Arch. Metall. Mater. **71** (2026), 3, 933-948

DOI: https://doi.org/10.24425/amm.2026.1985

SZYMON SITARZ[1*], MATEUSZ SITKO[1], LUKASZ MADEJ[1]

NEURAL CELLULAR AUTOMATA FOR CURVATURE-DRIVEN GRAIN GROWTH WITH GPU-ACCELERATED TRAINING DATA GENERATION

Neural Cellular Automata (NCA) offers a promising alternative to manually defined transition rules in classic Cellular Automata (CA) models of microstructure evolution. This concept is applied in the paper to develop a robust model of microstructure evolution driven by grain boundary curvature effects. However, the effectiveness of the NCA approach depends critically on the availability of large, high-quality training datasets. For physically motivated approaches such as the curvature-driven grain boundary migration model, generating such datasets with the available CPU-based CA code at the required size is computationally excessive. Therefore, in this work, a GPU-parallelized CUDA pipeline is developed as a high-throughput data-generation engine for training the NCA model of curvature-driven grain growth. The GPU-based CA grain growth model operates on the Mason formulation, and yields speedups of 20-22× over sequential CPU execution. As a result, it allows the reduction of dataset generation for NCA from a computational bottleneck to a practical pre-processing step. For the developed NCA curvature-driven grain boundary migration model, two approaches are used and tested within the research: a convolutional mask-based NCA, and a ring-based input pre-processing solution that exploits the underlying physics' isotropic symmetry. The NCA predictions are validated against the ground-truth results from a sequential CA grain growth model. This work demonstrates the capabilities of the two tested approaches and clearly identifies GPU-accelerated data generation as a critical step for scalable NCA-based microstructure modelling. It also provides a foundation for extending the approach to full-field static recrystallization simulations in the presence of coupled physical fields.

Keywords: Cellular automata; curvature-driven grain growth; neural cellular automata

1. Introduction

The Cellular Automata (CA) method is widely used to develop full-field models for microstructure evolution in various metallic materials, including grain growth [1], precipitation evolution [2,3], recrystallization [4], solidification [5] or phase transformation [6]. While simple CA models are computationally efficient, even in 3D, physically motivated CA approaches such as curvature-driven grain boundary migration require costly CA neighborhood searches, making large-scale simulations computationally demanding [7], which still prevents them from large-scale industrial applications. CA models parallelization strategies for CPUs and GPUs have traditionally focused on accelerating simulations to enable the use of large-scale, high-resolution CA computational domains, especially in 3D, for precise microstructure predictions. However, recent advances in data-driven modeling, particularly Neural Cellular Automata (NCA), have introduced a paradigm shift in which CA transition rules are generated from large datasets rather than explicitly defined by materials scientists [8].

The theoretical foundation for data-driven CA was established by Gilpin [9], who formally demonstrated that any cellular automaton can be represented as a convolutional neural network. Building on this equivalence, Mordvintsev et al. [8] introduced Growing Neural Cellular Automata, a differentiable self-organizing model in which a compact neural network replaced the manually defined transition rule and was trained end-to-end via the backpropagation algorithm. The NCA paradigm preserves the fundamental CA structure, including local interactions, defined neighborhood, iterative CA space, and CA state updates, while making the transition rules learnable. Recently, the concept was applied in the area of materials science by Tang et al. [10], who adapted the NCA concept to solidification microstructure modeling. This work demonstrated potential simulation speed-ups of up to 6 orders of magnitude while generalizing beyond the training domain. Then Seibert et al. [11] proposed NCA for microstructure reconstruction from statistical descriptors. Beyond NCA, other machine learning architectures have also been applied to grain growth prediction, including physics-regularized neural networks [12,13], convolutional recurrent

[1] AGH UNIVERSITY OF KRAKOW, AL. MICKIEWICZA 30, 30-059 KRAKOW, POLAND

* Corresponding author: sitarzs@agh.edu.pl

models [14,15], and surrogate frameworks based on spatial correlations [16].

However, a critical aspect common to all data-driven approaches is the requirement for large training datasets that capture the full diversity of CA neighborhood configurations and their corresponding state changes. For complex CA models, this requirement becomes the dominant computational cost, shifting the challenge from running simulations to generating data efficiently. To eliminate such problems, various optimization techniques, such as shared memory utilization, look-up tables, and data packing, can be used [17]. Other approaches based on CPU parallelization using OpenMP and MPI (Message Passing Interface) allow for even greater reductions in computational time across different materials science problems, such as static recrystallization simulations [18]. Finally, GPU implementations based on CUDA have been explored for solidification and dendritic growth [19,20], leveraging the inherently local nature of CA transition rules, which allows individual CA cell updates to be mapped directly to GPU threads.

Therefore, the present work addresses the mentioned challenge and proposes the NCA approach for curvature-grain growth modelling, enhanced with a neural network trained on GPU-accelerated CA model results. A curvature-driven grain boundary migration model following the Mason formulation [21] represents the physically motivated model and is used as a case study. First, a convolutional mask-based NCA is investigated. Then, a ring-based input pre-processing is proposed that exploits the isotropic symmetry of the curvature model, compressing the local CA neighborhood into a compact vector representation and enabling a linear neural network with 6 parameters in 2D and 10 in 3D to accurately reproduce the CA transition rule. The resulting NCA model was validated against the underlying CA transition rules from a classical CA model, and the conditions under which GPU acceleration provides significant benefits for data generation were analyzed. The presented approach establishes a foundation for extending NCA-based modeling to more sophisticated models like static recrystallization or phase transformations in the presence of continuous physical fields such as stored energy, temperature distributions etc.

2. Methodology

The methodology consists of three main components described in the following chapters. First, two developed CA grain-growth approaches are presented: a simplified unconstrained grain growth model used to provide input digital microstructure for further simulations and a physics-motivated curvature-driven grain boundary evolution model. Next, the GPU-parallelized data generation pipeline is presented as a critical stage for the development of the NCA model. Finally, the proposed NCA model architecture and training procedures are outlined along with the discussion on its capabilities, limitations, and possible further improvements.

2.1. Generation of input digital microstructure for grain boundary curvature-driven growth modelling

The initial digital microstructure model used as input for further modelling was generated with the cellular automata unconstrained grain growth algorithm [22]. The CA computational domain was discretized into a regular square lattice of CA cells, each described by a grain identifier (ID) initially equal to zero. A set of grain nuclei with a unique non-zero ID was then placed at randomly selected positions within the empty CA domain.

The digital microstructure evolves through an iterative procedure based on a majority-vote transition rule and the Moore neighborhood, comprising 8 neighbors of the investigated CA cell in 2D and 26 in 3D. At each time step, an unoccupied CA cell adopts the grain ID that appears most frequently among its neighbors (Fig. 1). In the case of a tie, the ID is selected deterministically by iterating through the neighboring cells in row-major order and selecting the first cell ID that participated in the tie (Fig. 2). The simulation continues until all CA cells had been assigned a non-zero ID, at which point the CA domain is filled with a set of grains and the digital microstructure is formed.

Periodic boundary conditions were applied in all cases during the simulation. An example of unconstrained grain growth algorithm simulation results is shown in Fig. 3.

Such resulting digital microstructure models serve as the starting points for the curvature-driven grain boundary migration model described in the following section.

While the Moore neighbourhood is presented here for clarity of illustration, the curvature-driven simulations employed in this study to generate NCA training data utilize a random pentagonal neighbourhood, which provides more equiaxed grain morphologies and reduces grid-induced anisotropy artefacts.

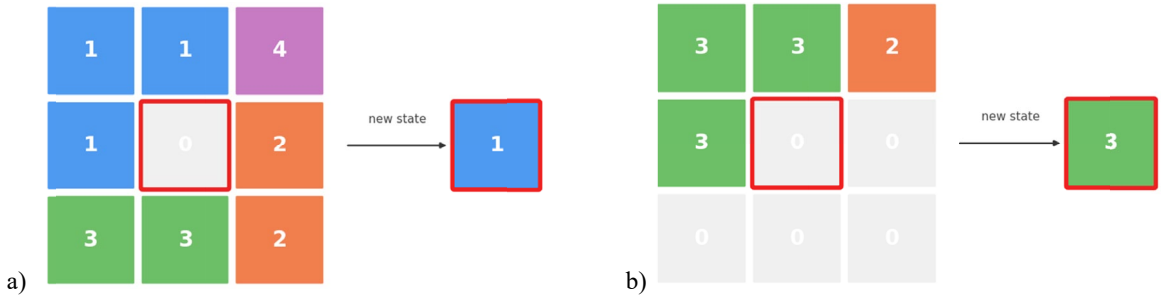

Fig. 1. Majority-vote rule: (a) clear majority: ID = 1 wins with four votes; (b) unoccupied cells (ID = 0) are excluded from voting

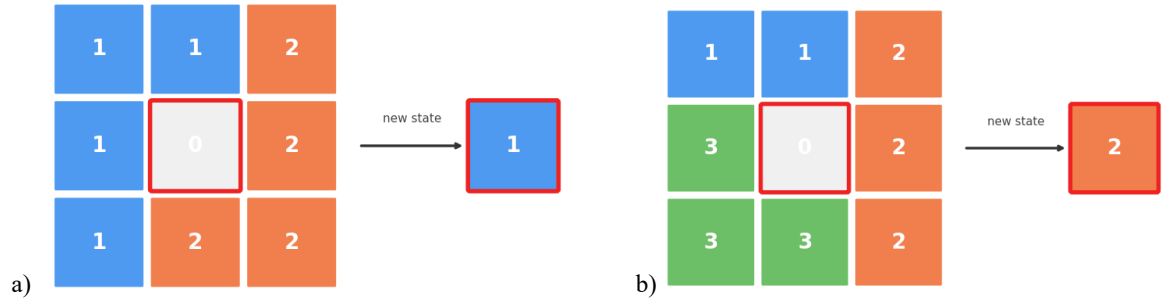

Fig. 2. Tie-breaking by row-major traversal: (a) tie between ID = 1 and ID = 2: ID = 1 selected first; (b) three-way tie: ID = 2 selected as first encountered

