How to use prior knowledge for injection molding in industry 4.0

Párizs R. D., Török D.

Contents lists available at ScienceDirect

Results in Engineering

journal homepage: www.sciencedirect.com/journal/results-in-engineering

How to use prior knowledge for injection molding in industry 4.0

Richárd Dominik Párizs [a], Dániel Török [a,b,*]

[a] Department of Polymer Engineering, Faculty of Mechanical Engineering, Budapest University of Technology and Economics, Műegyetem Rkp. 3., H-1111, Budapest, Hungary

[b] MTA-BME Lendület Lightweight Polymer Composites Research Group, Műegyetem Rkp. 3., H-1111, Budapest, Hungary

ARTICLE INFO

Keywords:
Injection molding
Reinforcement learning
Actor-critic algorithm
Industry 4.0
Self-adjustment

ABSTRACT

Searching for the optimal injection molding settings for a new product usually requires much time and money. This article proposes a new method that uses reinforcement learning with prior knowledge for the optimization of settings. This method uses an actor-critic algorithm for the optimization of the filling phase and the holding phase. For five different injection molded products, the filling phase and holding phase were adjusted with the above-mentioned method. The learning algorithm optimized the settings for one product (pre-learning) and used this acquired knowledge (prior knowledge) to optimize the injection molding settings for a new product (post-learning). This research shows that the method is able to optimize the injection molding parameters in a reasonable time when prior knowledge is derived from a product with a different material, gate design or even geometry. On average, less than 16 injection molding cycles were needed for the algorithm to optimize the filling phase and less than 10 cycles to optimize the holding phase. The presented method can greatly facilitate the development of self-adjusting injection molding machines.

1. Introduction

Injection molding is a very popular polymer processing technique and is used to manufacture a large proportion of plastic products [1]. There is a complex relationship between the technological parameters, which should be investigated before mass production [2]. If the technological parameters are set incorrectly, it may lead to a number of defects during production, such as warpage, flow marks, short shot, and sink marks [3]. Besides, the control of different injection molding machines can differ, so the required machine settings can differ as well [4]. Also, there are several types of polymers with different thermal properties, requiring different process parameters [5]. Furthermore, the quality of the product may change if injection molding settings are changed [6]. The design and material of the mold can also have a significant influence on the injection molding process [7]. For example, the design of the gate can have a great effect on process optimization [8]. Finding the proper machine settings is therefore not easy. If the cavity is filled too fast, it can produce flash on the product; in contrast, filling the cavity too slowly can cause a short shot [9]. However, all these potential problems exist because injection molding can use a wide range of materials and technological parameters. Different materials have different mechanical and thermal properties due to their different molecular structure [10]. Of course, if someone wants to exploit the true potential of polymers, the properties of the selected polymer must be taken into account when a new mold is designed for a new product [11].

In order to model a complex and multi-parameter process such as injection molding, methods to analyze the effects of the parameters and their interactions are needed. One option is to use traditional mathematical techniques. Mukras et al. [12] used a central composite design for seven injection molding parameters to optimize the shrinkage and warpage of the products. Roy and Li [13] created an information model based, among other things, on the properties of the polymer to reduce waste during injection molding. Chen et al. [14] showed how to use sequential design to optimize injection molding. Barghash and Alkaabneh [15] created regression models that describe the relationship between the selected quality parameters and injection molding settings.

Another possibility is to investigate the effect of injection molding parameters on simulated results rather than actual results. With today's computing power, several finite element–based software packages are available to simulate injection molding [16]. Simulations can be useful when it needs to be analysed or shown how the mold is filled with the material during injection molding [17]. With injection molding

* Corresponding author. Department of Polymer Engineering, Faculty of Mechanical Engineering, Budapest University of Technology and Economics, Műegyetem rkp. 3., H-1111, Budapest, Hungary.
E-mail address: torok@pt.bme.hu (D. Török).

https://doi.org/10.1016/j.rineng.2024.102667
Received 31 May 2024; Received in revised form 30 July 2024; Accepted 1 August 2024
Available online 6 August 2024

simulations, experimental designs can be planned, such as the Taguchi analysis [18]. Furthermore, injection molding simulations can be useful for mold materials with non-linear elastic behavior [19]. Of course, different software packages may use different boundary conditions, so the results may differ [20]. It is important to note that simulation results can differ significantly from actual injection molding results. Still, the effect of process parameters from simulations can be transferred to injection molding [21].

In contrast to traditional methods, machine learning–based solutions have recently become popular [22]. Gim and Rhee [23] used a neural network to model the relationship between in-mold pressure and part weight. Gao et al. [24] showed the versatility of machine learning methods for conformal cooling channels in injection molds. They used machine learning to design cooling systems for more uniform product temperature during injection molding. In addition, numerous studies have used machine learning to predict product quality [25–27].

In the early 2010s, the concept of Industry 4.0 emerged, with no precise definition, but the term refers to the 4th industrial revolution and includes manufacturing based on communication and interaction between machines and sensor-based manufacturing processes, among others [28]. Industry 4.0 is also a commonly used concept in injection molding. Khosravani et al. [29] interpreted the concept as a combination of injection molding, 3D printing, virtual reality, and generative design. The intelligent system they described was able to speed up production and reduce production cost. Rousopoulou et al. [30] created a control system that uses cognitive analytics and machine learning for the detection of anomalies and can retrain itself if needed. Farahani et al. [31] consider Industry 4.0 the processing and analysis of sensor data. They focused on detecting different malfunctions during injection molding by using features extracted from sensor data.

According to Benešová and Tupa [32], the 4th industrial revolution will change job opportunities in the industrial environment, as it happened in the former revolutions. Combemale et al. [33] think that machines will do jobs that require low and medium levels of skill in the future. With regard to injection molding, a control system capable of setting up the machine may be necessary due to the possible changes in jobs.

Reinforcement learning algorithms are a subset of machine learning methods that interact with the environment to learn how to perform specific tasks [34,35]. Therefore, these algorithms could be appropriate control systems under the Industry 4.0 principle. There are already a few examples in the literature on how reinforcement learning can be used for injection molding: for process optimization [36], for production scheduling [37], and for regulation in the case of non-optimal products [38]. One may ask, of course, what is the difference between reinforcement learning–based control systems and conventional control systems. According to Ugurlu et al. [39], a reinforcement learning–based system can control itself better in a new environment, but traditional control systems are more stable. For example, PID regulators can be used effectively in simple, linear changing environments, but these controllers are not recommended for sudden changes in the environment [40]. In contrast, one of the greatest disadvantages of reinforcement learning–based regulators is the need for a lot of training samples and time [41].

For a brand-new product, setting up the injection molding machine is often based on the experience of the technician and trial and error. However, these types of tasks are likely to performed by machines over time as a result of the Industry 4.0 ideology. This study, therefore, explores a novel method, which can be used to set up the injection molding machine for a new product at different phases of the injection molding cycle. This method is based on reinforcement learning; more precisely, it is actor-critic algorithm–based, and the main idea is to use prior knowledge that the algorithm learns from a previous product and use it for a new one. This study shows how to use the proposed method to optimize the filling and the holding phase with products injection molded from different materials, with different types of gates, part thicknesses, part geometries, and sizes.

2. Materials and methods

2.1. Injection molding and tests

For the experiments, several injection molds (see Fig. 1) and injection molding machines were used: an Arburg Allrounder Advance 270 S 400–170, an Arburg Allrounder 320C 400–170, an Arburg Allrounder 420C 1000–290, and an Arburg Allrounder 470 A 1000-290 machine (from Arburg GmbH + Co., Loβburg, Germany). Terluran GP-35 acrylonitrile butadiene styrene (ABS) and Ingeo Biopolymer 3100 HP polylactic acid (PLA) were used for the injection molding tests. All parts were made from ABS, but the lid parts (Fig. 1 b) and plates with a fan gate (Fig. 1 e) were made from PLA, too. Two series of experiments were performed; filling phase optimization with the first series, where plate-like products and the lid product were used (Fig. 1 b). With the second series of experiments, the holding phase was investigated—the small lid, the lid, and the 1 mm thick plate were used for these tests (Fig. 1 a, b, d).

2.1.1. Injection molded parts for the examination of the filling phase

To investigate the use of prior knowledge from different products, injection molded parts with three different gate designs and with three different thicknesses but the same base geometry (80 mm × 80 mm) were used. Plates were made (with a fan gate) from PLA to investigate the effect of the different material (PLA instead of ABS), and the effect of a completely different geometry on the performance of the method was investigated on injection molded lid products. The aim of the research was to teach the learning algorithm how to use the selected injection molding settings to make filled products and use the acquired knowledge for another product. The 1 mm and 2.5 mm thick plates had a film gate, and other 1.2 mm thick plate products with fan and double-edge gates were used during the tests. In the filling experiments, in-mold pressure was measured with two different types of in-mold pressure measurement systems: the Kistler CoMo system (Kistler Group, Winterthur, Switzerland) and Cavity Eye in-mold sensors (Cavity Eye Ltd., Kecskemét, Hungary). For each setting combination, a photo of the product was taken with a Microsoft LifeCam Cinema webcam (Microsoft Corporation, Redmond, Washington, USA) after mold opening. Two injection molding parameters were changed for the investigation of filling: injection rate and switch-over volume. With these settings, short shots (partially filled products), overfilled products (too high in-mold pressure) and properly filled products were made. The switch-over volumes were the following: 22, 20, 18, 16, 14, 12, 10, 9, 8.5, 8, 7.5, 7, 6.5 and 6 cm^3. Four different injection rate settings were used with each switch-over setting (15, 30, 45 and 60 cm^3/s). The injection molding settings can be seen in Table 1. Five samples were made with each setting combination of the 1.2 mm thick plate and three samples with the 1 mm thick plate. From resampling, it was clear that there was no significant difference between the standard deviation of each setting combination. Therefore, no more 2.5 mm thick plate samples were molded, except for one setting combination, and used that variance for each of the other setting combinations. In the case of the lid product and the plate from PLA, three samples were made for each switch-over volume with a given injection flow, and used these data to predict machine noise. The measured and collected data with these settings were used to simulate the learning of the algorithm.

2.1.2. Injection molded parts for the examination of the holding phase

Products of different sizes, geometries and materials were injection molded in the investigation of the impact of prior knowledge on the algorithm in optimizing the holding phase. Three different products were used: a lid with a complex geometry, a 1 mm thick plate, and a small lid with a simple geometry, but the mold had 16 cavities (Fig. 1b–d, and a, respectively). Test specimens of the complex lid geometry using ABS and PLA were also molded. During the experiments,

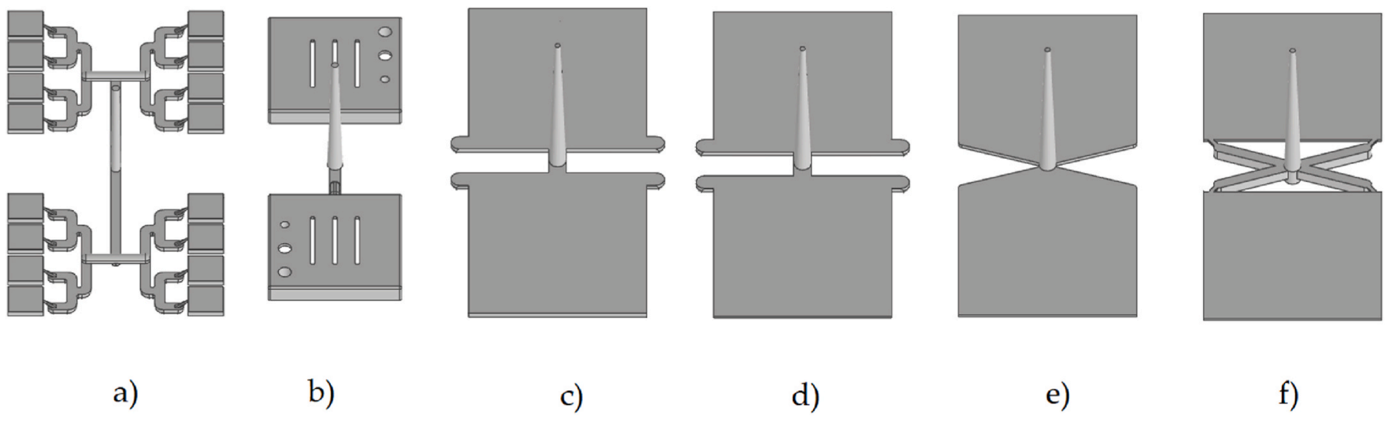

Fig. 1. The model of the injection molded products a) small lid with 16 cavities b) lid product c) plate with a film gate (2.5 mm) d) plate with a film gate (1 mm) e) plate with a fan gate (1.2 mm) f) plate with a double gate (1.2 mm).

Table 1
The injection molding settings for the filling experiments.

