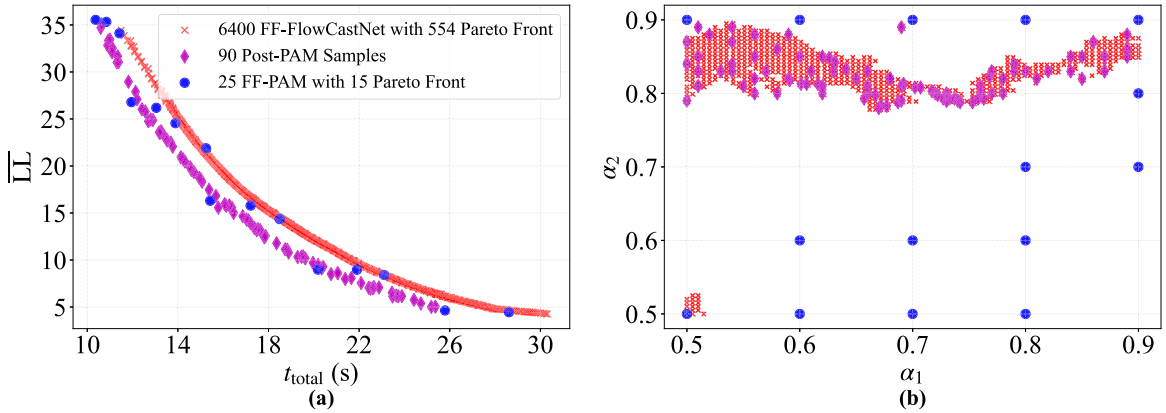

Fig. 16. Multi-criteria analysis of scenario 1: (a) Pareto fronts minimizing t_{total} and $\overline{LL}$, and (b) corresponding Pareto sets in the variables space.

and rapid evaluation of large VARTM design spaces at negligible computational cost. In this work, only two manufacturing scenarios are presented to illustrate the model capability, however, the number of possible cases that can be explored is almost unlimited within the design domain. Moreover, the surrogate model avoids repeated mesh regeneration typically required in FE-based workflows when geometric parameters change. Naturally, the surrogate model could also be combined with traditional numerical simulations to preform global design space exploration followed by local refinement (for example with FE-based optimization).

5. Conclusion

This study introduced FlowCastNet, a data-driven CNN-based model designed for rapid prediction of 3D filling patterns in VARTM processes with distribution medium. The model architecture incorporates a learnable feature embedding module, which encodes scalar process

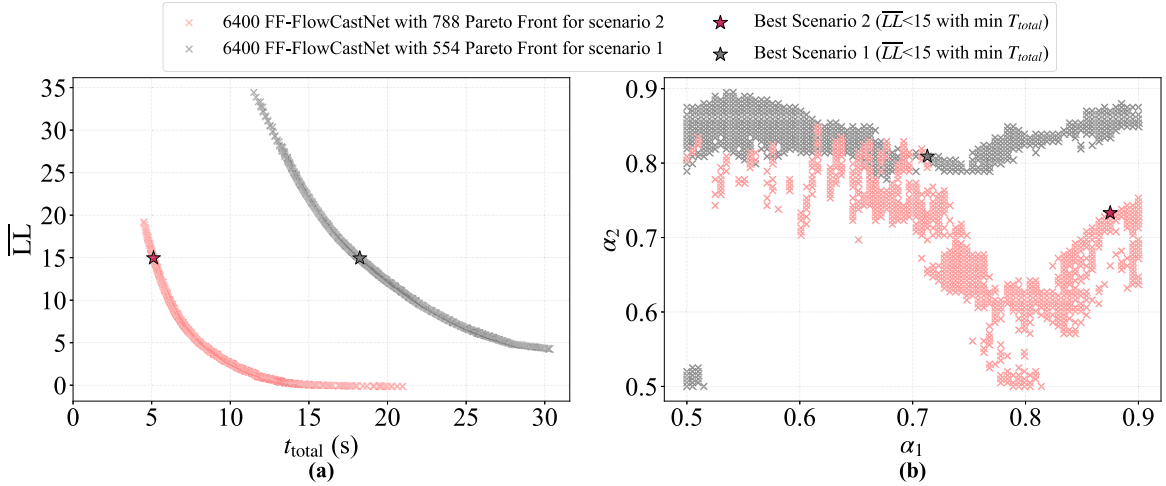

Fig. 17. Bi-objective optimization conducted with FlowCastNet for different manufacturing conditions (a) Pareto fronts, and (b) corresponding Pareto sets.