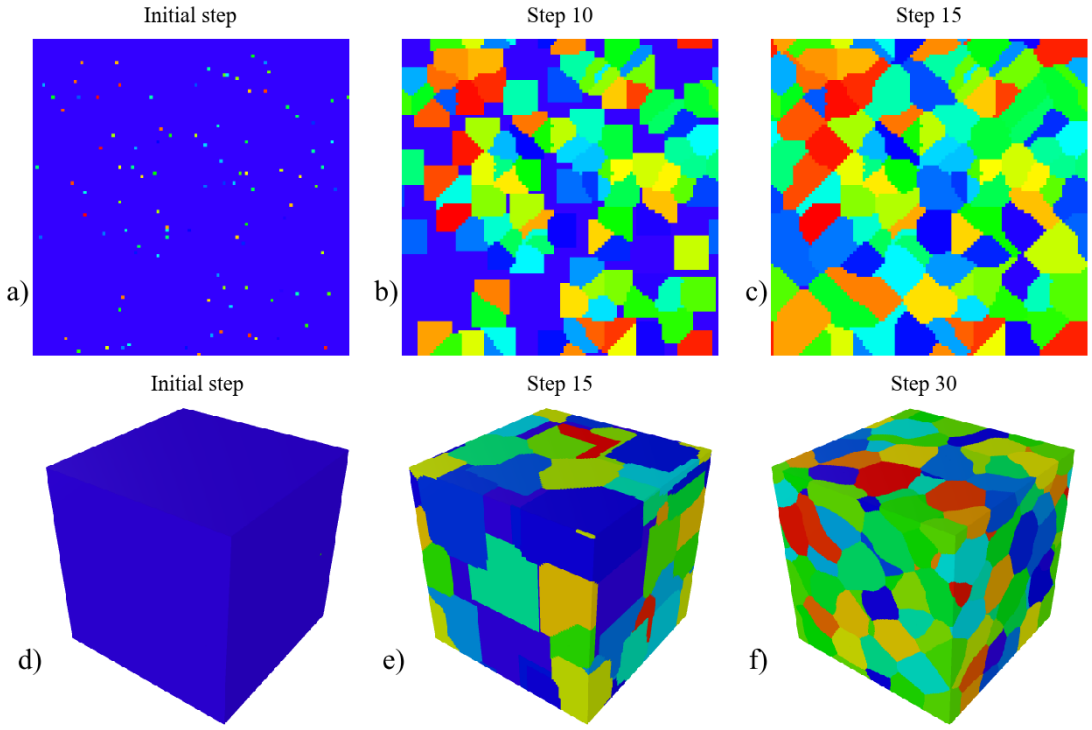

Fig. 3. Example of unconstrained grain growth simulation with the majority-vote CA model using the Moore neighborhood: (a-c) 2D evolution at the initial step, step 10, and step 15, respectively; (d-f) 3D evolution at the initial step, step 15, and step 30, respectively

2.2. Cellular Automata grain boundary curvature-driven growth model

The current investigation focuses exclusively on grain boundary curvature-driven microstructure evolution at elevated temperatures. The driving force for microstructural evolution is therefore limited only to curvature-related effects, allowing for a controlled assessment of the governing transition rules. The role of crystallographic orientation, grain boundary misorientation and inclination angles, or variations in grain boundary energy, is not taken into account. This simplified setting provides a suitable basis for evaluating the feasibility of replacing conventional, physically motivated CA transition rules with data-driven approaches, such as NCA, while preserving the essential characteristics of grain boundary migration.

Therefore, in this research, all CA cells after initial grain growth are treated as fully annealed/recrystallized cells without any deformation stored energy, and the only driving force behind further evolution is the net pressure related to grain boundary curvature [23].

As a result, the velocity of the i^{th} CA cell located at the grain boundary is a function of the mobility and the net pressure:

$$v_i = M_G P_{GB}, \tag{1}$$

where: M_G – grain boundary mobility and P_{GB} – net pressure on the grain boundary:

$$P_{GB} = -m_c \kappa \gamma \tag{2}$$

where: m_c – scaling coefficient, γ – grain boundary energy and κ – grain boundary curvature.

In the present study, the expression $M_G \cdot \gamma \cdot m_c$ was set to unity, providing a normalized baseline that allows assessment of the NCA learning capability independently of any specific material system.

As mentioned, the grain boundary curvature is calculated using a model proposed by Mason [24], which depends on the

number of neighboring CA cells, the fraction of cells belonging to the same grain, the cell size, and lattice-geometrical factors. This model, in contrast to the Kreymayer one [25], incorporates weight interactions between the investigated CA cells. In this case, the interaction strength decreases with increasing distance from the central CA cell. The interaction between neighboring cells in 2D is described by Eq. (3), where the weighting function depends on the spatial coordinates and a standard deviation parameter. In 3D simulations, the third part of the equation is extended to x_k and y_k, respectively:

$$F_{ij} = \exp\left[-\frac{\left(x_j - x_i\right)^2 + \left(y_j - y_i\right)^2}{2\sigma^2}\right] \tag{3}$$

where: σ – standard deviation expressed in m, x and y – CA cell coordinates in the global system of the cellular automaton space.

The cumulative influence of all neighboring CA cells is then obtained from Eq. (4), which accounts for grain membership through the Kronecker delta function:

$$D = \sum_{j:F_{ij} \in S_i} F_{ij}\left(2\delta_{q_i q_j} - 1\right) \tag{4}$$

where: $\delta_{q_i q_j}$ – Kronecker delta, q_i, q_j – variables that determine the affiliation of CA cells i and j, respectively, with a given grain, S_i – set of all CA cells involved in the curvature calculation for ith CA cell.

The calculation of grain curvature for a given CA cell in 2D space is expressed as:

$$\kappa = \exp\left(\frac{\beta^2 a^2}{2\sigma^2}\right)\left[\frac{\alpha_P D}{\sqrt{2\pi}\sigma^3} - \frac{\sqrt{2\pi}}{\sigma} erf\left(\frac{\beta a}{\sqrt{2}\sigma}\right)\right] \tag{5}$$

where: β – coefficient equal to 0.5 [23], α_p – CA cell area, a – physical CA cell size.

The parameter σ must be calibrated, as it depends on both the CA cell geometry and the definition of the neighborhood [26]. According to the theoretical background of that model, the grain boundary exists between adjacent cells, and the calculated curvature has units of 1/m. A detailed study of the role of various grain curvature models in the CA grain growth simulations can be found in earlier authors' work [27].

Then, in the presented approach, Eq. (1) is considered for each CA cell, which is located at the grain boundary front in the t^{th} CA time step, to evaluate the volume fraction of the migrating grain:

$$f_{i,t} = f_{i,t-1} + \left(\frac{v_i t_{step}}{a}\right) \tag{6}$$

where: $f_{i,t-1}$ – volume fraction of the migrating grain in i^{th} CA cell in the t–1 time step, v_i – velocity of the moving front in the cell from Eq. (1).

Finally, the local evolution of CA states in growing grains is evaluated for each CA cell located at the moving front at a given CA time step, accounting for the migration volume fraction from the previous step, the local interface velocity, and the physical CA cell size:

$$Y_{i,j}^{t+1} = \begin{cases} CG \Leftrightarrow f_{i,t} > 1.0 \\ Y_{i,j}^{t} \end{cases} \tag{7}$$

where: CG – cell change state in curvature growth, $Y_{i,j}^{t}$ – state of the neighboring cell (i,j) in a particular time step t.

The simulation is carried out on a regular lattice of square CA cells, with interactions defined by the Moore neighborhood.

2.3. Neural Cellular Automata grain boundary curvature-driven growth model

The concept of the NCA approach is based on the assumption that the manually defined CA transition rules developed by materials experts are replaced by a neural network trained to reproduce local CA state updates from a large amount of data (Fig. 4).

The network developed in this research receives a representation of the local neighborhood patch (a fixed 5×5 in 2D or

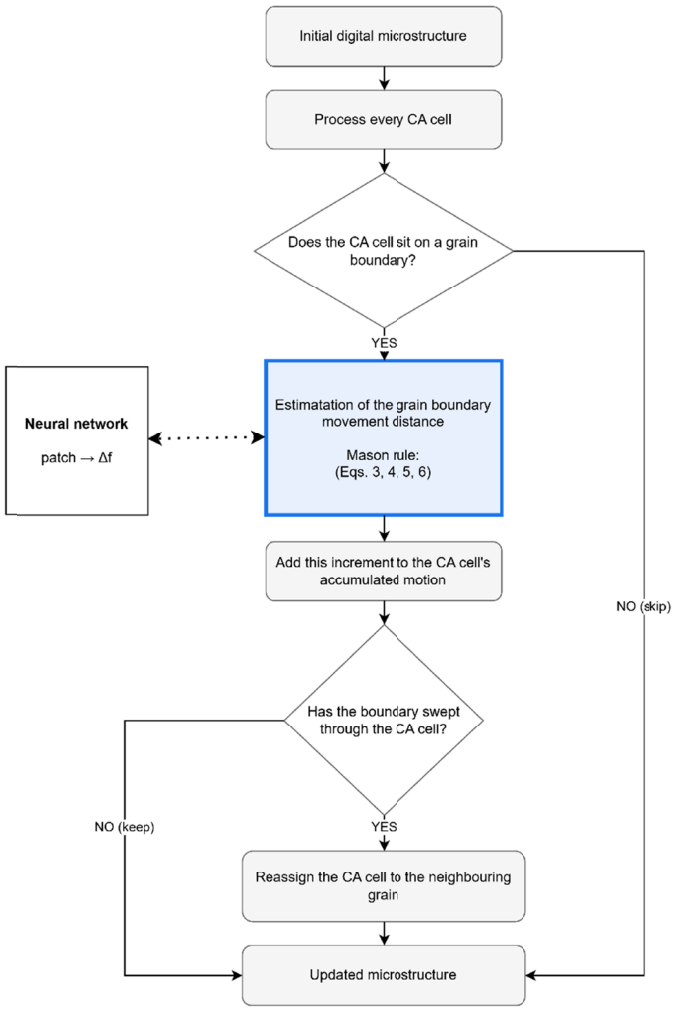

Fig. 4. Step-by-step update procedure for the curvature-driven NCA model, with the neural network predicting the grain fraction increment Δf from the local neighborhood patch

5×5×5 in 3D window of CA cells centered on the investigated cell, as illustrated in Fig. 5) and predicts the updated state of the central CA cell. Training data is generated from the CA grain boundary curvature-driven growth model from Chapter 2.2, with individual patches extracted as input–output pairs.