Process parameter	Plate 1.2 mm (fan gate ABS)	Plate 1.2 mm (double gate)	Plate 1.2 mm (fan gate PLA)	Plate 1 mm (film gate)	Plate 2.5 mm (film gate)	Lid (ABS)
Injection molding machine	270 S	270 S	270 S	420C	420C	320C
Material	ABS	ABS	PLA	ABS	ABS	ABS
Shot volume [cm^3]	30	30	30	30	44	30
Peripheral speed of screw [m/min]	25	25	25	25	25	25
Back pressure [bar]	60	60	30	30	30	60
Decompression [cm^3]	5	5	5	5	5	5
Injection flow [cm^3/s]	15/30/45/60	15/30/45/60	15/30/45/60	15/30/45/60	15/30/45/60	15/30/45/60
Switch-over volume [cm^3]	6–22	6–22	6–22	6–22	6–22	6–22
Injection pressure limit [bar]	1700	1700	1700	1700	1700	1700
Clamping force [kN]	400	400	400	600	600	400
Holding pressure [bar]	25	25	25	25	25	25
Holding time [s]	7.2	7.2	7.2	7.1	7.1	6.1
Holding flow [cm^3/s]	0	0	0	0	0	0
Cooling time [s]	10	10	45	20	20	10
Cycle time [s]	23.5	23.5	58.5	47	47	22.5
Melt temperature [°C]	225	225	200	225	225	220
Mold temperature [°C]	40	40	25	40	40	40
Sensor type	Kistler	Kistler	Kistler	Cavity Eye	Cavity Eye	Cavity Eye

the holding pressure and time were changed, as these settings define the holding phase, and product weight was measured with an Ohaus Explorer analytical balance (OHAUS Europe GmbH, Uster, Switzerland). Holding pressure was changed between 0 bar and 1000 bar, typically with 100 bar steps. The plate and the small lid products were the exception to this. In the case of the former, there were flashes on the products at holding pressures above 800 bar. In the case of the latter, the weight measurement for a 16-cavity mold is tedious, therefore only four holding pressures was used: 0 bar, 200 bar, 600 bar, and 1000 bar. Holding time was changed with steps of 0.5 s between 0 s and 5 s. The 16-cavity small lid parts were again the exception to this because part weight does not change significantly above a holding time of 3 s. The plate had a film gate that froze more slowly, therefore the maximum holding time for this part was 6 s. The other injection molding settings can be seen in Table 2. With each setting combination, five samples were produced in the case of the ABS lid product and the small lid product. As with filling, it was clear that the variance of the data does not change significantly with different settings. Therefore, in the case of the plate product, three samples were produced with each holding time setting at one holding pressure level (400 bar), and the variance in the search space is estimated from these data. In the case of the lid from PLA, three samples were produced with each holding time setting with three holding pressure levels (0 bar, 500 bar, 1000 bar).

2.2. *Digital image processing*

For the filling phase optimization, the learning algorithm uses the in-

Table 2
The injection molding settings for the holding experiments.

Process parameter	Lid (ABS)	Lid (PLA)	Plate 1 mm (film gate)	16-cavity small lid
Injection molding machine	320C	320C	420C	470 A
Material	ABS	PLA	ABS	ABS
Shot volume [cm^3]	30	30	30	26
Peripheral speed of screw [m/min]	25	25	25	25
Back pressure [bar]	60	60	30	40
Decompression [cm^3]	5	3	5	5
Injection flow [cm^3/s]	30	40	30	50
Switch-over volume [cm^3]	9	9.5	7.5	5.8
Injection pressure limit [bar]	1500	1500	1700	1500
Clamping force [kN]	400	400	600	700
Holding pressure [bar]	0–1000	0–1000	0–800	0–1000
Holding time [s]	0–5	0–5	0–6	0–3
Holding flow [cm^3/s]	40	40	30	25
Cooling time [s]	10	30	20	18
Cycle time [s]	22.5	42	41	30
Melt temperature [°C]	220	200	225	225
Mold temperature [°C]	40	25	40	40
Product weight goal [g]	8.930	10.655	7.200	0.4745

mold pressure and the image of the product. Therefore, the product had to be detected, from which the algorithm would know how much the cavity is filled. For this, the Matlab R2023b platform (MathWorks Inc., Natick, Massachusetts, USA) was used. The part was recognized by the image detection algorithm, which had the following preprocessing steps: transformation to a binary image, sprue recognition, gate detection, and noise filtering (Fig. 2). The presented image detection algorithm can be seen in Table 3.

By binarizing the image in one step, many things were filtered out that are not part of the product, such as mold geometry, fastening screw, and so on. Of course, some features cannot be deleted this way, such as the ejector pins or flashback. Only the upper product is needed for the investigation, so first, the images were divided into upper and lower parts. The sprue is in the middle of the two products, so finding this could help detect the selected product. Therefore, the image processing algorithm searches the binarized image from the top left corner for the first white pixel in each row (i.e. the left side of the sample). The result is a function which can be plotted as the number of the vertical pixels (rows) in the image (Fig. 3). Near the middle of this function is a local extremum, which shows the location of the sprue.

Once the location of the sprue is known in the picture, only the part of the image above it is needed, since that is where the selected product is. This is followed by gate detection. For this, the algorithm had to consider that the function of the first white pixels along the product does not change significantly. Therefore, the algorithm calculated the derivative of this function; when the derivative crossed a threshold value, it indicated the start of the part and the end of the gate (Fig. 4).

When the location of the gate is known, only the parts of the image above it are important, as these are the parts of the product. In this selected area, the algorithm checks the connectivity between the white pixels. Usually, there are many white pixels in the selected region, but some are not part of the product. These smaller white areas are the ejectors or reflections of light (Fig. 5 a). The separate small areas have been filtered out. With the double gate design, the product may consist of two individual pieces for a relatively large switch-over volume setting (when little material is injected into the mold). The algorithm handled this case by treating the two areas at the bottom of the image as identical (Fig. 5 b).

The remaining white pixels on the image are the product. Therefore, the number of these white pixels is a good measure of the filling of the cavity. To prove this, the weight of the products (with one specific injection flow setting for each product geometry) was measured and compared to the number of pixels detected by the algorithm. These relations (Fig. 6) are strongly positive linear at the significance level of 0.05, but there is a strong saturation of the pixel numbers as the switch-over volume decreases. This phenomenon is not so visible with the

Table 3
The pseudocode of the image processing algorithm.

```
Input:   Load the image
Input:   Define the threshold value for binarization: $\delta_{bin}$
Input:   Define the threshold value for gate detection: $\delta_{gate}$
1:       Transform the color image into a grayscale image
2:       Transform the greyscale image into a binary image: $BI = \begin{cases} 1 \text{ if } pixel \geq \delta_{bin} \\ 0 \text{ if } pixel < \delta_{bin} \end{cases}$
3:       Loop over each row of the image ($x$)
4:           Find the first white pixel in the given row: $f(x)$
5:       End loop.
6:       Find the location (row of the image) of the sprue in the interquartile range of rows: $x_{sprue} = \underset{x\in\chi_{sprue}}{\operatorname{argmax}}(f(x))$ where $\chi_{sprue} = [0.4 \bullet x_{max}, 0.6 \bullet x_{max}]$
7:       Calculate the derivative of the function: $f'(x) = df(x)/dx$ where $x \in [1, x_{sprue}]$
8:       Find the location (row of the image) of the gate, where $f'(x) > \delta_{gate}$ for the first time in the range $x \in [0.8 \bullet x_{sprue}, x_{sprue}[$
9:       Label the different regions of connected white pixels above the gate
10:      Filter out regions with fewer white pixels.
11:      Count the number of white pixels: $n_{pixel}$.
Output:  $n_{pixel}$
```

weight data (except in the case of the lid product) because during injection molding, when the melt fills the cavity, injecting more material can compress the cushion of material in the mold, and thus, part weight increases. However, this kind of mass increase during filling is not desirable, as it causes residual stress in the product, and/or the mold may open due to the high in-mold pressure, which causes flash on the product. The fine-tuning of product weight is recommended with the holding phase optimization because the control of that phase is more suitable for this.

2.3. In-mold pressure processing

For filling phase optimization, in-mold pressure sensors were used to detect overfilling and the filling time of the cavity. The injection molds have several in-mold sensors but the sensor positions are not the same for each product design, so two sensors for each product were choosen for the measurements, one closest to the gate and one closest to the center of the product. This means sensors CH2 and CH4 for the 1.2 mm thick plates (Fig. 7 a), sensors F1K and F2K for the 1 mm and 2.5 mm thick plates (Fig. 7 b), and sensors C1SG andC1S2 for the lid product (Fig. 7 c) were used.

For the detection of overfilling, the maximum pressure of the middle sensors (CH2, F2K, and C1S2 in Fig. 7) were used. The best choice would be the end-of-cavity sensors, but in the case of the Kistler sensors, there

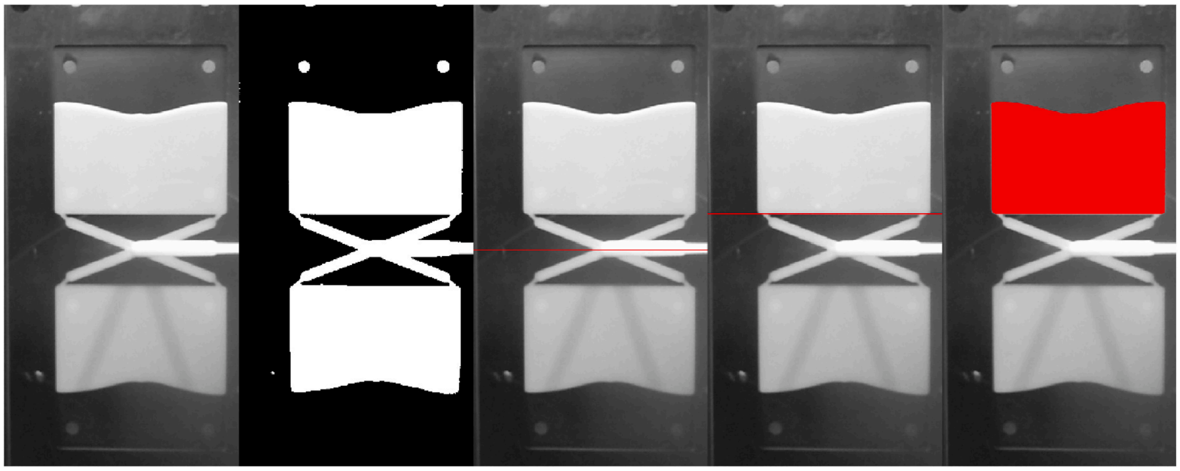

Fig. 2. The steps of image processing (from left to right) transformation to binary image, sprue recognition, gate detection, noise filtering.

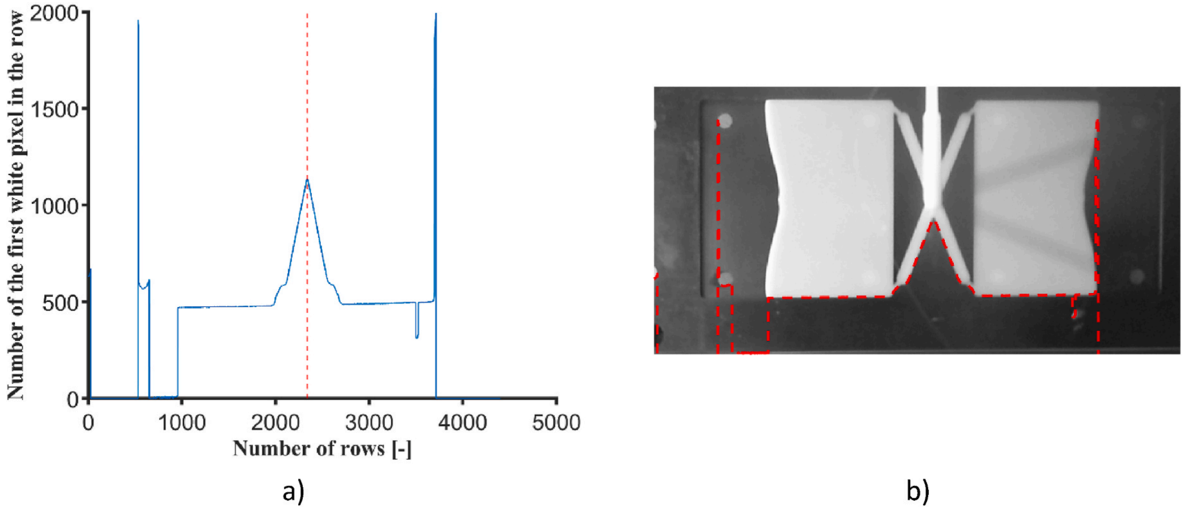

Fig. 3. a) The function of the first white pixel on the binarized image b) the function on the original rotated image.

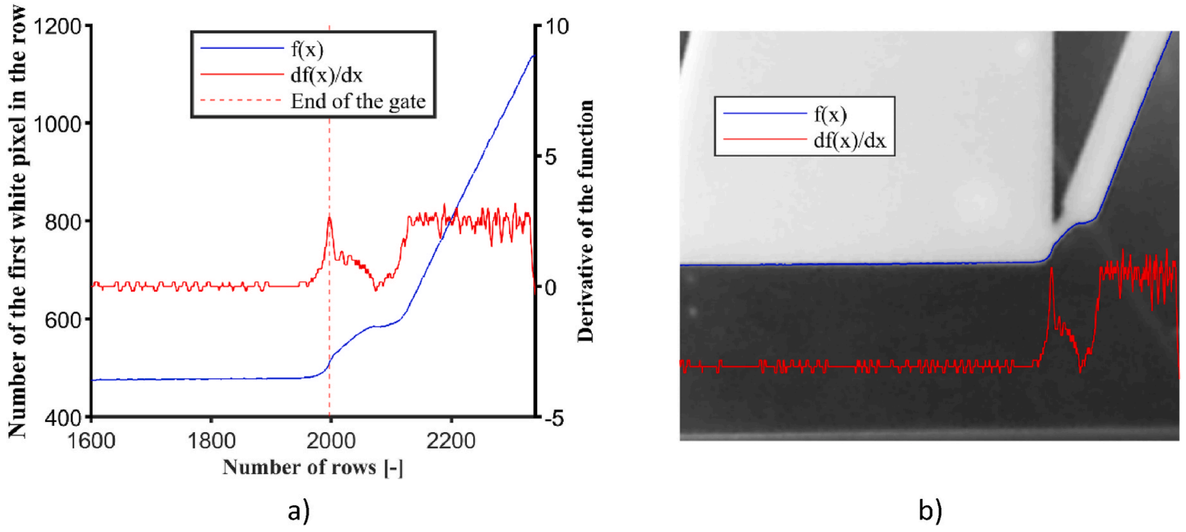

Fig. 4. a) The function of the first white pixel on the binarized image and the derivative b) the function and the derivative on the original rotated image.