parameters that are typically difficult to represent simultaneously using conventional 2D image-based inputs, into the CNN backbone. As output, the model predicts a specified number of images representing filling time distributions on selected surfaces of the laminate.

The proposed approach was applied to the manufacturing of flat panels, considering several varying process parameters such as preform properties, DM configuration and race-tracking. A synthetic dataset was first generated with physics-based simulations, and the model was trained with a hybrid loss function combining pixel-wise information and overall structural similarity. After training, the predictive accuracy was evaluated on an independent test dataset comprising 200 cases within the design space. The model achieves satisfactory performance with an overall R^2 value of 0.992 and MSE of the order of 10^{-3}. Filling patterns on both top and bottom surfaces of the laminate are well-predicted, allowing further analysis of flow efficiency and lead–lag effects.

Compared to numerical simulations, the CNN surrogate model delivers extremely fast predictions, with an inference time of approximately 1 ms. This remarkable property enables the use of a brute-force approach to tackle multi-objective optimization problems. To illustrate this potential advantage, two typical manufacturing scenarios were analyzed to identify optimal DM configurations based on bi-objective optimization of impregnation speed and flow uniformity. The computational speed of FlowCastNet allowed exhaustive search of the design space and instantaneous generation of continuous Pareto Fronts. While some small quantitative errors remain compared to physics-based simulation, the presented results suggest that FlowCastNet provides good estimates of optimal solutions.

Overall, the study shows that CNN-based models have promising potential as quick manufacturing modeling tools that could eventually complement traditional numerical simulations. It should be reminded that the current framework relies on grid-based representations, which limits its direct applicability to non-rectangular geometries and highlights the need for further methodological development. Future work could focus on extending the present approach to more complex geometries through structured representations such as voxelization or masking, as well as through alternative learning paradigms that operate on unstructured data [51–53]. Since the proposed model architecture supports flexible input scalability, different part geometries and resin properties could be incorporated as additional process parameters. The predicted fields could also be extended beyond the final filling time images to include spatiotemporal distributions of temperature and pressure. By integrating these predictions with in-mold sensors, the framework could support real-time monitoring and closed-loop process correction in future developments. Different strategies should also be investigated to improve predictive accuracy, such as optimizing the size of the training dataset or incorporating physical constraints into the model.

CRediT authorship contribution statement

Yuwei Feng: Writing – review & editing, Writing – original draft, Visualization, Methodology, Investigation, Conceptualization. **Bin Yang:** Writing – original draft, Conceptualization. **Yunshi Huang:** Methodology. **Jihui Wang:** Writing – review & editing, Supervision, Conceptualization. **Philippe Causse:** Writing – review & editing, Supervision, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was partially funded by the Natural Sciences and Engineering Research Council of Canada (discovery grant 03992). Authors also gratefully acknowledge the financial support from the Research Center for High Performance Polymer and Composite Systems (CREPEC). This research was enabled in part by support provided by Calcul Québec (www.calculquebec.ca) and the Digital Research Alliance of Canada (alliancecan.ca). The authors also thank Prof. Cédric Béguin of Polytechnique Montréal for his support with the numerical simulations.

Data availability

Data will be made available on request.

References

[1] Correia NC, Robitaille F, Long AC, Rudd CD, Šimáček P, Advani SG. Analysis of the vacuum infusion moulding process: I. Analytical formulation. Compos Part A: Appl Sci Manuf 2005;36(12):1645–56.
[2] Van Oosterom S, Allen T, Battley M, Bickerton S. An objective comparison of common vacuum assisted resin infusion processes. Compos Part A: Appl Sci Manuf 2019;125:105528.
[3] Seemann William H. Plastic transfer molding apparatus for the production of fiber reinforced plastic structures. 1991, US Patent 5, 052, 906.