Two neural network input data representations were investigated within the paper, differing in the degree of pre-processing applied to the local CA neighborhood patch. The choice of input representation is the central design decision in any NCA model, since the input defines the function class, which the network can represent and determines which structural properties of the underlying physical model – such as rotational symmetry or translational invariance – must be inferred from data and which can be encoded directly into the architecture. A representation that exposes physically relevant invariants reduces the effective complexity of the learning problem, lowers the required model capacity, and improves both training stability and generalization, while a generic representation places the entire burden on the network and on the size of the training dataset. The two representations evaluated in this work were therefore selected to span this spectrum: a generic convolutional pipeline that operates directly on the binary same-grain mask, and a symmetry-aware ring-based pipeline that explicitly encodes the isotropic structure of the Mason curvature model.

2.3.1. Full patch data representation in the NCA model for curvature-driven grain growth

At first, an approach shown in Fig. 5 with 5×5 (2D) or 5×5×5 (3D) neighborhood patches of grain IDs extracted around the central CA cell was investigated. In this case, a binary occupancy mask of the same size is constructed by comparing each CA cell to the central one: a value of 1 is assigned if the CA cell belongs to the same grain as the center, and 0 otherwise. This binary mask of 25 values in 2D or 125 values in 3D is passed directly to a convolutional neural network (CNN): Conv2d(1→32, 5×5) followed by a fully connected head (32→32→1) in 2D, and the analogous Conv3d architecture in 3D.

After the initial results discussed in Chapter 3, the approach was modified to include an additional pre-processing stage based on a proposed ring concept, with CA cells at specific distances from the central, investigated CA cell.

2.3.2. Ring-based patch data representation in the NCA model for curvature-driven grain growth

As mentioned, the second developed approach extends the pipeline shown in Fig. 4, by adding a ring-based compression stage. In this case, each position in the patch is assigned to a concentric ring based on its squared Euclidean distance from the center, $r^2 = \Delta x^2 + \Delta y^2$ in 2D. Positions at equal squared distances are grouped into the same ring regardless of direction. For a 5×5 patch, this yields five unique non-zero squared distances – $r^2 \in \{1,2,4,5,8\}$ – corresponding to concentric five rings. In 3D, the analogous grouping of a 5×5×5 patch yields nine concentric rings. The ring assignment table is pre-computed once at the beginning of the simulations and remains fixed for all patches. The binary mask is then compressed into a ring summation vector S of length 5 in 2D and 9 in 3D, where each entry S_r aggregates the mask values within ring r as:

$$S_r = \sum_{k \in ring\ r} 2m_k - 1 \tag{8}$$

where: $m_k \in \{0,1\}$ – binary mask value at position k, mapping same-grain CA cells to +1 and different-grain CA cells to –1. The resulting vector S serves as the sole input to the developed neural network.

The workflow modification according to the described concept is shown in Fig. 6.

It is worth mentioning that this compression is physically motivated, as the Mason curvature model is isotropic, meaning that the driving force depends on the radial distribution of grain boundary positions relative to the central cell, rather than on their absolute direction. Grouping positions by squared distance directly encodes this rotational symmetry, eliminating directional redundancy that would otherwise require the neural network

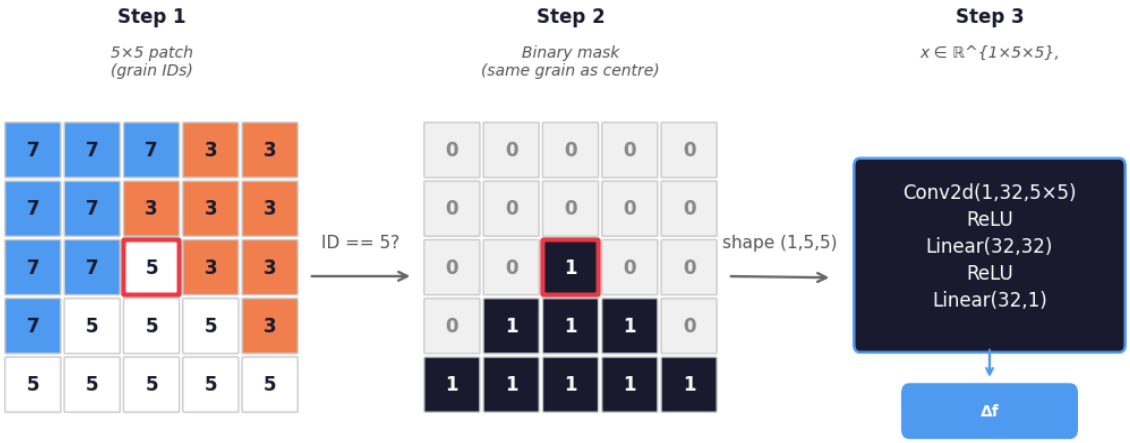

Fig. 5. Input pre-processing pipeline for the convolutional NCA model: (Step 1) 5×5 grain ID patch, (Step 2) binary same-grain mask, (Step 3) convolutional network predicting Δf

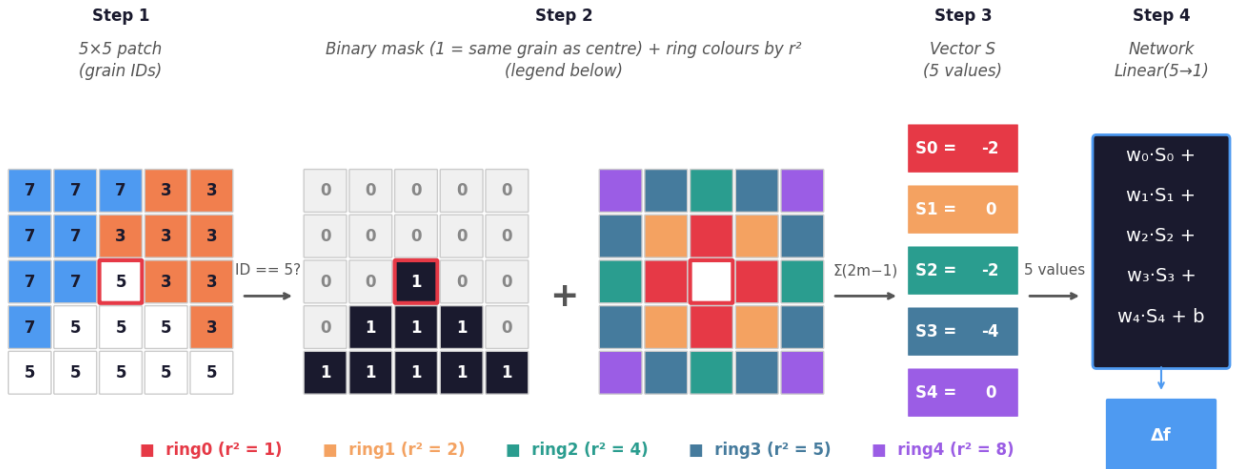

Fig. 6. Ring-based pre-processing pipeline for the curvature-driven NCA: (Step 1) 5×5 grain ID patch, (Step 2) binary same-grain mask, (Step 3) ring summation vector S of length 5 in 2D (9 in 3D), (Step 4) linear network predicting Δf

to learn isotropy from data. The first approach from Chapter 2.3.1, which retains the full spatial layout, does not exploit this symmetry and consequently requires a larger neural network to achieve comparable accuracy.

For the ring summation pre-processing approach, three neural network architectures were evaluated: a single Linear layer (Linear, $n \rightarrow 1$), a wider Multi Layer Perceptron (MLP) ($n \rightarrow 32 \rightarrow 16 \rightarrow 1$), and a compact MLP ($n \rightarrow 8 \rightarrow 1$), where $n = 5$ in 2D and $n = 9$ in 3D. All models were trained with Root Mean Square Error (RMSE) loss, and the Adam optimizer at a learning rate of 10^{-3} with *ReduceLROnPlateau* scheduling. Input features and targets were standardized using z-score normalization fitted exclusively on the training set to prevent data leakage.

The predicted output from the neural network is the increase in the volume fraction Δf of the growing grain in a particular CA cell, which enters the simulation through Eq. (6) to determine grain boundary migration.

However, as was pointed out, to effectively train such neural networks an extensive data set from classical CA simulations is required. In this case, the long computing times, especially in 3D may be a significant constraint. That is why data preparation for training was carried out in two stages. In the first stage, to reduce the CA simulation time, it was decided to parallelize the curvature-driven grain growth CA model with the use of a GPU and CUDA framework capabilities. In the second stage, this GPU-parallelized CA model was used to generate large datasets and divide them into batches for network training.

3. Development of the training data sets

3.1. GPU-accelerated data generation

The parallelization strategy originates directly from the locality of the Mason transition rule: the update of each CA cell depends only on its 5×5 (2D) or 5×5×5 (3D) neighborhood and is independent of the updates performed in the same time step on other CA cells. Each CA cell is therefore mapped to a single GPU thread, and threads are grouped into blocks of 16×16 in 2D and 8×8×8 in 3D. A double-buffered layout is used for both grain identifiers and recrystallized volume fractions, with the previous-step state read from one buffer and the updated state written to the second one, eliminating the need for synchronization between CA cell updates within a single time step. The Gaussian neighborhood weights from Eq. (3) are pre-computed once on the host unit and stored in GPU constant memory, which provides cached broadcast access for all threads and avoids redundant evaluation of the exponential term. Periodic boundary conditions are handled directly through modular index arithmetic, without explicit halo regions. Finally, an early-exit test based on the Moore neighborhood identifies CA cells located in the grain interior and neglects the full curvature calculation for them, which is a significant optimization solution given that interior CA cells dominate typical microstructures.