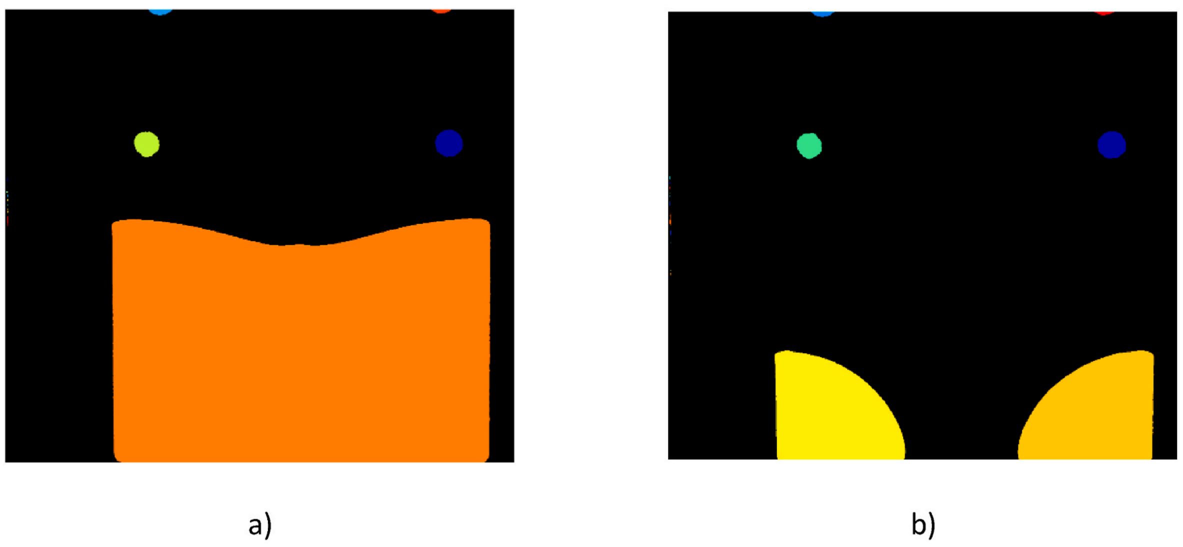

Fig. 5. a) The differently labeled areas in the binarized image b) The differently labeled areas with two separate parts of the product at the bottom.

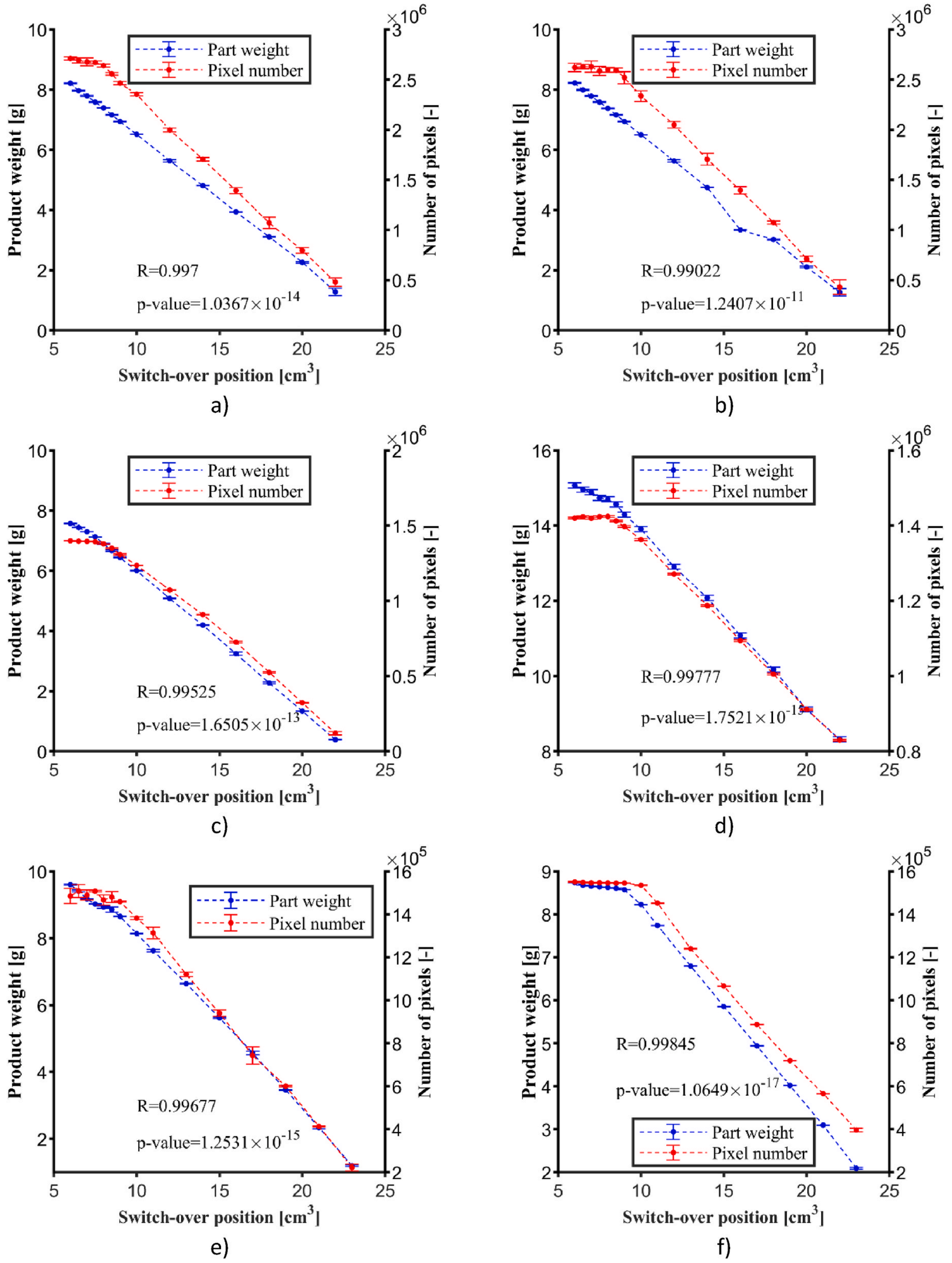

Fig. 6. The relationship between the weight and the pixel number of the detected product a) plate (1.2 mm) with a fan gate (ABS) b) plate (1.2 mm) with a double gate c) plate (1 mm) with a film gate d) plate (2.5 mm) with a film gate e) plate (1.2 mm) with a fan gate (PLA) f) lid product (ABS).

were no end-of-cavity sensors. Therefore, the middle sensors in each case were measured and investigated. The gate (CH4, F1K, and C1SG) and middle sensors (CH2, F2K, and C1S2) were used to calculate filling time (Fig. 8). For both sensors, the time was determined when the pressure at the sensor reached or exceeded 5 bar for the first time, and the difference between the time measured at the two sensors is called filling time (t_{fill}). The definition of the states derived from the image processing and the evaluation of pressure curves is discussed in detail later in section 2.4.1.

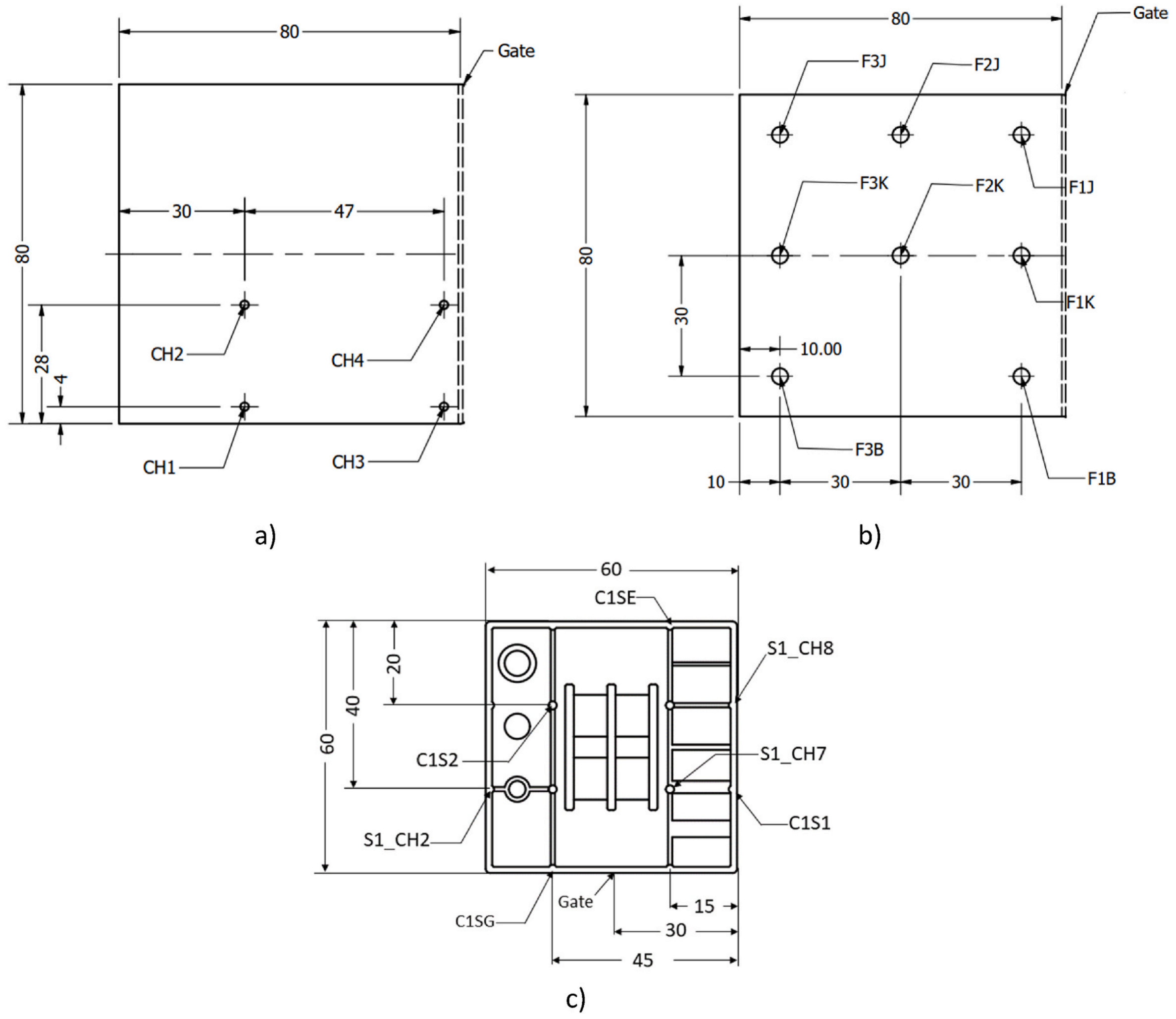

Fig. 7. The sensor map of the products a) plate (1.2 mm thick) with a fan gate and a double gate (Kistler sensors) b) plate (1 mm and 2.5 mm thick) with a film gate (Cavity Eye sensors) c) lid (Cavity Eye sensors).

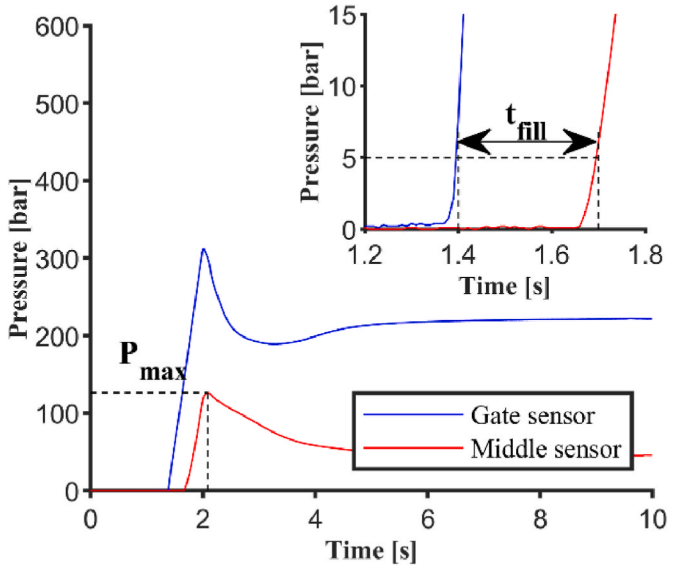

Fig. 8. In-mold pressure and the defined features (maximum pressure, fill time).

2.4. Learning algorithm

The aim of this study was to show how to use prior knowledge for injection molding derived from another mold. For this, a learning algorithm was needed to decide what settings to change and how much to change them. For this purpose, an actor-critic algorithm was made whose operation can be seen in Table 4. The actions of the algorithm are the changes in injection molding setting combinations. The changed injection molding settings for filling optimization are switch-over volume and injection flow. For the holding phase optimization, the varied settings are holding pressure and holding time. During the filling optimization, the state was defined based on the size of the detected product from the image, the measured maximum pressure, and the measured filling time. These values were used as normalized values compared to the ideal or target values. For holding phase optimization, the state means the percent of product weight compared to the target weight. The algorithm uses state aggregation, i.e., it groups states close to each other and evaluates them similarly. One can imagine this as if the products of 15 g and 16 g are equally good for the algorithm, but products of 20 g weight have a different goodness to the algorithm. The goodness of a state is calculated in the state–value function: $\hat{v}(s, \underline{w})$, where $\underline{x_s}$ is an index vector that shows which aggregated state the algorithm is in, and $\underline{w}$ is the weight vector of the aggregated states. The policy function approximation ($\pi(a|s,\theta)$) uses this state aggregation too, because it calculates the goodness of each action for a given aggregated state. In the

Table 4
The pseudocode of the presented learning algorithm.