[4] Adhikari Debabrata, Gururaja Suhasini, Hemchandra Santosh. Resin infusion in porous preform in the presence of HPM during VARTM: Flow simulation using level set and experimental validation. Compos Part A: Appl Sci Manuf 2021;151:106641.

[5] Adhikari Debabrata, Gururaja Suhasini, Hemchandra Santosh, Kočí Jan. Resin infusion in porous preform in the presence of HPM considering complex geometry and gravity: Flow simulation and experimental validation. J Reinf Plast Compos 2023;42(19–20):971–89.

[6] Sun Xiudong, Li Shoujie, Lee L James. Mold filling analysis in vacuum-assisted resin transfer molding. Part I: SCRIMP based on a high-permeable medium. Polym Compos 1998;19(6):807–17.

[7] Heider Dirk, Simacek Pavel, Dominauskas Aurimas, Deffor Hope, Advani Suresh, Gillespie Jr John W. Infusion design methodology for thick-section, low-permeability preforms using inter-laminar flow media. Compos Part A: Appl Sci Manuf 2007;38(2):525–34.

[8] Bian LP, Yang JS, Xiao JY. Effect of distribution medium on resin flow behavior in vacuum infusion molding process. In: Proceedings of the 18th ICCM international conferences on composite materials, jeju island, Korea. 2011, p. 21–6.

[9] Dutra Carla Machado Bulsing, Amico Sandro Campos, Souza Jeferson Avila. Evaluation of flow-mesh influence in resin injection processes. Appl Compos Mater 2021;28:369–80.

[10] Hsiao Kuang-Ting, Devillard Mathieu, Advani Suresh G. Simulation based flow distribution network optimization for vacuum assisted resin transfer moulding process. Modelling Simul Mater Sci Eng 2004;12(3):S175.

[11] Sas Hatice S, Šimáček Pavel, Advani Suresh G. A methodology to reduce variability during vacuum infusion with optimized design of distribution media. Compos Part A: Appl Sci Manuf 2015;78:223–33.

[12] Seyednourani Mahsa, Yildiz Mehmet, Sas Hatice S. A two-stage optimization methodology for gate and vent locations and distribution media layout for liquid composite molding process. Compos Part A: Appl Sci Manuf 2021;149:106522.

[13] Loudad Raounak, Saouab Abdelghani, Beauchene Pierre, Agogue Romain, Desjoyeaux Bertrand. Numerical modeling of vacuum-assisted resin transfer molding using multilayer approach. J Compos Mater 2017;51(24):3441–52.

[14] Chebil Naziha, Deleglise-Lagardere Mylene, Park Chung Hae. Efficient numerical simulation method for three dimensional resin flow in laminated preform during liquid composite molding processes. Compos Part A: Appl Sci Manuf 2019;125:105519.

[15] Szarski Martin, Chauhan Sunita. Instant flow distribution network optimization in liquid composite molding using deep reinforcement learning. J Intell Manuf 2023;34(1):197–218.

[16] Bishara Dana, Xie Yuxi, Liu Wing Kam, Li Shaofan. A state-of-the-art review on machine learning-based multiscale modeling, simulation, homogenization and design of materials. Arch Comput Methods Eng 2023;30(1):191–222.

[17] Cassola Stefano, Duhovic Miro, Schmidt Tim, May David. Machine learning for polymer composites process simulation–a review. Compos Part B: Eng 2022;110208.

[18] Raschka Sebastian, Liu Yuxi Hayden, Mirjalili Vahid. Machine Learning with PyTorch and Scikit-Learn: Develop machine learning and deep learning models with Python. Packt Publishing Ltd; 2022.