The execution times of the GPU-based CUDA workflow were compared with the classic sequential CPU implementation of the Mason curvature CA model. Benchmarks were performed on an AMD Ryzen 7 9800X3D CPU and an NVIDIA RTX 5070 Ti Strix GPU, using 3D domains of 100^3 and 256^3 cells, each initialized with approximately 100 grains, over 5000 simulation steps, and the results are summarized in TABLE 1.

TABLE 1

Comparison of GPU and CPU execution times for 5000 steps of the Mason curvature model in 3D

CA domain size	GPU time (s)	CPU time (s)	Speedup
100×100×100	0.81	16.2	~20×
256×256×256	9.19	205	~22×

For 100×100×100 CA cells, the GPU approach completed 5000 steps in 0.81 s compared to approximately 16.2 s on the CPU, yielding a speedup of approximately 20×. For a 256×256×256 domain, the GPU implementation required 9.19 s, compared with approximately 205 s for the CPU code, resulting in a speedup

of approximately 22×. The modest increase in speedup with domain size is consistent with improved GPU occupancy at larger problem sizes: as the number of CA cells grows, the fixed kernel launch overhead becomes negligible relative to the computational workload, and a greater fraction of GPU cores is utilized. The Mason curvature update has a high computational cost per CA cell, since each update involves a weighted summation over the extended 5×5 (2D) or 5×5×5 (3D) mask together with the evaluation of exponential and error-function terms. This makes the workload per GPU thread substantial enough to amortize kernel launch overhead and benefit fully from GPU parallelism.

For the larger CA domain sizes, CPU-based generation of the full 5000-step dataset would take over 3 minutes per simulation. At the scale required for NCA training, encompassing diverse initial microstructures and sufficient coverage of the input space, this cost becomes prohibitive without hardware acceleration. Thus, the GPU pipeline reduces this to under ten seconds per run, making large-scale dataset generation for NCA training practically feasible.

3.2. Training data sets definition

For the convolutional NCA model based on the full 5×5 (2D) and 5×5×5 (3D) binary mask representation, the training data were collected directly from full GPU-accelerated CA simulations of the Mason curvature model. Ensembles of independent runs were performed across multiple domain configurations: in 2D, seven configurations spanning 256^2 and 512^2 domains with 50-1000 initial grains, and in 3D, five configurations covering 48^3 to 80^3 domains with 60-500 grains were used. Each configuration was repeated across two or three independent runs initialized with different grain nuclei positions, and patches were collected at fixed CA-step intervals throughout each simulation. In total, 2.5×10^6 patches were collected for both the 2D and the 3D variants for the NCA training. This sampling strategy ensures that the dataset captures the natural distribution of grain morphologies and grain-boundary configurations that arise during curvature-driven evolution, ranging from sharp early-stage interfaces to smoother equilibrated boundaries.

Two filtering mechanisms were applied during the GPU collection stage to remove redundant or non-informative patches. First, CA cells located in homogeneous Moore (1) neighborhoods, i.e., in grain interiors, were entirely omitted, as they provide no boundary information. Second, near-boundary CA cells with low computed curvature ($\kappa \leq 10^{-6}$) were retained only with a probability of 12%, while cells with non-negligible curvature were always retained. This retention probability was tuned empirically during dataset preparation to suppress the dominance of trivial near-zero-curvature samples in the raw collection while still preserving a representative population of low-curvature cases. This stochastic sub-sampling was implemented directly in the collection kernel using a hash-based pseudo-random generator seeded with the CA cell index, ensuring reproducibility without inter-thread synchronization. The resulting datasets are concentrated around configurations that contribute meaningful gradient information during training while still preserving the natural distribution of low-curvature cases.

For the ring-based NCA model, a fundamentally different data generation strategy was proposed. Since the ring-summation vector S is the sole input to the model, the dataset was generated synthetically, by directly enumerating or sampling configurations of S and computing the corresponding Δf analytically from the Mason curvature formula, without running any CA simulation. In 2D, the input space is bounded: each ring r contains a fixed number of cells, so each S vector takes integer values in a finite range, and the total number of unique configurations is only 5625. The complete 2D dataset was therefore obtained by exhaustive enumeration of all 5625 valid combinations on the GPU, with one CUDA thread per configuration, yielding deterministic, exhaustive coverage of the input space. In 3D, the analogous space contains approximately 1.05×10^{10} configurations, and full enumeration is not feasible; a dataset of 10^7 samples was therefore generated by uniform random sampling of S vectors, with Δf again computed analytically from the Mason formula. This synthetic approach removes the dependence on simulation runtime entirely and ensures that the training distribution covers the input space that the model will encounter at inference, rather than the empirical distribution induced by any particular set of microstructure runs.

In both pipelines, the collected samples were partitioned into training, validation, and test subsets in a 70:15:15 ratio with a fixed random seed, and both inputs and targets were standardized by z-score normalization fitted exclusively on the training subset to prevent data leakage.

The developed training data sets can now be used to train and evaluate the proposed NCA model. The performance of the trained NCA model is evaluated at two levels. At the local, patch level, accuracy metrics are computed on a held-out test set to assess how precisely the network reproduces the CA transition rule for individual neighborhood configurations. At the more global, microstructural level, the trained models are deployed in a full simulation, and the resulting grain evolutions are compared with the reference CA simulation results in a step-by-step manner.

4. Result and discussion

4.1. Convolution neural network training

The convolutional neural network, which operates directly on the binary same-grain mask without ring compression, provides the baseline for evaluating the impact of physics-informed pre-processing. As shown in Fig. 7 and Fig. 8, the validation loss exhibits pronounced oscillations throughout training. Training RMSE followed an identical trajectory across all experiments, confirming that the oscillations reflect optimization dynamics rather than overfitting. This instability is a consequence of the absence of an encoded symmetry. The neural network must infer isotropy from data rather than receive it as a structural training

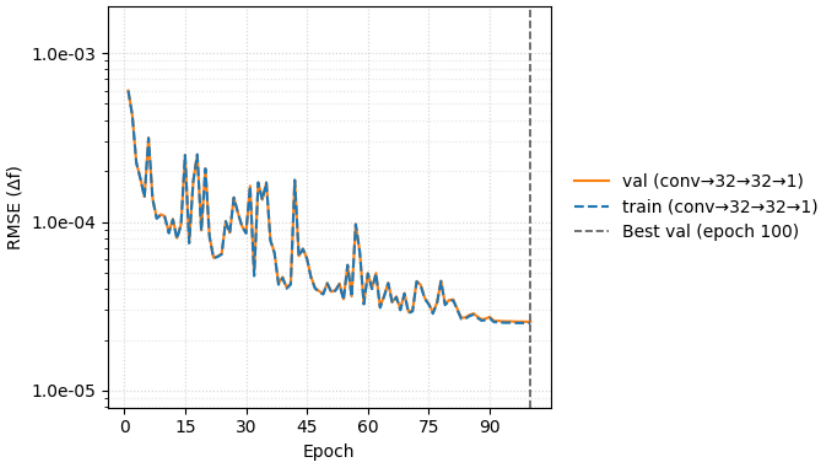

Fig. 7. Validation RMSE during training of the convolutional NCA model in 2D. Training RMSE follows an identical trajectory throughout the training operations

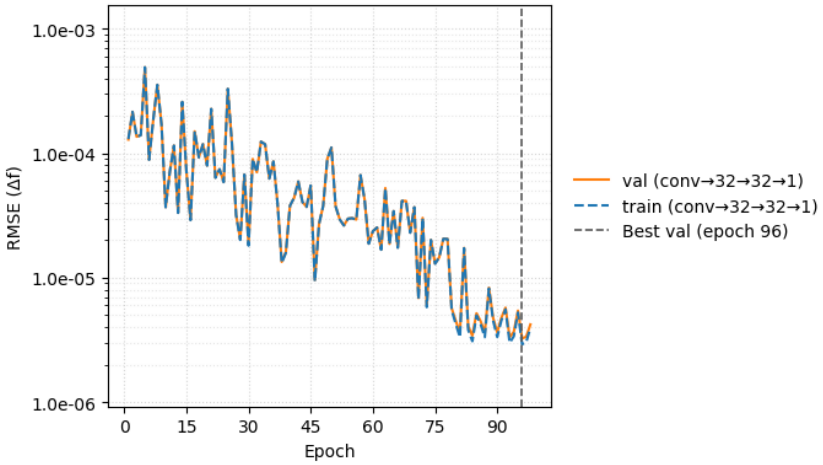

Fig. 8. Validation RMSE during training of the convolutional NCA model in 3D. Training RMSE follows an identical trajectory throughout the training operations

constraint, which renders the loss surface less regular. Despite this, the model converges to a functional approximation of the CA transition rule, confirming that the convolutional architecture is capable of learning the relevant response.

As mentioned, in Chapter 2.3, the ring-based pre-processing goes a step further than the above-described approach as it encodes the isotropic symmetry of the Mason curvature model directly into the input representation, reducing the learning problem to fitting a linear function of the ring summation vector *S*. The training results from Fig. 9 and Fig. 10 confirm the advantage of this approach.

As shown in Fig. 9 and Fig. 10, the linear model converges to significantly lower RMSE values than both MLP variants, despite having far fewer parameters, namely 6 in 2D and 10 in 3D, including the bias. This outcome is a direct consequence of the inductive bias introduced by the pre-processing: the MLP architecture, lacking this constraint, attempts to model nonlinearities absent in the data, leading to slower convergence and higher final error. As presented in all cases, the training RMSE was indistinguishable from the validation RMSE throughout training, indicating that all models generalize well within the patch-level task.

From the NCA training stage, it is evident that when the input representation does not reflect the symmetry of the underlying physical model, training is possible but unstable, and the network requires greater capacity to properly approximate the CA transition rule. At the same time, when rotational symmetry is encoded directly in the input data through the ring-based pre-processing, the learning problem reduces to a linear one. This is consistent with the structure of the Mason curvature formula itself, in which the driving force depends linearly on the weighted neighborhood sum *D* (Eq. 4); the ring summation vector preserves this linear dependence, and the training results confirm that no additional nonlinearity is needed to reproduce the CA transition rule. For this reason, the linear model with 6 parameters in 2D and 10 in 3D was selected as the deployed variant for the ring-based NCA, while the convolutional architecture is retained as the baseline representation without physics-informed pre-processing.