Input:	Initial weight vectors for function approximation: $\underline{w} \in \mathbb{R}_*^{n+}$, $\underline{\theta} \in \mathbb{R}_*^{n+}$
Input:	Value function: $\widehat{v}(s,\underline{w}) = \underline{x_s} \bullet \underline{w}$
Input:	Policy function: $\pi(a\vert s,\underline{\theta}) = \exp\left(\underline{x_h}(s,a) \bullet \underline{\theta}\right) / \sum_{b\in\mathscr{A}} \exp\left(\underline{x_h}(s,b) \bullet \underline{\theta}\right)$
Input:	Initial average reward: $\overline{R} = 0$, target state s_{goal}, end of the learning t_{end}
Input:	Target value: $s_{goal} \in \mathbb{R}^+$
Input:	Learning rate: $\alpha \in \mathbb{R}^+$
Input:	Initial state based on initial injection molding settings: $s_{initial} \in \mathbb{R}_*^+$
Input:	Random noise on the states: $\sigma \in \mathbb{R}_*^+$
Input:	End of the learning: $t_{end} \in \mathbb{Z}^+$
Input:	Whether there is a restart in learning: $n_{restart}$, $t_{step} \in \mathbb{Z}^+$
1:	Initial step definition: $t = 1, s_t = s_{initial} \in \mathbb{R}_*^+$
2:	Repeat until $t = t_{end}$
3:	Check if restart is needed and if so $s_{t+1} = s_{initial}$
4:	$a\leftarrow$ Based on policy π
5:	Observe s_{t+1} based on a,σ
6:	Calculate R from s_{t+1} and s_{goal}
7:	$\delta\leftarrow R - \overline{R} + \widehat{v}(s_{t+1},\underline{w}) - \widehat{v}(s_t,\underline{w})$ (Temporal difference error)
8:	$\overline{R}\leftarrow\overline{R} + \alpha \bullet \delta$ (update average reward)
9:	$\underline{w}\leftarrow\underline{w} + \alpha \bullet \delta \bullet \underline{\nabla\widehat{v}}(s_t,\underline{w})$ (update state weight)
10:	$\underline{\theta}\leftarrow\underline{\theta} + \alpha \bullet \delta \bullet \underline{\nabla \ln \pi}(a\vert s_t,\underline{\theta})$ (update state-action weight)
11:	$s_t\leftarrow s_{t+1}$
12:	$t\leftarrow t+1$
Output:	$\underline{w}, \underline{\theta}, R\,\overline{R}, s$ for each t

policy function, $\underline{x_h}$ is an index vector that shows in which state the algorithm chooses which action and $\underline{\theta}$ is the weight vector for each state–action combination. The initial values for $\underline{w}$ and $\underline{\theta}$ are 0.5 for each discrete state and state–action pair. During learning, the algorithm updates the $\underline{w}$ and $\underline{\theta}$ vectors to estimate the value of each state and state–action combination better. With more accurate estimation, the algorithm can make better and better decisions in each state, and the output will be closer to the target value.

A restart option was also included in the algorithm. With this option, it resets itself to the initial injection molding settings after a given learning steps (t_{step}) at the specified time ($n_{restart}$). Thanks to this option, it was possible to analyze whether repeating the injection molding settings at the beginning of the learning process helps with learning a new product. Consequently, when teaching the algorithm, data from real injection molding experiments was used, and when the algorithm tried to use settings that fall between measurement points, it used interpolation for teaching. This way, later in practice, through a design of experiments, one can perform the pre-learning of the algorithm (with offline data) and use the learned policy for post-learning with online data (in real-time production). This approach (see Fig. 9.) is based on data processing and machine-to-machine communication, which are the building blocks of Industry 4.0.

The final goal, of course, will be to connect the algorithm to the injection molding machine so that it can set up the machine in real-time production. However, this study used pre-measured data (offline data) because it made the analysis easier and faster. So, the algorithm figures out how it would change the machine settings, and the output value(s) associated with the resulting settings are retrieved from the database. In reality, of course, the injection molding machine cannot produce a product of exactly the same quality even in two consecutive cycles, as there may be some variation in the melt, and the control of the machine can only work with a certain accuracy. Therefore, the algorithm gives a random noise to the value of the database when it is selected. The random noise is normally distributed with a mean of zero and with the variance defined from the measured standard deviation. This way, the random noise of the injection molding technology was also considered during the experiments.

For the optimization of the filling phase, the learning algorithm changes the injection flow and the switch-over volume. The smaller the switch-over volume, the more melt is injected into the cavity. However, after a certain switch-over volume, the cavity becomes filled. Then, by reducing the switch-over volume, the melt is compressed in the cavity, which increases the in-mold pressure. The injection flow changes the speed of the melt injected into the cavity. Due to the changes in the melt flow, the shear stress in the melt changes too, and so does the viscosity, temperature and specific volume of the melt. Of course, the greater the flow, the faster the melt fills the cavity. Therefore, the algorithm will learn how to vary the aforementioned settings for different filling conditions.

The learning algorithm can change the holding pressure and holding time for the optimization of the holding phase. With these settings, the injection molding machine can control how much additional melt is injected into the cavity after filling. In this injection molding phase, the melt will shrink due to cooling, so it will not fill the cavity accurately. The additional melt will change the mass of the product but only slightly and reduce the effect of shrinkage on the final shape of the product. The

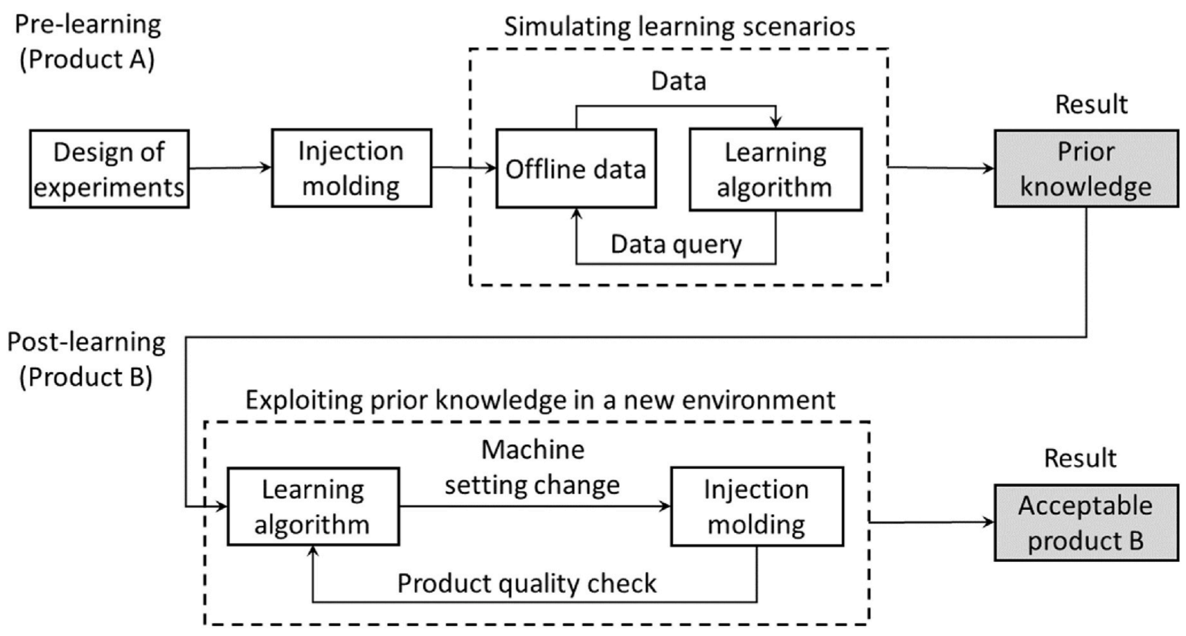

Fig. 9. A reinforcement learning–based method for setting up the injection molding machine for a new product.

greater the holding pressure, the more melt can be injected into the cavity. However, holding pressure is only maintained for as long as the holding time is set. Therefore, increasing holding time will also increase the amount of melt injected into the cavity additionally. It is important to note that this phenomenon only persists until the gate (which separates the product from the channel) freezes. Therefore, the algorithm can change the product's weight with these two parameters to different degrees.

In a sense, the essence of a learning algorithm is the reward function. The reward is the feedback for the algorithm from the environment. The algorithm will learn which decisions are good or bad based on the reward given for the actions. The reward of the algorithm can be interpreted as a punishment because the algorithm aims for 100 % (with a given tolerance), so deviations from this are penalised in all cases. Therefore, the reward function was described as (1).

$$R = -|S_{actual} - 100|, \tag{1}$$

where R is the reward function and S_{actual} is the actual state value for filling (S_{fill}) or holding (S_{hold}). It is clear form (1) that 100 is the goal value because the punishment is a negative difference from this. The negative sign is necessary because of the convergence of learning.

The decisions of the algorithm are stochastically based on the policy function $\pi(a|s,\theta)$. Therefore, 100 different learning scenarios were made for each learning case. The performance of the algorithm can be shown with the actual state of the given step (Fig. 10 a) or the collected reward (Fig. 10 b). The figures show the median value and the interquartile range. The two plots have the same interpretation and can be converted into each other with (1). However, in the filling phase optimization, the limits of the tolerance zone are unequal distances from the target state (−3% and +2 % from the target), so in this case, it is more beneficial to show the state values instead of the collected reward. In contrast, in the optimization of the holding phase, the limits are equidistant from the target state (±0.5 % from the target), so the collected rewards are shown in this case because this is a more common practice in the literature.

2.4.1. The state space for the filling phase

The state value for filling optimization is calculated from three parts. The first part of the state is calculated from the detected product from the image (2). In this equation n_{pixel} stands for the number of white pixels detected during part detection, S_1 stands for the image state, and n_{goal} stands for the number of pixels needed for the filled product in the image.

$$S_1 = \frac{n_{pixel}}{n_{goal}} * 100, \tag{2}$$

The second part of the state is calculated from filling time. Filling time is a target value, and if the machine fills the mold in less time, it is good for the learning algorithm, but it is not necessarily an advantage. Therefore, a ramp function was used (3) to transform the data. In this equation, t_{fill} stands for filling time (Fig. 8), S_2 means the time state and t_{goal} means the ideal filling time. This function transforms the ratio into a percentage of filling compared to the goal.

$$S_2 = \max\left(\frac{t_{fill}}{t_{goal}} * 100, 100\right), \tag{3}$$

The third part of the state was calculated in a similar way. This time, a pressure limit value was used, and if the measured maximum pressure was smaller than this limit, the value was transformed to 100; otherwise, the ratio of the two pressures was used (4). In this equation, P_{max} is the measured maximum pressure, S_3 stands for the pressure state and P_{limit} is the predefined pressure limit.

$$S_3 = \max\left(\frac{P_{max}}{P_{limit}} * 100, 100\right), \tag{4}$$

The state for filling can be defined from these three parts with a piecewise function (5), where S_{fill} stands for filling state. This function uses the state part from the image if the value of the pixel ratio is smaller than 97 %, the power mean of the pixel ratio, and filling time if the pixel ratio is less or equal to 99.5 % but more than 97 %. When the pixel ratio showed an almost entirely filled product (more than 99.5 %%), then the power mean of the pressure ratio and the filling time ratio was used. This way, if the cavity is not filled, the learning algorithm does not take the pressure into account since filling did not achieve its goal. However, if the cavity is filled sufficiently, the pressure and filling time will influence whether filling is good.

$$S_{fill} = \begin{cases} S_1 & \text{if } S_1 \le 97 \\ \sqrt{\dfrac{{S_1}^2 + {S_2}^2}{2}} & \text{if } 97 < S_1 \le 99.5 \\ \sqrt{\dfrac{{S_2}^2 + {S_3}^2}{2}} & \text{if } 99.5 < S_1 \end{cases}, \tag{5}$$

From function (5), the state in the test space for each product can be derived from the measured data (Fig. 11). The same filling time goal (t_{goal}) was used for each product (0.2 s) and the pressure limit (P_{limit}) was

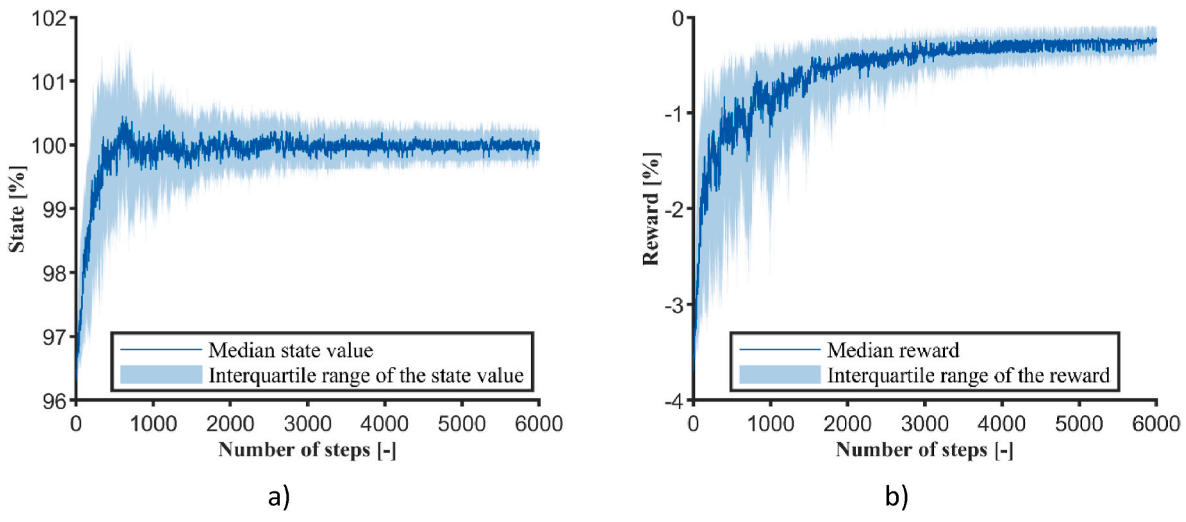

Fig. 10. a) Example of the observed state values during learning b) example of the collected rewards during learning.