[19] Chai Boon Xian, Eisenbart Boris, Nikzad Mostafa, Fox Bronwyn, Blythe Ashley, Bwar Kyaw Hlaing, Wang Jinze, Du Yuntong, Shevtsov Sergey. Application of KNN and ANN metamodeling for RTM filling process prediction. Materials 2023;16(18):6115.

[20] Zhu Junhong, Droste David, Dimassi Adli, Herrmann Axel S. A neural-network-assisted method for flow-front estimation in resin transfer molding using pressure sensors. J Compos Mater 2022;56(19):2979–93.

[21] Chen Yang, Simacek Pavel, Advani Suresh. Physics-informed neural networks for resin flow prediction in fibre textiles. In: 21st European conference on composite materials. The European Society for Composite Materials (ESCM) and the Ecole Centrale …; 2024, p. 292–9.

[22] Wang Y, Xu S, Bwar KH, Eisenbart B, Lu G, Belaadi A, Fox B, Chai BX. Application of machine learning for composite moulding process modelling. Compos Commun 2024;101960.

[23] Matsuzaki Ryosuke, Morikawa Masato, Oikawa Yuya, Ushiyama Kengo. Predicting thickness impregnation in a VaRTM resin flow simulation using machine learning. Compos Part C: Open Access 2021;5:100158.

[24] Zhang Aston, Lipton Zachary C, Li Mu, Smola Alexander J. Dive into deep learning. 2021, arXiv preprint arXiv:2106.11342.

[25] Caglar Baris, Broggi Guillaume, Ali Muhammad A, Orgéas Laurent, Michaud Véronique. Deep learning accelerated prediction of the permeability of fibrous microstructures. Compos Part A: Appl Sci Manuf 2022;158:106973.

[26] Korolev Denis, Schmidt Tim, Natarajan Dinesh K, Cassola Stefano, May David, Duhovic Miro, Hintermüller Michael. Hybrid machine learning based scale bridging framework for permeability prediction of fibrous structures. 2025, arXiv preprint arXiv:2502.05044.

[27] Jean Jimmy Gaspard, Broggi Guillaume, Caglar Baris. An image-based deep learning framework for flow field prediction in arbitrary-sized fibrous microstructures. Compos Part A: Appl Sci Manuf 2025;109337.

[28] Stieber Simon, Schröter Niklas, Schiendorfer Alexander, Hoffmann Alwin, Reif Wolfgang. FlowFrontNet: improving carbon composite manufacturing with CNNs. In: Joint European conference on machine learning and knowledge discovery in databases. Springer; 2020, p. 411–26.

[29] González Carlos, Fernández-León Joaquín. A machine learning model to detect flow disturbances during manufacturing of composites by liquid moulding. J Compos Sci 2020;4(2):71.

[30] Fernandez-Leon Joaquin, Keramati Keayvan, Garoz David, Baumela Luis, Miguel Carlos, González Carlos. A machine learning strategy for race-tracking detection during manufacturing of composites by liquid moulding. Integr Mater Manuf Innov 2022;11(2):296–311.

[31] Fernández-León Joaquín, Keramati Keayvan, Miguel Carlos, González Carlos, Baumela Luis. A deep encoder-decoder for surrogate modelling of liquid moulding of composites. Eng Appl Artif Intell 2023;120:105945.

[32] Zhang Runyu, Liu Yingjian, Zheng Thomas, Eddin Sarah, Nolet Steven, Liang Yi-Ling, Rezazadeh Shaghayegh, Wilson Joseph, Lu Hongbing, Qian Dong. A fast spatio-temporal temperature predictor for vacuum assisted resin infusion molding process based on deep machine learning modeling. J Intell Manuf 2024;35(4):1737–64.

[33] Mendikute Julen, Plazaola J, Baskaran Maider, Zugasti Ekhi, Aretxabaleta Laurentzi, Aurrekoetxea Jon. Impregnation quality diagnosis in resin transfer moulding by machine learning. Compos Part B: Eng 2021;221:108973.

[34] Chai Boon Xian, Wang Jinze, Dang Thanh Kim Mai, Nikzad Mostafa, Eisenbart Boris, Fox Bronwyn. Comprehensive composite mould filling pattern dataset for process modelling and prediction. J Compos Sci 2024;8(4):153.