As both trained NCA models appear to successfully approximate the CA transition rule, they are then validated in Section 4.2 against ground-truth data from the standard CA grain growth model.

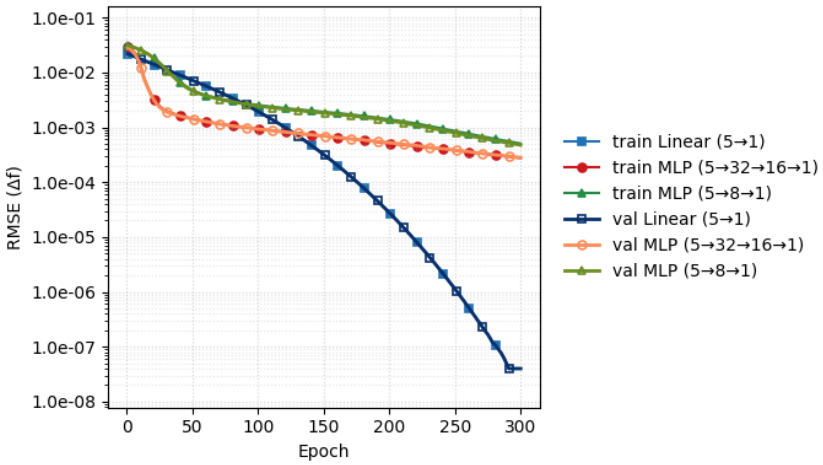

Fig. 9 Training and validation RMSE during training of ring-based NCA models in 2D

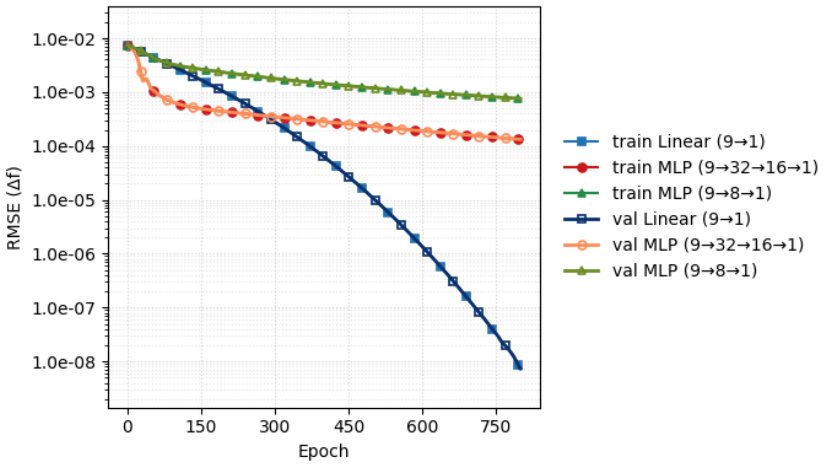

Fig. 10 Training and validation RMSE during training of ring-based NCA models in 3D

4.2. Numerical validation of the NCA model predictions

The developed and trained NCA models were deployed for full-field grain growth simulations, and the resulting outputs were evaluated against the reference CA model results at the microstructure level. Two CA domain sizes were tested for each model variant to assess whether accuracy scales with problem size, namely 100×100 (10000 CA cells) and 2000×2000 (4000000 CA cells). For 3D cases, the CA domain sizes of 100×100×100 (1000000 CA cells) and 256×256×256 (16777216 CA cells) were used. Each CA cell corresponds to a physical area of 1 μm×1 μm (1 μm^2). A qualitative comparison of results obtained from convolutional NCA models is shown in Figs. 11-14, while corresponding quantitative metrics are summarized in TABLE 2 and TABLE 3.

To facilitate identification of local differences in both model predictions, results discrepancy maps highlighting cell-level ID mismatches are provided for the 2D case in Fig. 11(g-i) and Fig. 12(g-i).

As presented in Figs. 11-14, convolutional NCA results remain qualitatively consistent with the reference CA results throughout the simulation. The grain boundary morphology and growth topology are well preserved at all timesteps. The difference maps reveal that disagreements between the NCA and CA models are concentrated along grain boundaries rather than distributed randomly across the domain, which is physically consistent with the local nature of the transition rule approximation error. Quantitatively, as shown in TABLE 2 and TABLE 3, ID agreement degrades gradually with time step: in 2D, 44 mismatched CA cells appear at time step 500 for the smaller domain, rising to 9183 CA cells at time step 5000 for the 2000×2000 CA domain with a fraction MAE of 4.2×10^{-3}. In 3D, the larger CA domain accumulates 41215 mismatched CA cells at time step 5000, corresponding to a fraction MAE of 2.2×10^{-2}. These errors remain small in absolute terms, with ID agreement staying above 99.75% in all investigated cases, and confirm that the convolutional NCA model is a reliable approximation of the CA transition rule suitable for practical simulations.

The second NCA approach was then tested under the same assumptions. In general, the ring-based linear NCA model also achieves perfect agreement with the reference CA in 2D and near-perfect agreement in 3D. The comparison of the outcomes

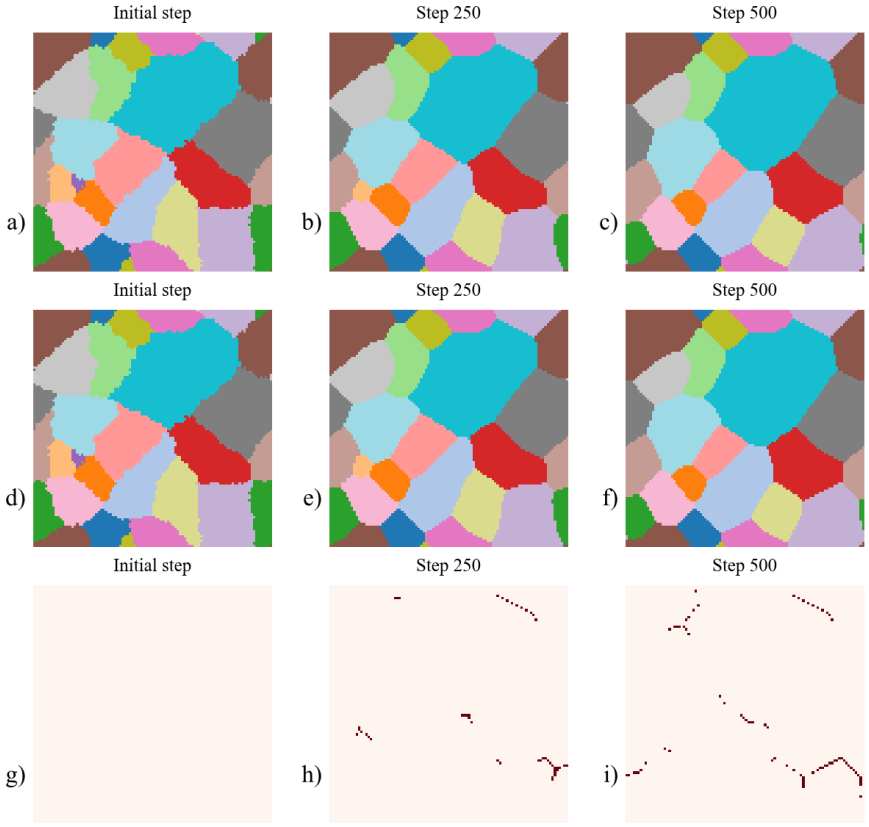

Fig. 11. Microstructure comparison for the convolutional NCA model in 2D: (a) reference CA at initial state, (b) reference at step 250, (c) reference at step 500, (d) NCA model at initial state, (e) model at step 250, (f) model at step 500, (g) discrepancy map at initial state, (h) discrepancy map at time step 250, (i) discrepancy map at step 500. Red pixels indicate cells with a mismatched grain ID. Domain size 100×100 cells

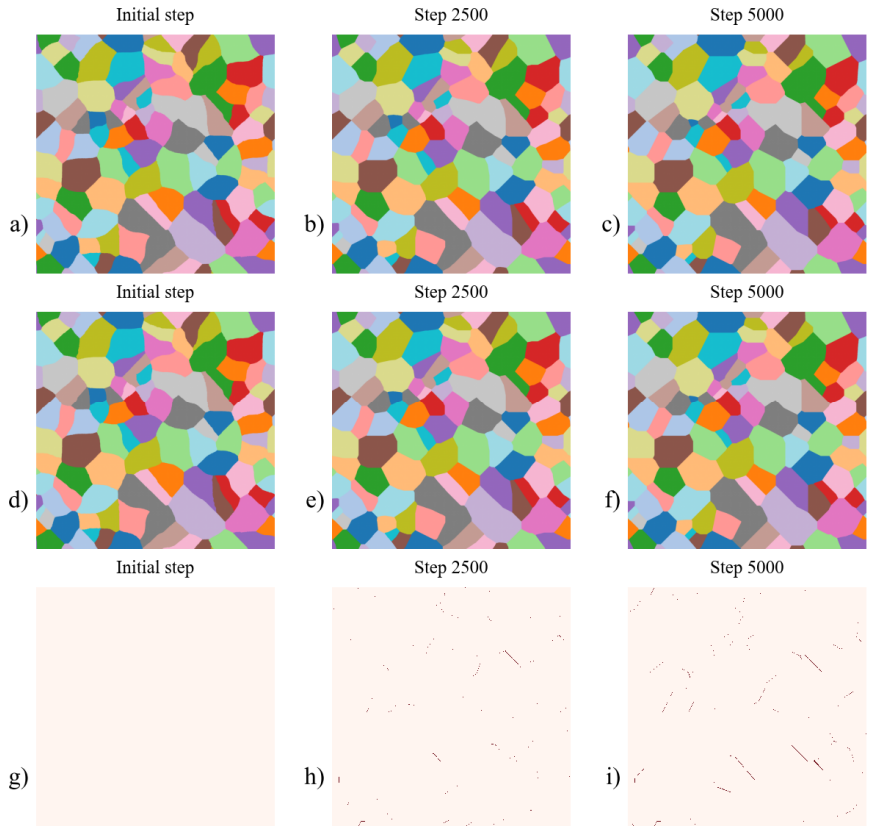

Fig. 12. Microstructure comparison for the convolutional NCA model in 2D: (a) reference CA at initial state, (b) reference at step 2500, (c) reference at step 5000, (d) NCA model at initial state, (e) model at step 2500, (f) model at step 5000, (g) discrepancy map at initial state, (h) discrepancy map at time step 2500, (i) discrepancy map at step 5000. Red pixels indicate cells with mismatched grain ID. Domain size 2000×2000 cells