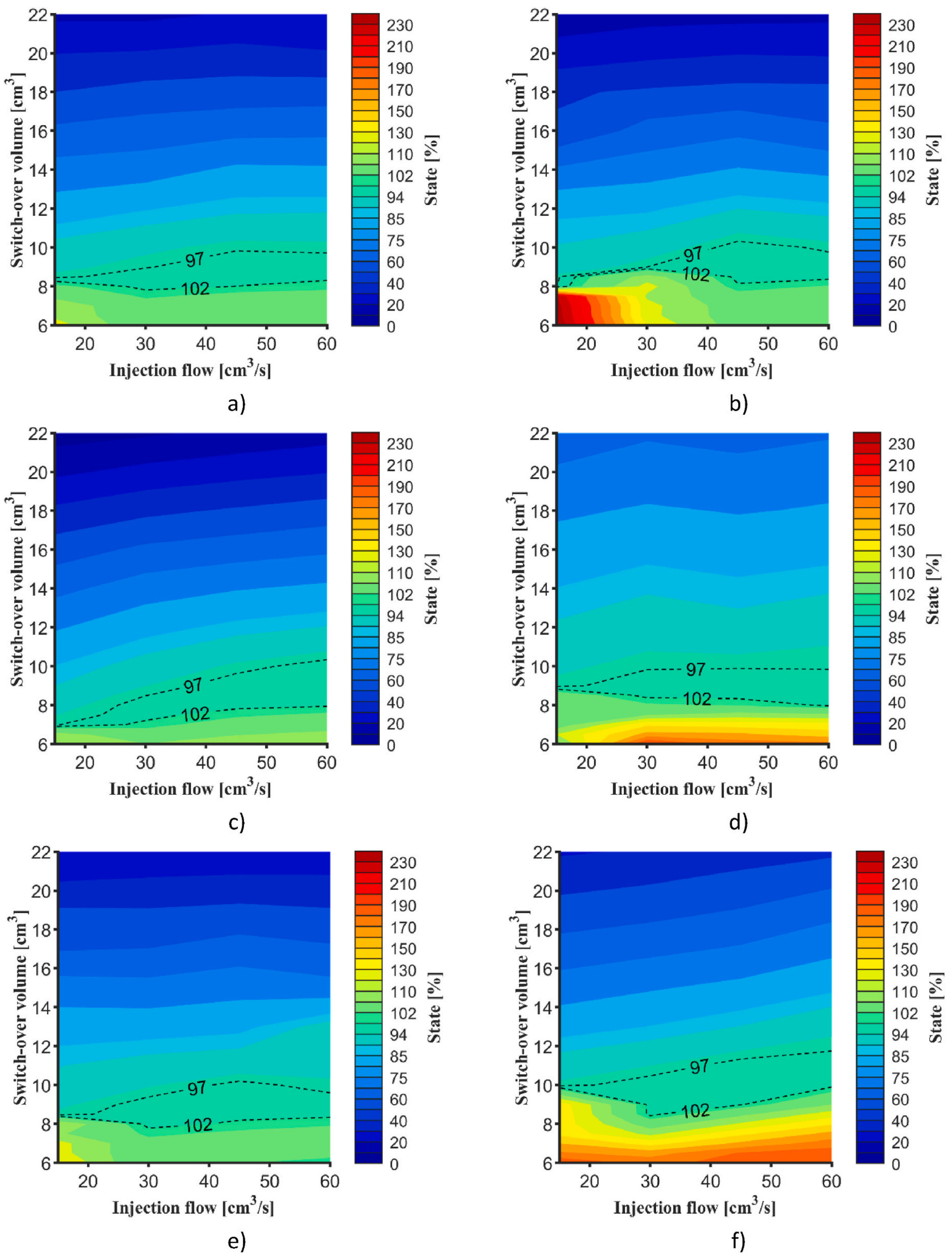

Fig. 11. The state space for different products a) plate 1.2 mm thick with a fan gate from ABS, b) plate 1.2 mm thick with a double gate, c) plate 1 mm thick with a film gate, d) plate 2.5 mm thick with a film gate, e) plate 1.2 mm thick with a fan gate from PLA, f) lid product from ABS.

the same for each product (220 bar) except for the plate with 2.5 mm thickness and the lid product. In the case of 2.5 mm thick plates, this was necessary because the pressure does not go as high with thicker products as with thinner ones, and to fill a cavity with more melt takes more time. Therefore, in the case of the 2.5 mm thick plate, the pressure limit was 50 bar, and the filling time goal was 0.4 s. In the case of lid products, the filling time goal was 0.3 s, and the in-mold pressure limit was 240 bar because this is a more complex product with a much smaller gate, more features on the product (tubes, ribs), and some of these are much thinner, while others are thicker than the original plates.

State aggregation is needed for the algorithm, and the state values were plotted according to the aggregation. In addition, the target state was shown separately with dotted lines, which means a function value of 97–102 % for the filling process (Fig. 11). In this way, the algorithm will adjust the injection molding settings so that the cavity will not be overfilled and will not be too unfilled. Another state aggregation can be used, but prior knowledge should be derived from a similar state aggregation. Fig. 11 shows significant differences between the state spaces of the products. In general, however, they are all similar in that the product is unfilled ($\ll$100 %) at high switch-over volumes, overfilled at low switch-over volumes ($\gg$100 %), and the acceptable window is greater with higher injection flow (>30 cm^3/s) than with low injection rates (<30 cm^3/s).

2.4.2. *The state space for the holding phase*

The state value for holding phase optimization can be described much more easily. For this, only the measured product weight was used. Weight measurement is very simple and can be automated easily, so it would not be very difficult to link the algorithm to the injection molding machine and a balance. Therefore, the state value was calculated from the ratio of the weight of the part produced in the actual cycle (m_{actual}) to the target weight (m_{goal}) (function (6) where S_{hold} stands for holding state). The state aggregation is shown for each product that was used for the experiments (Fig. 12). Unlike filling, the target aggregated state here is symmetric, since the goal was to make a product with a given weight within a certain tolerance (± 0.5 %)

$$S_{hold} = \frac{m_{actual}}{m_{goal}} \bullet 100 \tag{6}$$

2.4.3. *Limitations of the algorithm*

The advantage of the presented algorithm is that it can learn how to behave in a given environment and use this knowledge in a new, somewhat similar, but still different environment. One of the biggest disadvantages comes from this option. Trying to apply the acquired knowledge in a significantly different environment often leads to non-optimal decisions at the beginning of learning. This follows the need to behave differently in certain situations than in the previous environment. Such cases can arise, for example, if pre-learning is performed on non-optimized simulation data and post-learning on actual injection molding data. This can easily happen if the material, mold geometry or the technology is not accurately modeled in the simulation. During the experiments, it was shown that some variation in product geometry, material or even gate geometry does not cause such a great a variation in the environment that the presented algorithm cannot effectively handle.

There are also further limitations to the algorithm. On the one hand, the algorithm must use the same number of aggregated states or action sets during pre- and post-learning. In addition, the aforementioned aggregated states and actions must be the same for the two learning processes. For example, if in pre-learning, one aggregated state means the 90–100 % filled product and in post-learning, the same aggregated state means 10–20 % filled product, the algorithm will obviously not

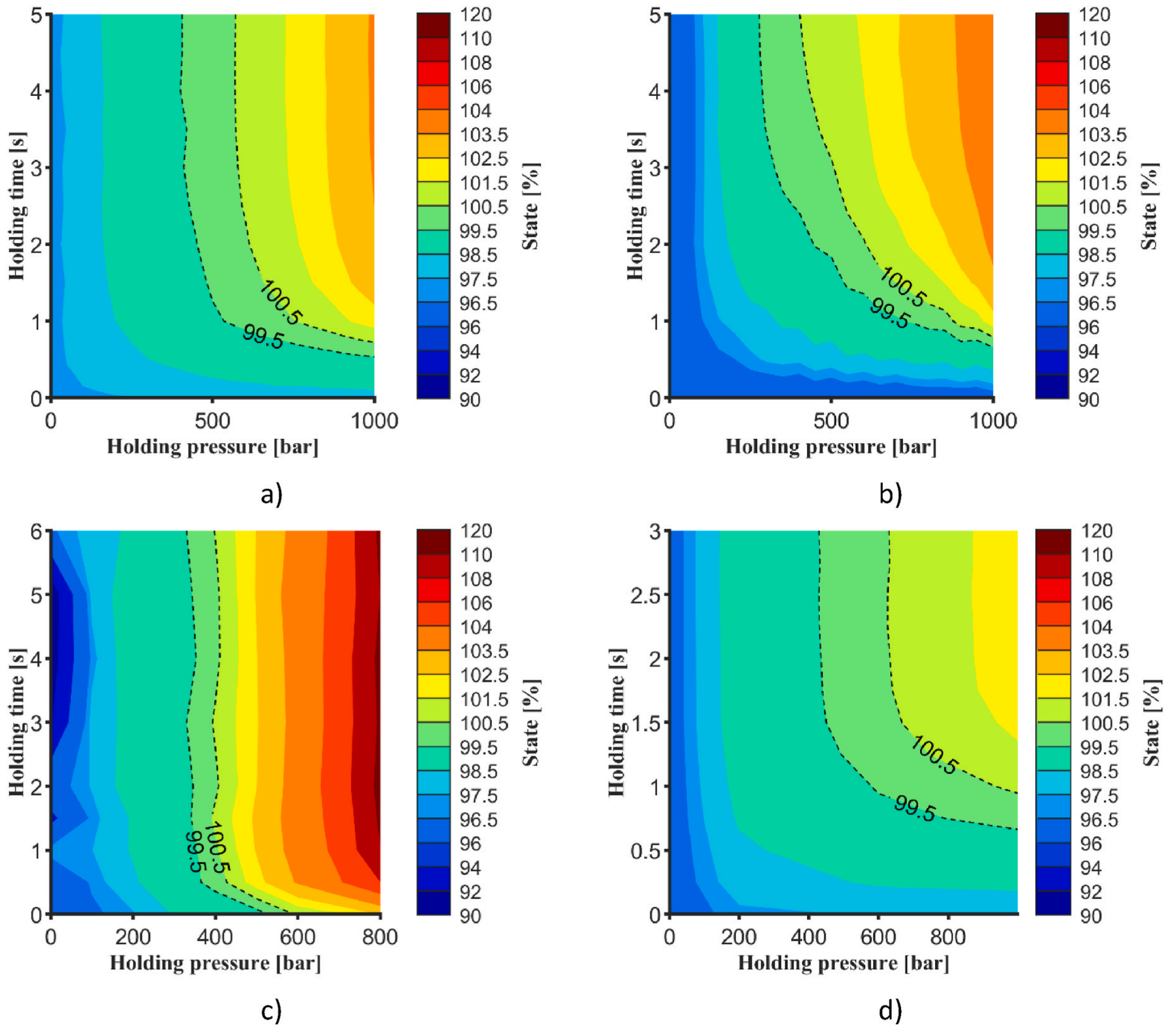

Fig. 12. The state space for holding phase optimization for different products a) lid from ABS b) lid from PLA c) plate 1 mm thick with a film gate d) small lid.

work well. Similarly, if you change the value of the actions that the algorithm can choose in transition from pre-learning to post-learning, the new actions will cause different effects, so the algorithm must learn how to use the new actions.

By increasing the value of the learning parameter (α), the values in each function weight ($\underline{w}$ and $\underline{\theta}$) can change more drastically. Therefore, if a learning parameter with a much larger value is used in post-learning than in pre-learning, the algorithm can easily favor suboptimal actions. In such cases, the algorithm often fails to find the best injection molding combination settings. Otherwise, if a learning parameter with a much smaller value is used in post-learning, the algorithm just slowly adapts to differences in the new environment. It is recommended using the same learning parameter for post-learning as for pre-learning.

3. Results

The presented experiment contains two main parts: filling and holding optimization. In each part, subparts were made based on the learning method, such as pre-learning and post-learning. Pre-learning is when the algorithm is taught for the first time to acquire prior knowledge, which the algorithm uses later in post-learning. Therefore, in pre-learning, the algorithm learns what to do in different states. Post-learning was used to check how well the algorithm can use the knowledge it learned when it was adjusting the injection molding settings for another product.

3.1. Filling phase optimization

The goal of filling phase optimization is to let the algorithm learn how to fill a cavity in a given time without overfilling it. Therefore, the algorithm uses the image of the product, the measured maximum pressure, and the measured filling time. During filling phase optimization, the algorithm can change the switch-over volume, which defines the end position of the screw during injection molding. Therefore, it influences the volume of the injected melt. In addition, the algorithm can also change the injection flow, which determines the speed of the screw. This setting allows the algorithm to indirectly alter the filling time of the cavity and the pressure in the melt.