[35] ESI Group, et al. User´ s guide & tutorials. 2014.

[36] Bickerton Simon, Advani Suresh G. Characterization and modeling of race-tracking in liquidcomposite molding processes. Compos Sci Technol 1999;59(15):2215–29.

[37] Arbter René, Béraud Jean-Marc, Binetruy Christophe, Bizet Laurent, Bréard Joël, Comas-Cardona Sébastien, Demaria Cristian, Endruweit Andreas, Ermanni Paolo, Gommer Frank, et al. Experimental determination of the permeability of textiles: A benchmark exercise. Compos Part A: Appl Sci Manuf 2011;42(9):1157–68.

[38] Yong Ana XH, Aktas Alper, May David, Endruweit Andreas, Advani S, Hubert Pascal, Abaimov Sergey G, Abliz Dilmurat, Akhatov Iskander, Ali Muhammad A, et al. Out-of-plane permeability measurement for reinforcement textiles: A benchmark exercise. Compos Part A: Appl Sci Manuf 2021;148:106480.

[39] Hsiao Kuang-Ting, Advani Suresh G. Flow sensing and control strategies to address race-tracking disturbances in resin transfer molding. Part I: design and algorithm development. Compos Part A: Appl Sci Manuf 2004;35(10):1149–59.

[40] Lawrence Jeffrey M, Barr John, Karmakar Rajat, Advani Suresh G. Characterization of preform permeability in the presence of race tracking. Compos Part A: Appl Sci Manuf 2004;35(12):1393–405.

[41] McKay Michael D, Beckman Richard J, Conover William J. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics 2000;42(1):55–61.

[42] Baudin M, Christopoulou M, Colette Y, Martinez JM. Pydoe: The experimental design package for python. 2017, URL https://Github.Com/Tisimst/PyDOE/.

[43] Virtanen Pauli, Gommers Ralf, Oliphant Travis E, Haberland Matt, Reddy Tyler, Cournapeau David, Burovski Evgeni, Peterson Pearu, Weckesser Warren, Bright Jonathan, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. Nature Methods 2020;17(3):261–72.

[44] Szandała Tomasz. Review and comparison of commonly used activation functions for deep neural networks. Bio-Inspired Neurocomputing 2021;203–24.

[45] Woo Sanghyun, Park Jongchan, Lee Joon-Young, Kweon In So. Cbam: Convolutional block attention module. In: Proceedings of the European conference on computer vision. ECCV, 2018, p. 3–19.

[46] Perez Ethan, Strub Florian, De Vries Harm, Dumoulin Vincent, Courville Aaron. Film: Visual reasoning with a general conditioning layer. In: Proceedings of the AAAI conference on artificial intelligence, vol. 32, (1). 2018.

[47] Zeiler Matthew D, Fergus Rob. Visualizing and understanding convolutional networks. In: European conference on computer vision. Springer; 2014, p. 818–33.

[48] Wang Zhou, Bovik Alan C, Sheikh Hamid R, Simoncelli Eero P. Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 2004;13(4):600–12.

[49] Collobert Ronan, Bengio Samy, Mariéthoz Johnny. Torch: a modular machine learning software library. Technical report, Idiap; 2002.

[50] Odena Augustus, Dumoulin Vincent, Olah Chris. Deconvolution and checkerboard artifacts. Distill 2016;1(10):e3.

[51] Maturana Daniel, Scherer Sebastian. Voxnet: A 3d convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ international conference on intelligent robots and systems. IROS, Ieee; 2015, p. 922–8.

[52] Gomez-Donoso Francisco, Escalona Felix, Cazorla Miguel. Par3dnet: Using 3dcnns for object recognition on tridimensional partial views. Appl Sci 2020;10(10):3409.

[53] Wang Cheng, Cheng Ming, Sohel Ferdous, Bennamoun Mohammed, Li Jonathan. NormalNet: A voxel-based CNN for 3D object classification and retrieval. Neurocomputing 2019;323:139–47.