TABLE 2

Quantitative metrics for the convolutional NCA model in 2D

Time step	Number of CA cells	ID agreement (%)	Number of mismatched CA cells	Fraction Mean Absolute Error (MAE)
0	10000	100	0	0
250	10000	99.74	26	0.010005
500	10000	99. 66	44	0.012085
0	4000000	100	0	0
2500	4000000	99.89	4354	0.002502
5000	4000000	99.77	9183	0.004174

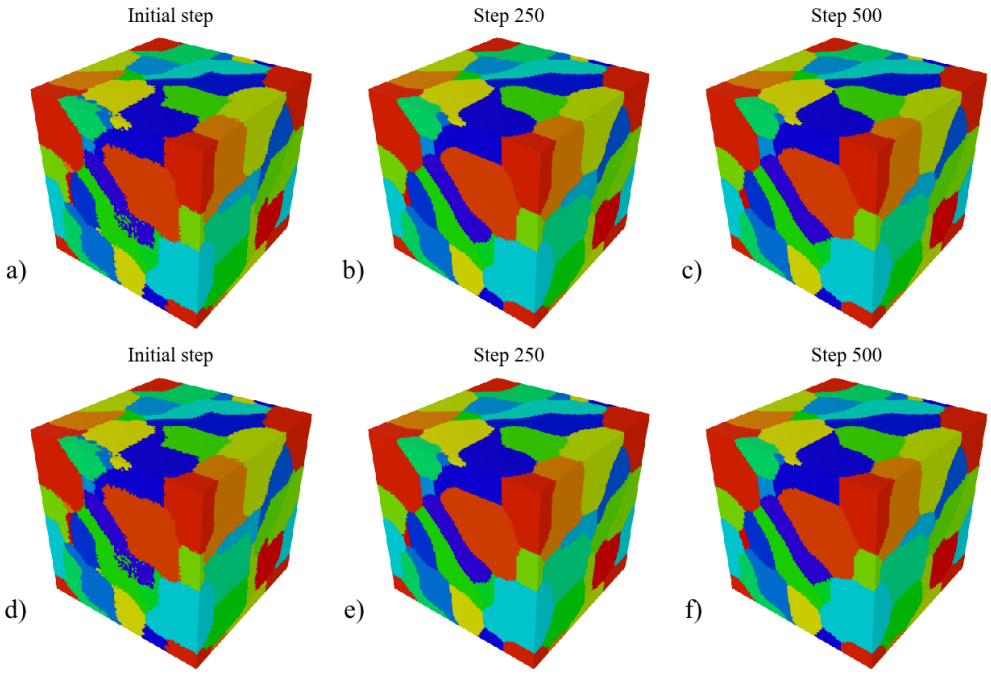

Fig. 13. Microstructure comparison for the convolutional NCA model in 3D: (a) reference CA at initial state, (b) reference at step 250, (c) reference at step 500, (d) NCA model at initial state, (e) model at step 250, (f) model at step 500. Domain size 100×100×100 cells

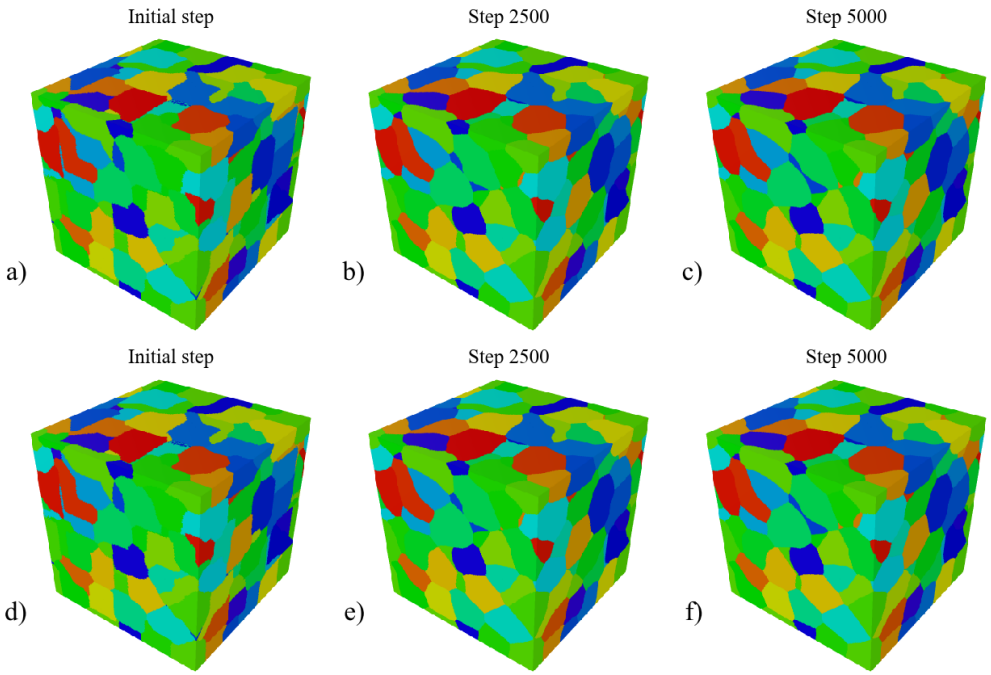

Fig. 14. Microstructure comparison for the convolutional NCA model in 3D: (a) reference CA at initial state, (b) reference at step 2500, (c) reference at step 5000, (d) NCA model at initial state, (e) model at step 2500, (f) model at step 5000. Domain size 256×256×256 cells

TABLE 3

Quantitative metrics for the convolutional NCA model in 3D

Time step	Number of CA cells	ID agreement (%)	Number of mismatched CA cells	Fraction MAE
0	1000000	100	0	0
250	1000000	99.999	11	0.000035
500	1000000	99.992	82	0.000357
0	16777216	100	0	0
2500	16777216	99.895	17547	0.008579
5000	16777216	99.754	41215	0.021875

from NCA and CA models is shown in Figs. 15-18 and TABLE 4 and TABLE 5. Mismatch maps are omitted for the ring-based 2D model as ID agreement remains at 100.0% level throughout the simulations.

As presented in Figs. 15-18 the grain boundary morphology and growth kinetics from NCA are visually indistinguishable from the reference CA results at all recorded time steps.

TABLE 4 and TABLE 5, indicate that in 2D ID agreement is perfect across both domain sizes with a fraction MAE below 10^{-6}. In 3D, in the smaller CA domain, perfect cell-level agreement with MAE of order 10^{-7} is visible. For the larger 3D CA domain 256×256×256 cells, a small number of mismatched CA cells at the level of 512 appear, which leads to an ID agreement of 99.98% and a fraction MAE of 1.4×10^{-3}. This represents a marginal and localized accumulation of error over an extended simulation time. This is consistent with the fact that the 3D input space of $\sim10^{10}$ configurations cannot be exhaustively covered during the NCA training stage, unlike the 2D case, where full coverage is possible.

The validation results presented above demonstrate that the NCA framework is capable of accurately reproducing the CA transition rule at the cell level, with ID agreement exceeding 99.75% in all investigated cases. It should be noted, however, that the present framework operates on discrete grain ID representations, which constitute a deliberate simplification relative to experimentally obtained microstructural data. In practice, microstructure characterization techniques such as EBSD provide crystallographic orientation data as Euler angles rather than discrete grain identifiers. Extending the present approach to such data would introduce additional complexity, including the handling of orientation-space metrics, crystallographic symmetry operations, measurement noise inherent to EBSD acquisitions, and grain segmentation from misorientation thresholds. While these challenges are non-trivial, the accuracy demonstrated here at the patch level suggests that such a data-driven approach remains a viable direction and will be further extended.

Finally, the computational cost of NCA models relative to the reference CA one was assessed. The ring-based linear model introduces an overhead of 2-5% compared to the original Mason CA, a negligible penalty given the significant reduction in model complexity. The convolutional model incurs approximately 20% higher overhead, attributed to the additional cost

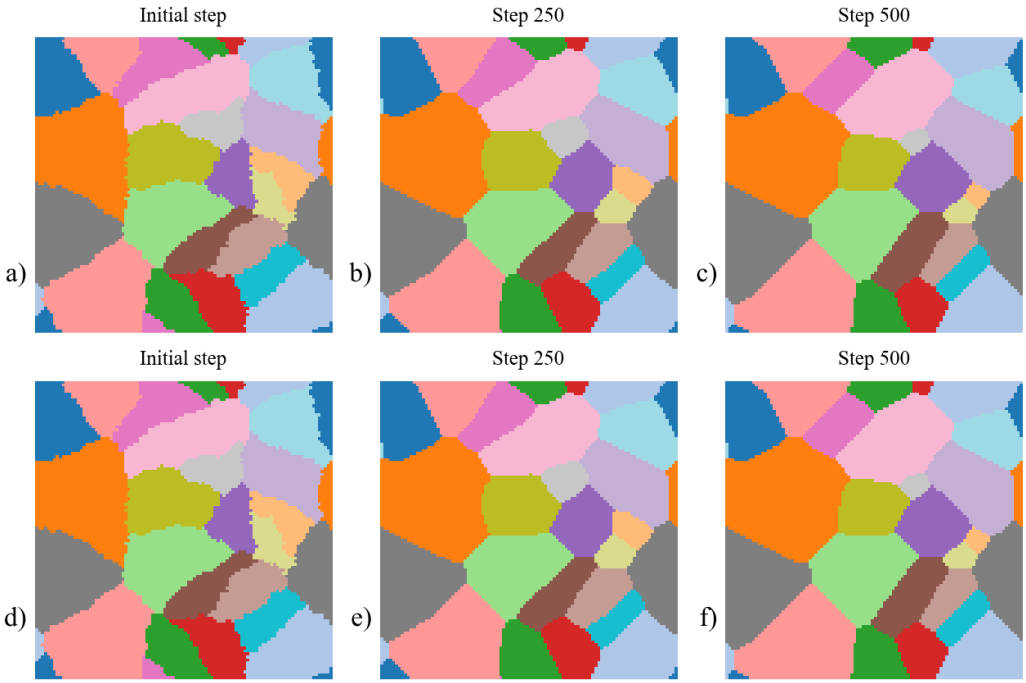

Fig. 15. Microstructure comparison for the linear NCA model in 2D: (a) reference CA at initial state, (b) reference at step 250, (c) reference at step 500, (d) NCA model at initial state, (e) model at step 250, (f) model at step 500. Domain size 100×100 cells