3.1.1. Pre-learning for the filling phase

For the optimization of the filling phase the plate with a thickness of 1.2 mm and the fan gate design was used for pre-learning (Fig. 1 e). The starting injection molding settings were a switch-over volume of 22 cm^3 and an injection flow of 15 cm^3/s. With this setting combination, the cavity was not even close to being filled, and the flow of the melt was relatively slow. Therefore, the cavity will not be overfilled by accident, and the mold will not be damaged. The algorithm was able to change the injection flow by 0,±5,±10, or±15 cm^3/s and the switch-over volume by 0,±1,±2, or±5 cm^3. Between two learning steps, the algorithm will select one value combination for both settings (action) based on its policy. The end of learning was defined when the algorithm has made 4000 learning steps. The learning algorithm was able to reset the

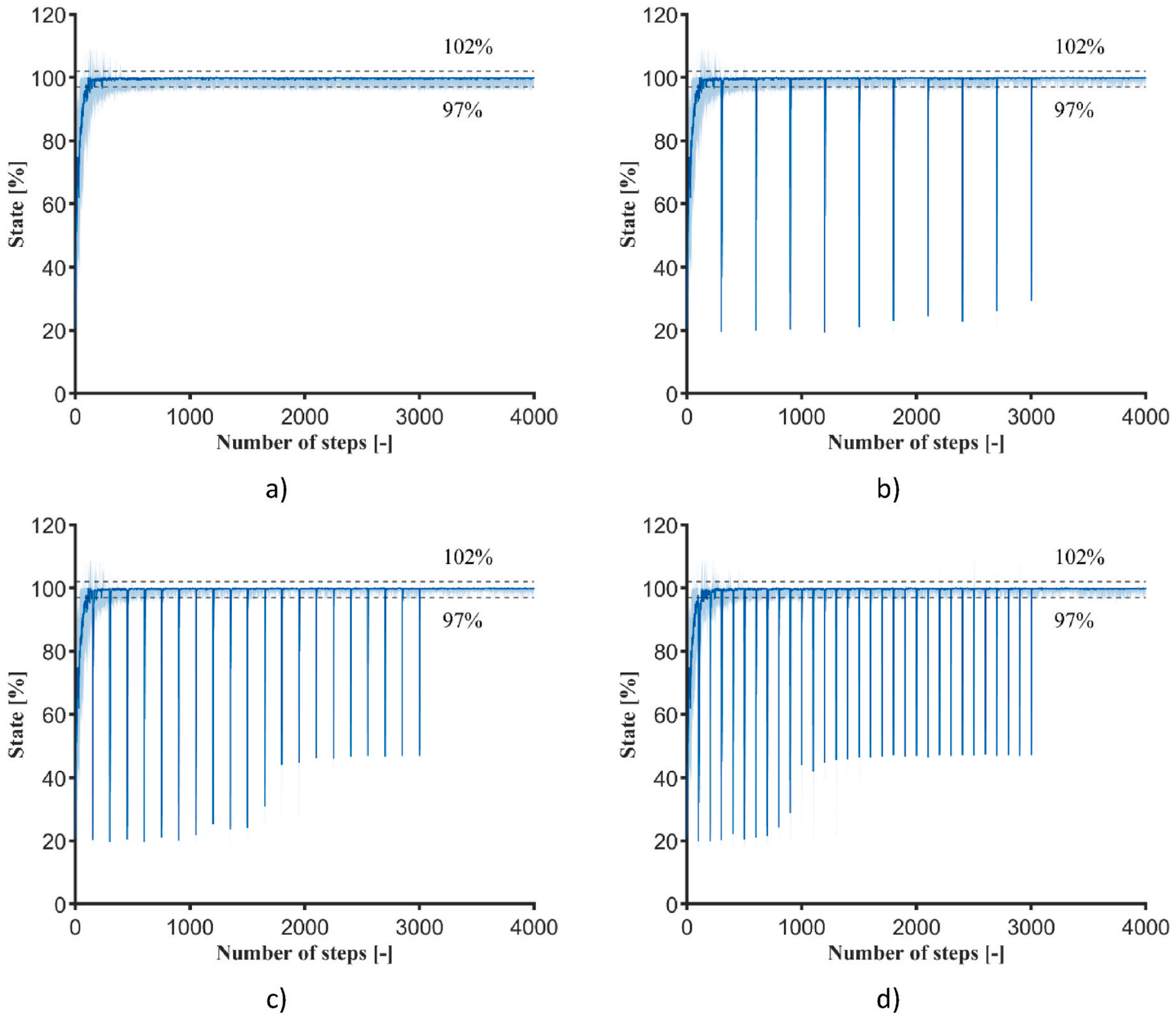

Fig. 13. Pre-learning from data of plate with 1.2 mm thickness and fan gate a) without restarting, b) with 10 restarts, c) with 20 restarts, d) with 30 restarts.

injection molding settings to the initial settings after a given step until it reached 3000 learning steps. Different learning scenarios were carried out with 0, 10, 20, or 30 restarts in 3000 steps (Fig. 13a–d respectively). It is clear that after a few hundred steps, the algorithm finds the optimal setting combinations because the state values go near 100 %. However, after the algorithm approaches the optimum, it will not really go back to a worse state, such as the initial state and its surroundings. On the one hand, this is good because the algorithm is not designed to look where it does not produce a good product. On the other hand, this way, the algorithm does not discover the optimal choices at the initial states. This is not the best choice if the algorithm starts from a similar state for another product. If the algorithm returned to the initial position in sufficient time, it would reach the target value much faster (see Fig. 13c and d). The knowledge that the algorithm gathered during pre-learning and used as prior knowledge was stored in the $\underline{w}$ and $\underline{\theta}$ weight vectors. These vectors were the main difference between pre-learning and post-learning. In pre-learning, the starting values of these weight vectors were the default (0.5 for each parameter in the vector). In post-learning, these weight vectors contained the mean values of the 100 pre-learning scenarios.

3.1.2. *The use of prior knowledge from a different gate type*

One key part of the design of the injection mold is the gate. During the filling phase, different gate designs mean, for example, different flow paths and shear stress in the melt. Therefore, knowing that the algorithm can be used for products with different gate designs is important. This implies that post-learning is performed on a product that differs from the pre-learning product only in the design of the gate (Fig. 14).

During pre-learning, it was clear that the algorithm needed a few hundred steps to reach the optimum state (see Fig. 13). This means exactly the same number of injection molding cycles, which takes a lot of time and produces many defective products. One can imagine pre-learning as someone trying to learn to use the injection molding machine. Post-learning, in contrast, is similar to an expert trying to adjust the machine settings. The results of post-learning from the different pre-learning settings are in Fig. 15. For the learning algorithm, post-learning took far fewer steps to get near the optimal state, and the starting slope of the states was steeper. This phenomenon can be seen in the plotted distributions, which show the number of injection molding cycles needed to produce a part with the required quality from the 100 scenarios. 75 % of the data was highlighted on the distribution with green areas and the upper quartile value of the distribution (Q3) was marked for clarity. The 75 % of the distribution was even narrower when restarting for pre-learning was applied. There was only a small difference between the performance of the different restart cases, so in subsequent analyses, the prior knowledge that was derived from pre-learning with 30 restarts was used. These results showed that if the algorithm learned the optimization process from a mold with a different gate design, it could find the optimum state in 22 injection molding cycles in 75 % of the cases. The distribution of the required cycle number was right-skewed, which means that the algorithm mostly found the optimal settings in a few cycles in most cases. This performance can be improved if the possible actions are changed or smaller requirements are used, and the performance could be close to that of a professional technician. These results show that the learning algorithm can be applied with prior knowledge to a similar product with a different gate design.

3.1.3. *The use of prior knowledge from a product produced from a different material*

In an industrial environment, producing a product from a different material may be necessary for sustainability, economic, or engineering reasons. In such cases, the optimal injection molding settings for the new material are usually not the same as the optimal settings for the previous material. Therefore, this may change the optimal injection molding settings because different materials can have different specific volumes, melt flow index, etc., so it is important to investigate whether the algorithm can use prior knowledge derived from another material to a new one for the filling phase. For this, the same plate was used with 1.2 mm thickness and a fan gate as for pre-learning, but this time, it was molded from PLA instead of ABS.

In order to show the effect of the material on post-learning, post-learning scenarios were performed with data from the original ABS material (Fig. 16 a) and with data from the new PLA material (Fig. 16 b). A comparison of the two post-learning processes shows that the case with ABS was better. However, in this case, the same data (i.e. product) was used for pre-learning and post-learning, so it was expected to have better performance. With this in mind, the performance of the PLA part was very good since the Q3 values in the two cases compared differed only by five injection molding cycles, even though a completely different

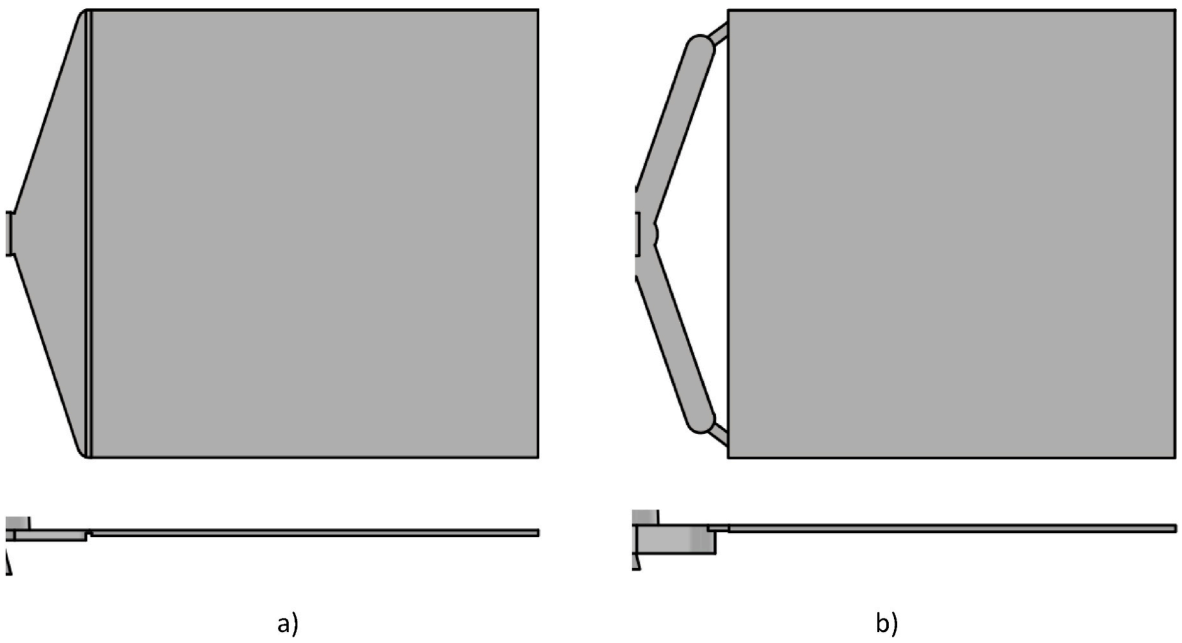

Fig. 14. Plate with 1.2 mm thickness a) with a fan gate design b) with a double gate design.

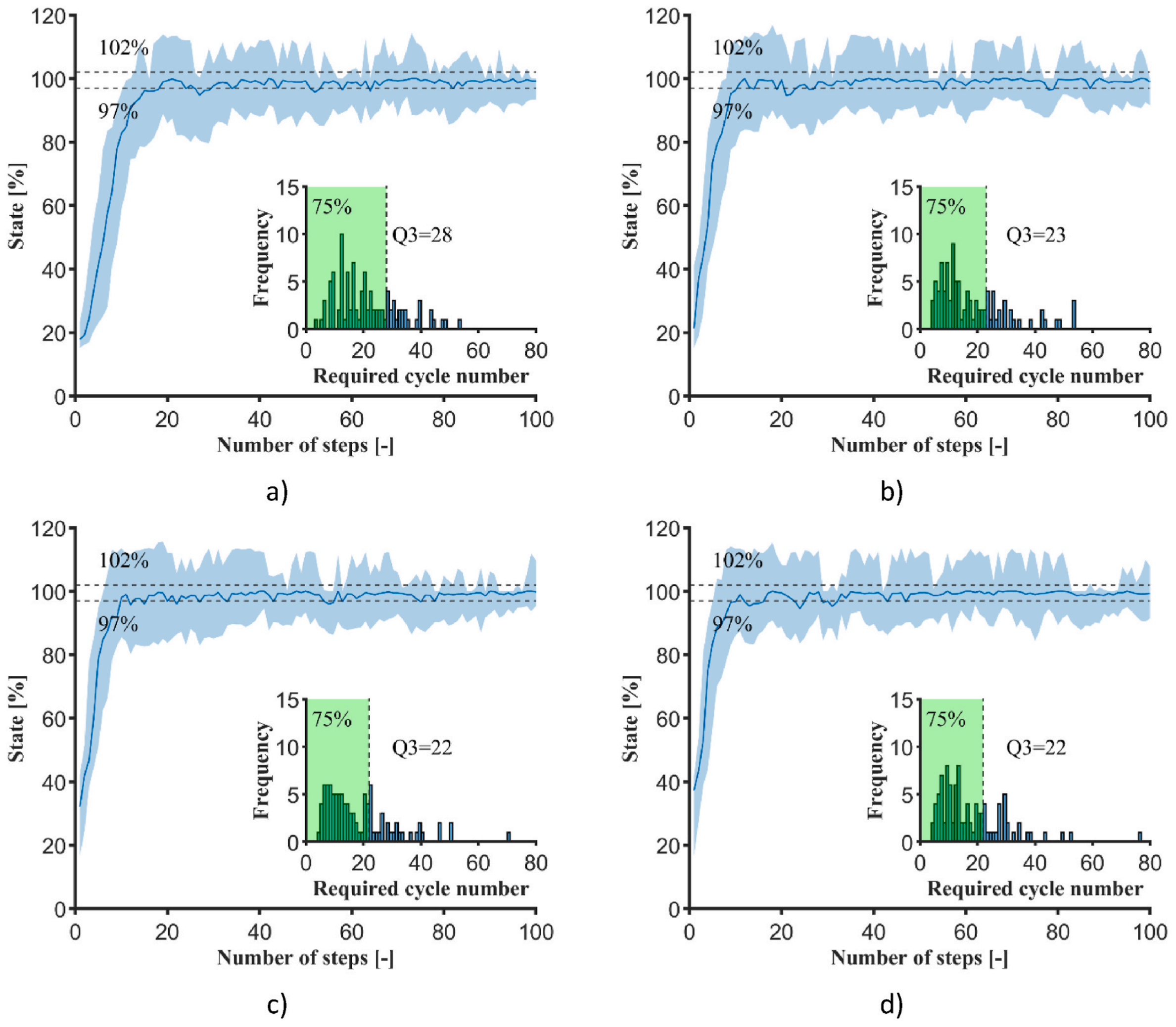

Fig. 15. Post-learning from data of the plate with 1.2 mm thickness and a double gate and the required cycles to reach the goal state a) without restarting, b) with 10 restarts, c) with 20 restarts, d) with 30 restarts.