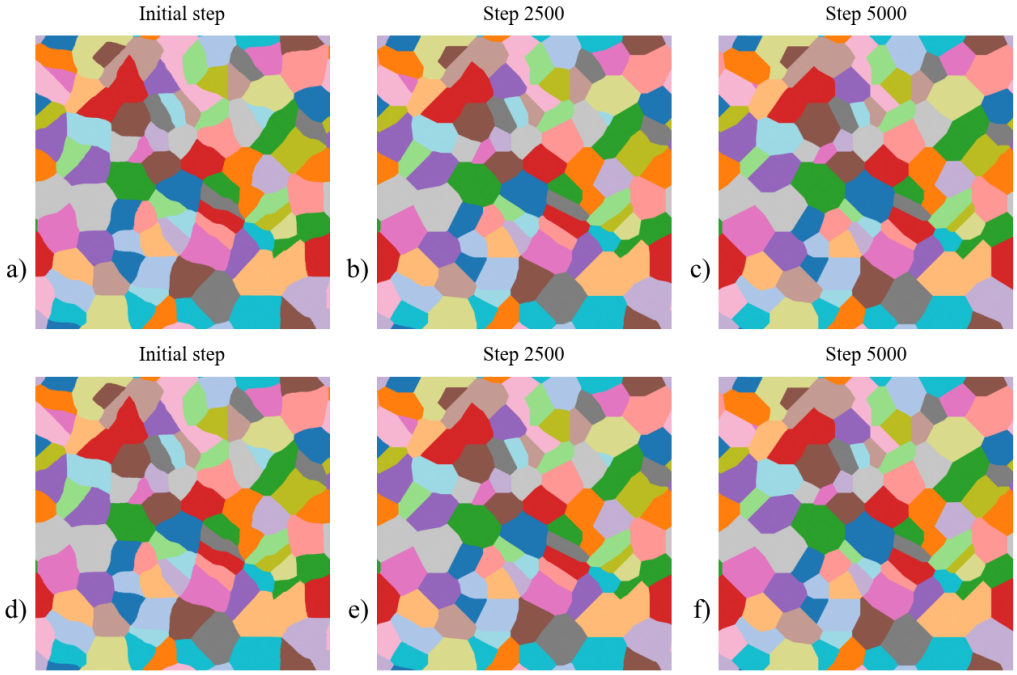

Fig. 16 Microstructure comparison for the linear NCA model in 2D: (a) reference CA at initial state, (b) reference at step 2500, (c) reference at step 5000, (d) NCA model at initial state, (e) model at step 2500, (f) model at step 5000. Domain size 2000×2000 cells

TABLE 4

Quantitative metrics for the ring-based NCA model in 2D

Time step	Number of CA cells	ID agreement (%)	Number of mismatched CA cells	Fraction MAE
0	10000	100	0	0
250	10000	100	0	1.33E-07
500	10000	100	0	1.26E-07
0	4000000	100	0	0
2500	4000000	100	0	2.25E-07
5000	4000000	100	0	8.62E-07

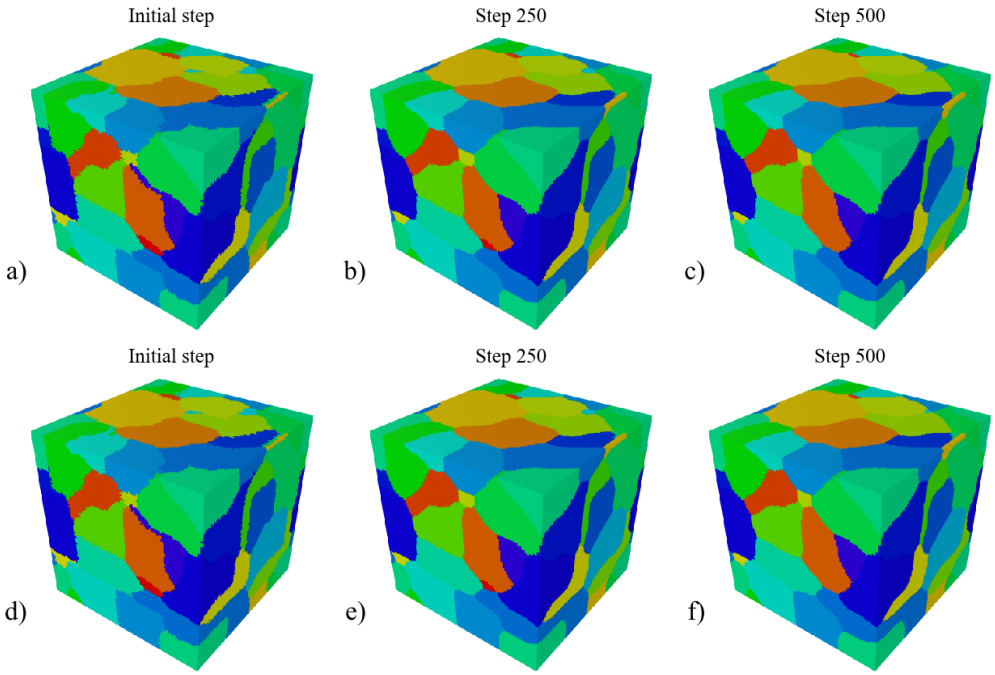

Fig. 17. Microstructure comparison for the linear NCA model in 3D: (a) reference CA at initial state, (b) reference at step 250, (c) reference at step 500, (d) NCA model at initial state, (e) model at step 250, (f) model at step 500. Domain size 100×100×100 cells

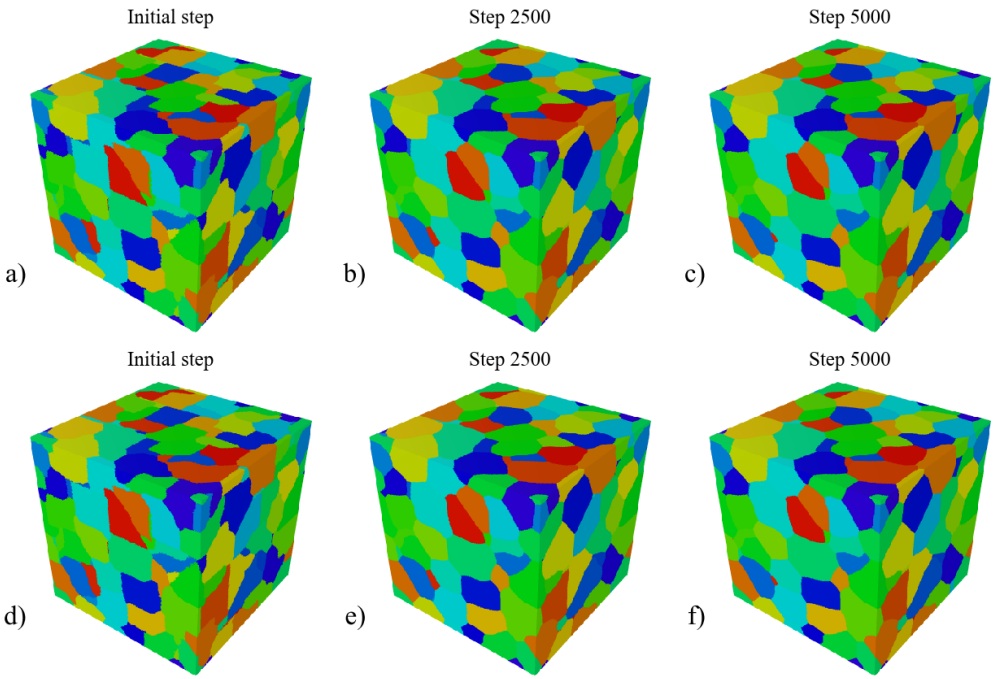

Fig. 18. Microstructure comparison for the linear NCA model in 3D: (a) reference CA at initial state, (b) reference at step 2500, (c) reference at step 5000, (d) NCA model at initial state, (e) model at step 2500, (f) model at step 5000. Domain size 256×256×256 cells

TABLE 5

Quantitative metrics for the ring-based NCA model in 3D

Time step	Number of CA cells	ID agreement (%)	Number of mismatched CA cells	Fraction MAE
0	1000000	100	0	0
250	1000000	100	0	5.42E-08
500	1000000	100	0	1.04E-07
0	16777216	100	0	0
2500	16777216	99.997	512	0.000218
5000	16777216	99.984	2684	0.001397

of patch extraction and convolution operations over the entire CA domain. While this represents a more substantial slowdown, the convolutional architecture retains an important advantage for future applications: unlike the ring-based model, which exploits a linearity specific to the Mason curvature formulation, the mask-based representation imposes no assumptions about the functional form of the transition rule. This makes it directly applicable to physically more complex models, such as static recrystallization, where the governing dynamics involve coupled fields and are inherently nonlinear.

5. Conclusions

This work presented a Neural Cellular Automata concept applied to the simulation of microstructure evolution under curvature-driven grain growth and demonstrated two developed complementary approaches to learning the Mason CA transition rule directly from data. Based on the analysis, the following conclusions can be drawn:

– encoding the isotropic symmetry of the Mason curvature formulation directly into the input representation reduces the learning problem to a linear one. A neural network with as few as 6 parameters in 2D and 10 in 3D is sufficient to reproduce the CA transition rule with near-perfect accuracy, achieving 100% cell-level ID agreement in 2D across all domain sizes tested and negligible error accumulation in 3D.

– the convolutional mask-based model, operating on the full binary neighborhood patch without physics-informed compression, requires a larger architecture and exhibits less stable training dynamics than the ring-based approach. Nevertheless, it can converge to a reliable approximation of the CA transition rule, maintaining ID agreement level above 99.75% across all tested configurations.

– unlike the ring-based model, the convolutional architecture imposes no assumptions about the functional form of the transition rule, making it directly applicable to physically more complex CA models where linearity cannot be assumed.

– generation of training data from GPU-parallelized CA model is a critical step to enable NCA-based microstructure training. Benchmarks on 3D domains demonstrated speedups of approximately 20-22× over sequential CPU

execution, with the gain increasing modestly with domain size, consistent with improved GPU occupancy. For the Mason curvature model, where each CA cell update requires a weighted summation over 25 (2D) or 125 (3D) neighbors, combined with the evaluation of exponential and error-function terms, GPU acceleration reduces data set generation from a computational bottleneck to a practical pre-processing step.

The current framework operates on discrete-grain ID representations in the CA domain. A natural further research direction is to use crystallographic orientation data, in the form of Euler angles obtained from EBSD measurements, as direct CA model input, enabling end-to-end learning of microstructure evolution from experimentally accessible quantities and improving the quality of the training data set. Such an approach requires the development of a neural network architecture operating on the full microstructure domain rather than on local patches. However, the accuracy demonstrated here for patch-level rule approximation suggests that data-driven microstructure evolution modeling of this kind is within reach. For problems such as static recrystallization, where the governing dynamics involve coupled physical fields and inherently nonlinear transition rules, the convolutional architecture provides a more natural foundation for this extension.

Further validation based on standard microstructural descriptors, including mean grain size evolution, number of grains, and a quantitative comparison with the expected parabolic grain-growth kinetics for a specific material system, is foreseen as part of future work.

Acknowledgement

Financial assistance of the National Science Centre project No 2024/55/B/ST8/00295 is acknowledged.

REFERENCES

[1] X. Liang, C. Bos, M. Hermans, I. Richardson, An Improved Cellular Automata Solidification Model Considering Kinetic Undercooling. Metallurgical and Materials Transactions B: Process Metallurgy and Materials Processing Science **54**, 1088-1098 (2023). DOI: https://doi.org/10.1007/s11663-023-02742-3

[2] J. Opara, B.B. Straumal, P. Zięba, Cellular Automata Modeling of Two Discontinuous Reactions in Fe-13.5 At. Pct Zn Alloy During Ageing and Annealing. Metall. Mater. Trans. A Phys. Metall. Mater. Sci. **55**, 1036-1048 (2024). DOI: https://doi.org/10.1007/s11661-024-07302-1

[3] J. Opara, B. Straumal, P. Zięba, Cellular automata modelling of discontinuous precipitation. Materials **14** (2021). DOI: https://doi.org/10.3390/ma14174985.

[4] Z.-J. Chen, Y.C. Lin, S. Zhang, J.-C. Zhu, N.-F. Zeng, G.-C. Wu, Q.-M. Yang, A continuum-field-enabled multiscale framework coupling crystal plasticity and cellular automaton for discontinuous dynamic recrystallization: Application to a nickel-based superalloy. J. Alloys Compd. 188264 (2026). DOI: https://doi.org/10.1016/j.jallcom.2026.188264

[5] M. Wróbel, A. Burbelko, D. Gurgul, Modelling of change in density of nodular cast iron during solidification using cellular automaton. Archives of Metallurgy and Materials **60**, 2709-2713 (2015). DOI: https://doi.org/10.1515/amm-2015-0436

[6] F. Xiong, Y. Lian, C. Panwisawas, J. Chen, M. jian Li, A. Liu, A microscale cellular automaton method for solid-state phase transformation of directed energy deposited Ti6Al4V. Addit. Manuf. **95** (2024). DOI: https://doi.org/10.1016/j.addma.2024.104517

[7] M. Sitko, K. Banaś, L. Madej, Scaling scientific cellular automata microstructure evolution model of static recrystallization toward practical industrial calculations. Materials **14** (2021). DOI: https://doi.org/10.3390/ma14154082

[8] A. Mordvintsev, E. Randazzo, E. Niklasson, M. Levin, Growing Neural Cellular Automata. Distill (2020). https://api.semanticscholar.org/CorpusID:213719058

[9] W. Gilpin, Cellular automata as convolutional neural networks. Phys. Rev. E **100**, 32402 (2019). DOI: https://doi.org/10.1103/physreve.100.032402

[10] J. Tang, S. Kumar, L. De Lorenzis, E. Hosseini, Neural cellular automata for solidification microstructure modelling. Comput. Methods Appl. Mech. Eng. **414** (2023). DOI: https://doi.org/10.1016/j.cma.2023.116197

[11] P. Seibert, A. Raßloff, Y. Zhang, K. Kalina, P. Reck, D. Peterseim, M. Kästner, Reconstructing Microstructures From Statistical Descriptors Using Neural Cellular Automata. Integr. Mater. Manuf. Innov. **13**, 272–287 (2024). DOI: https://doi.org/10.1007/s40192-023-00335-1

[12] W. Yan, J. Melville, V. Yadav, K. Everett, L. Yang, M.S. Kesler, A.R. Krause, M.R. Tonks, J.B. Harley, A novel physics-regularized interpretable machine learning model for grain growth. Mater. Des. **222** (2022). DOI: https://doi.org/10.1016/j.matdes.2022.111032

[13] J. Melville, V. Yadav, L. Yang, A.R. Krause, M.R. Tonks, J.B. Harley, Anisotropic physics-regularized interpretable machine learning of microstructure evolution. Comput. Mater. Sci. **238** (2024). DOI: https://doi.org/10.1016/j.commatsci.2024.112941

[14] K. Yang, Y. Cao, Y. Zhang, S. Fan, M. Tang, D. Aberg, B. Sadigh, F. Zhou, Self-supervised learning and prediction of microstructure evolution with convolutional recurrent neural networks. Patterns *2* (2021). DOI: https://doi.org/10.1016/j.patter.2021.100243

[15] P. Tep, M. Bernacki, High-fidelity grain growth modeling: Leveraging deep learning for fast computations. Acta Mater. **301** (2025). DOI: https://doi.org/10.1016/j.actamat.2025.121486

[16] S. Hashemi, S.R. Kalidindi, A machine learning framework for the temporal evolution of microstructure during static recrystallization of polycrystalline materials simulated by cellular automaton. Comput. Mater. Sci. **188**, 110132 (2021). DOI: https://doi.org/10.1016/j.commatsci.2020.110132

[17] M. Sitko, M. Czarnecki, K. Pawlikowski, L. Madej, Evaluation of the effectiveness of neighbors' selection algorithms in the random

cellular automata model of unconstrained grain growth, Materials and Manufacturing Processes 1-11, (2023).
DOI: https://doi.org/10.1080/10426914.2023.2196753

[18] M. Kühbach, L.A. Barrales-Mora, G. Gottstein, A massively parallel cellular automaton for the simulation of recrystallization. Model. Simul. Mat. Sci. Eng. **22**, 075016 (2014).
DOI: https://doi.org/10.1088/0965-0393/22/7/075016

[19] Y. Zhang, J. Zhou, Y. Yin, X. Shen, T.A. Shehabeldeen, X. Ji, Gpu-accelerated cellular automaton model for grain growth during directional solidification of nickel-based superalloy. Metals (Basel). **11**, 1-13 (2021).
DOI: https://doi.org/10.3390/met11020298

[20] Y. Zhang, J. Zhou, Y. Yin, X. Shen, X. Ji, Multi-GPU implementation of a cellular automaton model for dendritic growth of binary alloy. Journal of Materials Research and Technology **14**, 1862-1872 (2021).
DOI: https://doi.org/10.1016/j.jmrt.2021.07.095

[21] J.K. Mason, J. Lind, S.F. Li, B.W. Reed, M. Kumar, Kinetics and anisotropy of the Monte Carlo model of grain growth. Acta Mater. **82**, 155-166 (2015).
DOI: https://doi.org/10.1016/j.actamat.2014.08.063

[22] O. Vodka, M. Shapovalova, Exploration of cellular automata: a comprehensive review of dynamic modeling across biology, computer and materials science. Computer Methods in Material Science **23**, 57-80 (2023).
DOI: https://doi.org/10.7494/cmms.2023.4.0820

[23] F. Lin, M. Sitko, L. Madej, L. Delannay, Non-uniform Grain Boundary Migration During Static Recrystallization: A Cellular Automaton Study. Metall. Mater. Trans. A Phys. Metall. Mater. Sci. **53**, 1630-1644 (2022).
DOI: https://doi.org/10.1007/s11661-022-06599-0

[24] J.K. Mason, Grain boundary energy and curvature in Monte Carlo and cellular automata simulations of grain boundary motion. Acta Mater. **94**, 162-171 (2015).
DOI: https://doi.org/10.1016/j.actamat.2015.04.047

[25] K. Kremeyer, Cellular Automata Investigations of Binary Solidification. J. Comput. Phys. **142**, 243-263 (1998).
DOI: https://doi.org/10.1006/jcph.1998.5926

[26] Z. Li, J. Wang, H. Huang, Grain boundary curvature based 2D cellular automata simulation of grain coarsening. J. Alloys Compd. **791**, 411-422 (2019).
DOI: https://doi.org/10.1016/j.jallcom.2019.03.195

[27] S. Niewczas, M. Sitko, L. Madej, Influence of the Grain Boundary Curvature Model on Cellular Automata Static Recrystallization Simulations Szymon Niewczas. Key Eng. Mater. **926**, 1977-1985 (2022). DOI: https://doi.org/https://doi.org/10.4028/p-2v2esd