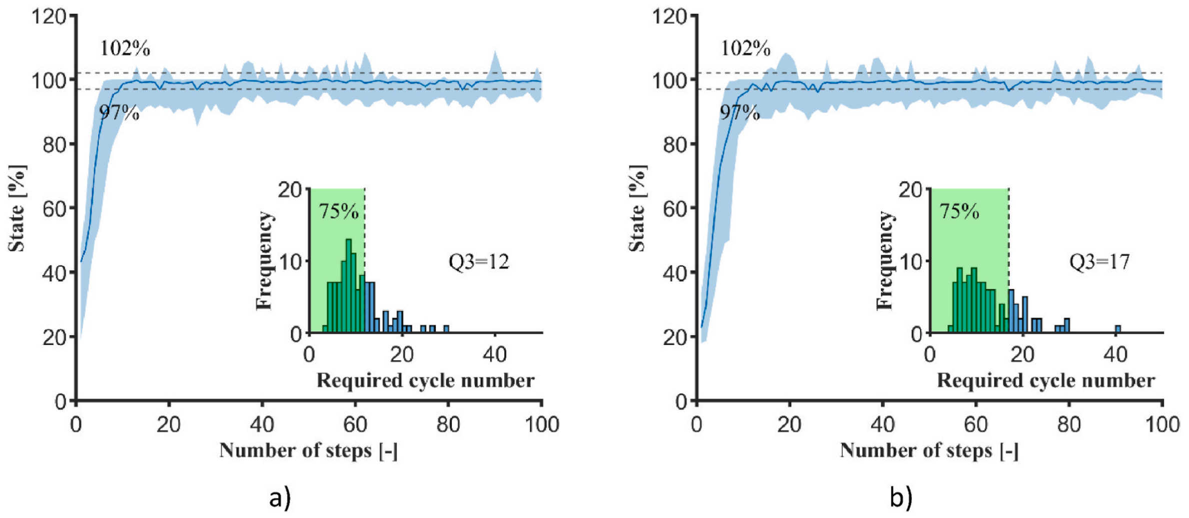

Fig. 16. Comparison of post-learning from different materials with 1.2 mm thick plate products a) product from ABS (same as pre-learning), b) product from PLA.

material was used in production. This result suggests that this method can be used with different materials.

3.1.4. The use of prior knowledge from a product with a different thickness

In practice, a company may produce similar products with different features such as thickness, nominal size, etc. Hence, it is also interesting whether the algorithm can use its prior knowledge for products with different thicknesses. For this purpose, plates with a thickness of 1 mm, 1.2 mm, and 2.5 mm (Fig. 17) were molded. The 1.2 mm thick plate was used for pre-learning, and then a thicker and thinner plate for post-learning. The thickness of the product is important in the filling phase because thinner products may freeze off if the injection flow is too low but need less melt to fill the cavity. Therefore, the injection molding settings can be very different for a product with a different thickness.

Fig. 18 shows the result of post-learning on 1 mm and 2.5 mm thick plates. Of the two cases, the optimization for the 1 mm plate was more efficient with the given prior knowledge. For the 2.5 mm thick plate, the search space (Fig. 11 d) changes much more when the part is overfilled. In contrast, the search space changes less while the part is underfilled. The specified pressure limit can explain the former phenomenon since this limit is 50 bar for the 2.5 mm plate (instead of 220 bar), which means a 5–10 bar change is significantly larger in proportion. The smaller change in the unfilled area is a consequence of the thickness of the cavity (and therefore, the product), as the same amount of melt will cause a smaller change during filling. These two effects make it harder for the algorithm to find the optimal state since the same actions will have a smaller effect when the part is underfilled and a larger effect when the part is overfilled. As Fig. 18 b shows, the algorithm starts from a much better state value than in the other cases. This is due to the fact that the algorithm changes only the injection flow and switch-over volume but not the prepared shot volume, which was set up on the injection molding machine. If the shot volume is larger, then the same switch-over will result in more melt in the cavity. This study regarded this part of the injection molding adjustment as the dosing phase, so the learning algorithm does not address it, nor does it address the optimization of the mold closing and opening phase. The only requirements are that the shot volume has to be greater than the possible switch-over volumes, and the shot volume used must be constant for the given learning process.

3.1.5. *The use of prior knowledge from a product with a different geometry for filling*

Changing the material for a given product is not uncommon in practice, but it is even more common for a company that wants to make a new product. In such cases, the results of injection molding simulations can help set the machine parameters. However, the software license is relatively expensive and using the software requires considerable expertise. Also, the simulations should usually be optimized based on real injection molding experiments. Consequently, such a process requires a lot of time, material, and money. In order to replace this process, the performance of the learning algorithm was investigated by using prior knowledge from the injection molding of a product with a given geometry for products with other geometries.

For these analysis, the lid product (Fig. 1 b) was used for post-learning because it contains many different features that the plate product does not. Therefore, it is sufficiently different from the original product. This difference is also clearly visible in the search space (Fig. 11a–f). The lid product also contains significantly thinner parts, which is why the in-mold pressure can rise much higher than in the case of the plate product. The effect of this phenomena can be seen in Fig. 19 when the state values increase above 150 %.

3.2. *Holding phase optimization*

The goal of holding phase optimization is to fine-tune the weight of the product. However, the holding phase has many general purposes, such as reducing shrinkage and sink marks. With the right measuring system and metric, the algorithm can be applied for these criteria similarly to the algorithm worked with product weight measurement. The holding phase comes after filling because the injected melt starts to cool and shrink. Therefore, additional melt is injected into the cavity at a given pressure (holding pressure) for a given time (holding time). Through this, the weight of the product changes but only slightly compared to the change in product weight during filling. During holding phase optimization, the algorithm can change holding pressure and holding time independently. There is a limit to the holding time for conventional cold runner molds, but one should not necessarily set a longer holding time than this limit. This limit is determined by the gate and the temperature conditions in the mold because the gate solidifies (freezes) after a given time. After this so-called freeze-off time, no more melt can be injected into the cavity. This is shown in Fig. 12; no significant change can be seen for a given holding pressure after a given holding time. Gate freeze-off time was not considered in the analysis, but injection molding scenarios were performed to get measurement points afterward.

3.2.1. *Pre-learning for the holding phase*

In this holding phase optimization, the ABS lid part (Fig. 1 b) was used for pre-learning. This product has a complex geometry with a simple point-like gate. The algorithm can change holding pressure with 0,±25,±50, or±100 bar and holding time with 0,±0.5,±1, or±2 s independently. Therefore, according to its policy, the algorithm will select one of the 49 (7 × 7) setting change combinations (49 actions). The end of learning was defined when the algorithm has made 6000 learning steps. During the analysis, more learning steps were used than during filling optimization because the algorithm needed more steps to learn the optimal policy. 100 learning scenarios were made with the selected algorithm settings similar to filling optimization and different learning scenarios with 0, 10, 20, or 30 restarts in 3000 steps (Fig. 20a–d respectively) were made. The initial injection molding settings are 0 bar holding pressure and 0 s holding time. These setting combinations for pre-learning were chosen for security reasons. Too much holding pressure can open up the mold, which can cause flash on the product and damage the mold. In contrast to filling phase optimization, it is not clear, whether increasing the number of restarts would significantly reduce the initial steps. This phenomenon is probably due to the non-linearity of the search space near the starting settings. However, at the end of learning, the algorithm approaches the goal quite well. Of course, 6000 learning

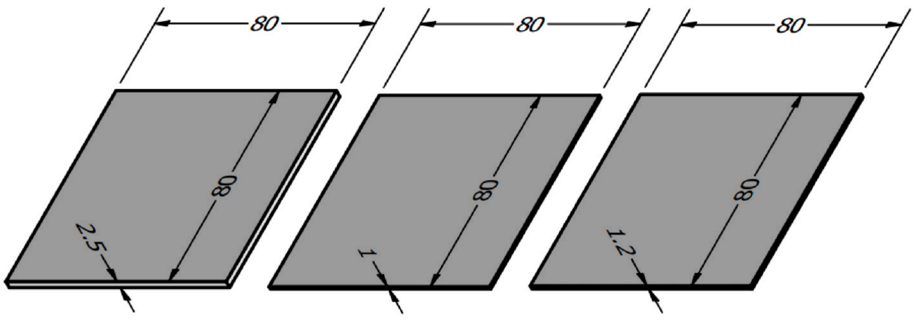

Fig. 17. The plates used with different part thicknesses.

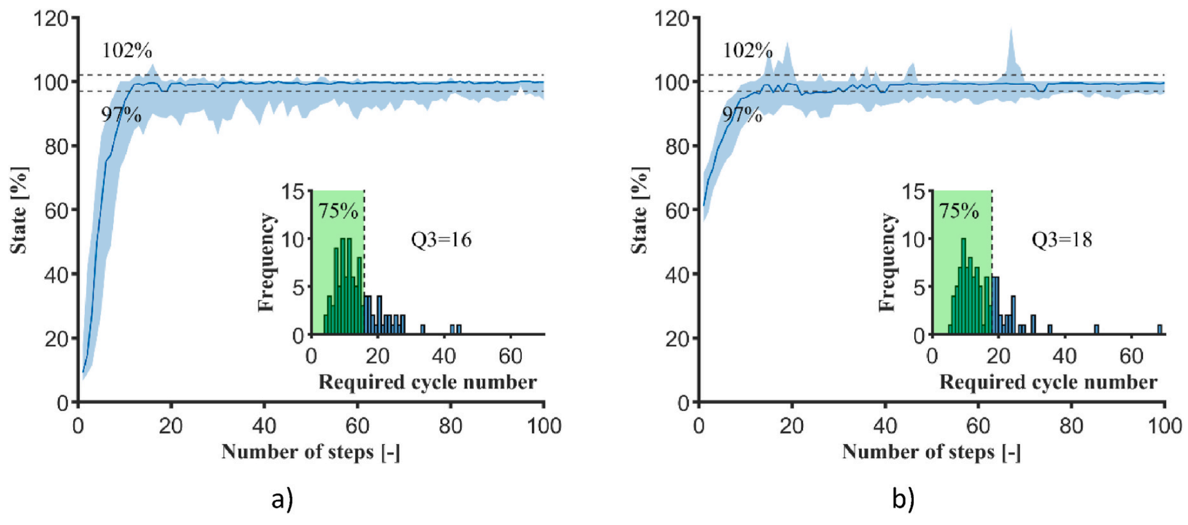

Fig. 18. Post-learning for the plate with a thickness of a) 1 mm and b) 2.5 mm (Pre-learning for the plate with a thickness of 1.2 mm).

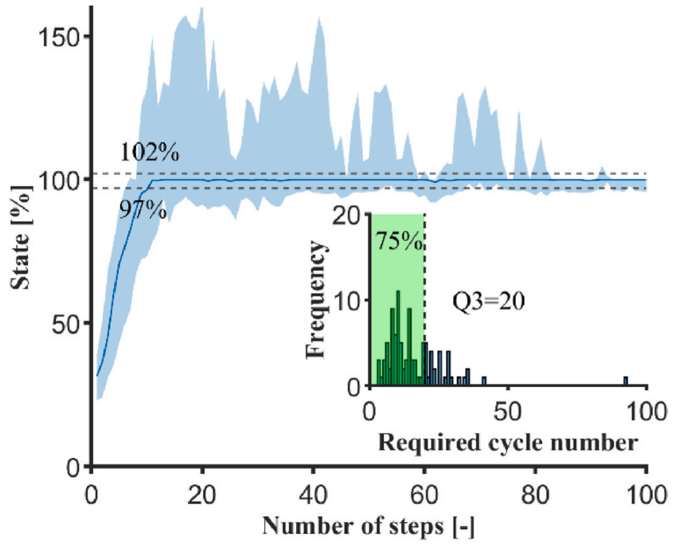

Fig. 19. Post-learning for the lid product produced from ABS (Pre-learning from the plate with a thickness of 1.2 mm).

steps is a lot if it would take the same number of injection molding cycles to learn the policy. Therefore, it is recommended to use a design of experiments to generate the data needed for pre-learning.

3.2.2. *The use of prior knowledge from a different material*

Earlier, the analysis for the post-learning for a new material (PLA) in filling optimization was presented (see 3.1.3). Of course, a change in the material also changes the optimal settings for the holding phase too. Different materials may require different processing temperatures (and consequently gate freeze-off times will be different), which can affect the required holding time. Besides, the shrinkage of the product and weight compensation can differ, too.

Fig. 21 a shows the result of post-learning with the new material. It takes far fewer learning steps to reach the optimal weight than during the pre-learning. In addition, Fig. 21 shows the distribution of how many learning steps (injection molding cycles) are needed for the algorithm to first produce a product with the required part weight in each of the 100 learning scenarios. In 75 % of the cases, it takes less than 20 learning steps for the algorithm to reach the goal. However, this would mean the production of scrap in 19 injection molding cycles, which is still not acceptable in some cases. It is clear from the initial stage of the reward function that the algorithm chooses actions that significantly reduce the error. Therefore, the main problem with learning is probably not its decision-making but that the initial injection molding settings are far from the settings producing the target. Since part of the learning task is to find these injection molding settings, the optimal settings found during pre-learning could be used as initial injection molding settings for post-learning. However, pre-learning consists of 100 different learning scenarios, each with 6000 learning steps. Therefore, for each of the 100 learning scenarios, all the injection molding settings were retrieved from the last 500 learning steps and chose the most frequent of these as the initial injection molding settings of post-learning. The results of post-learning with the new initial settings shows the advantage in Fig. 21 b compared to Fig. 21 a. From the distribution shown here, it is clear that in 75 % of the cases, fewer than four injection molding setting changes are sufficient for the algorithm to make a product of the desired quality. The major drawback of this solution is that the optimum injection molding settings of the product used for pre-learning may differ significantly from the optimum of the product used for post-learning. If this is the case, this method is only useful if the new initial settings are closer to the target settings than the default ones. In conclusion, this study showed that prior knowledge can be effectively applied to the algorithm for holding phase optimization to produce a product with the same nominal geometry using a new material.

3.2.3. *The use of prior knowledge from a different geometry*

The possibility of using prior knowledge from different products during filling has already been shown (see Fig. 19). However, it is also important to investigate whether prior knowledge from other geometries can be used to optimize the holding phase. The amount of compensation may vary from one product to another due to product volume or features on the product (ribs, tubes, thickness changes). In addition, the size and design of the gate may also vary for different products, and thus, the optimum of the holding phase may differ. Therefore, the effect of prior knowledge on holding phase optimization with products of different shapes, sizes, and gate geometries was investigated. The learning algorithm trained itself on the lid product (pre-learning). After that, it used the plate product with 1 mm thickness and the small lid products for post-learning (Fig. 22).

The plate product has a film gate, unlike the lid or small lid products, which have a point-like gate. The difference between the gate designs can cause a variation in the effect of holding time. The small lid is part of a 16-cavity mold, so holding pressure and holding time affect 16 products simultaneously. Besides, the small lid is significantly smaller and thinner. Therefore, the two chosen products significantly differ from the

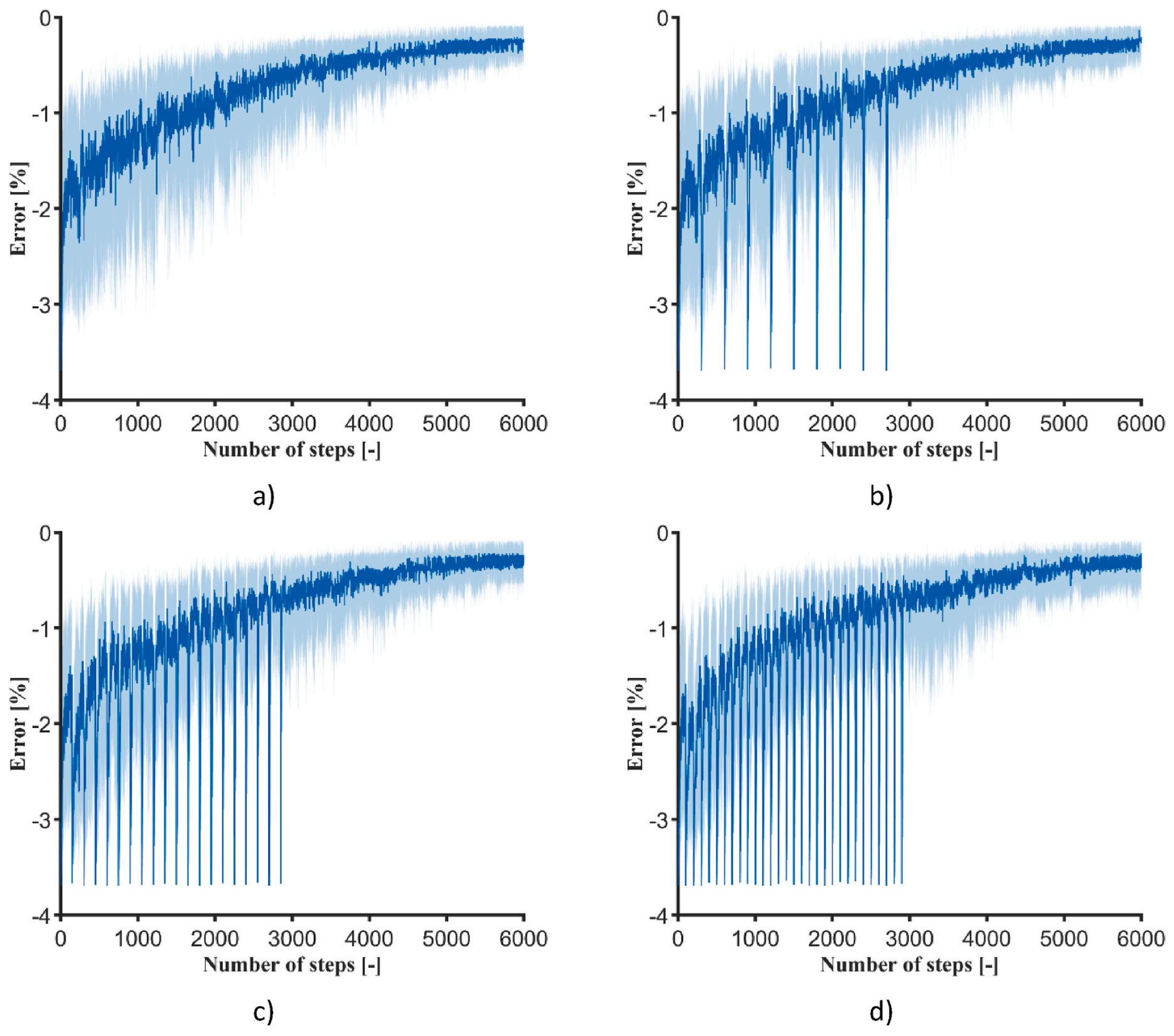

Fig. 20. Pre-learning from the data of lid product a) without restarting, b) with 10 restarts, c) with 20 restarts, d) with 30 restarts.

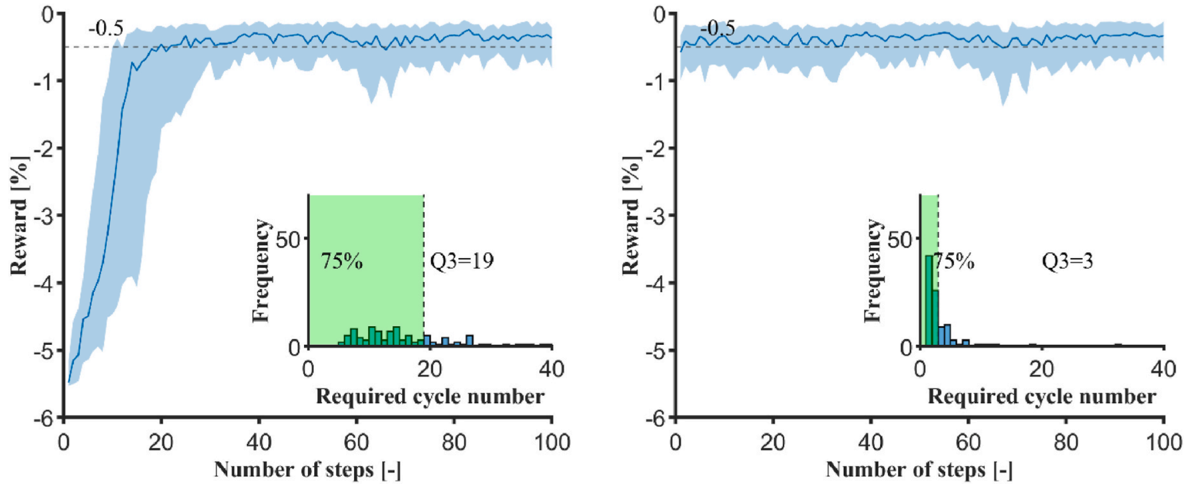

Fig. 21. Post-learning on lid data with PLA (Pre-learning from lid data with ABS) a) with the initial starting point, b) with a starting point defined from pre-learning.

lid. The results of post-learning are shown in Fig. 23. The results of post-learning with the initial injection molding settings (Fig. 23 a and c) show that within 20 (for the plate) or 16 (for the small lid) injection molding cycles, the algorithm reaches its goal in 75 % of the cases. This is not a bad performance for completely new products, but it is important to note that this is only holding phase optimization. Therefore, learning scenarios were performed with the starting point defined from pre-learning (Fig. 23 b and d). These results show that the initial injection molding settings can significantly reduce the number of cycles needed.

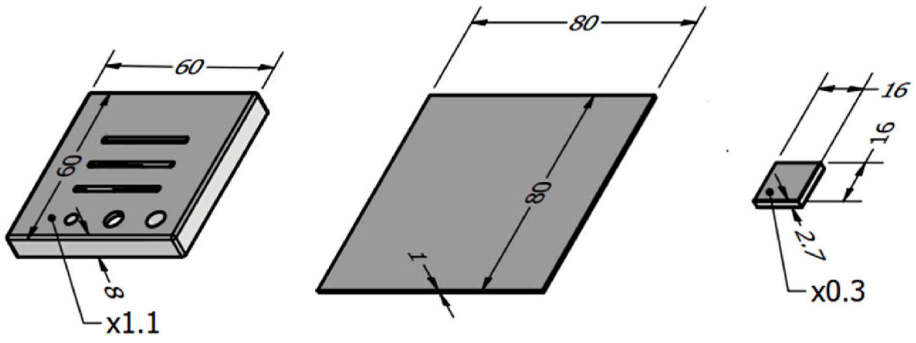

Fig. 22. The products used for holding phase optimization lid (left), plate (middle), small lid (right).

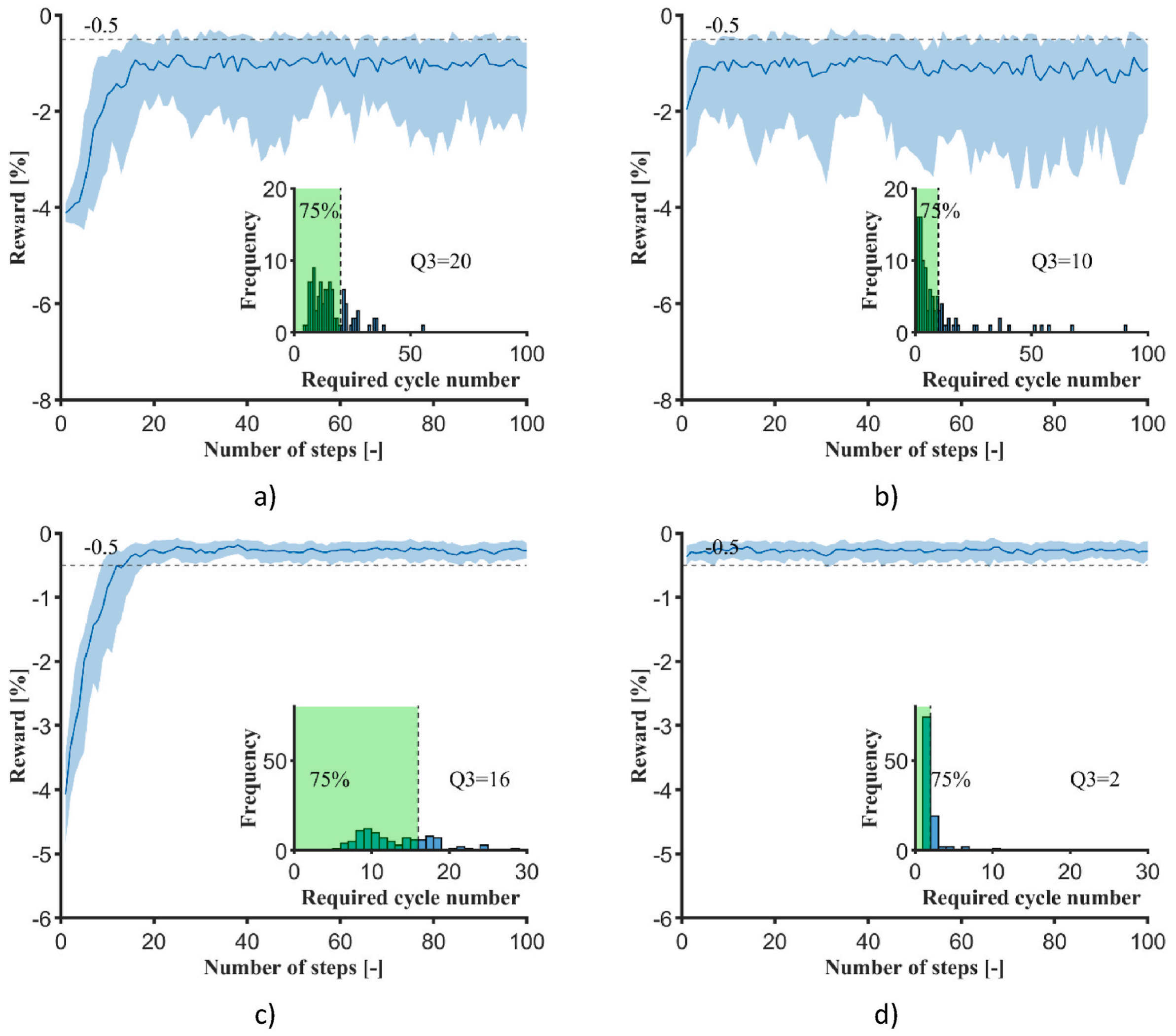

Fig. 23. a) Post-learning for the plate product with the default starting settings b) post-learning for the plate product with a starting point defined from pre-learning c) post-learning for the small lid product with the default starting settings d) post-learning for the small lid product with a starting point defined from pre-learning.

3.3. The performance of the algorithm for optimization

Previously, it was presented how prior knowledge can be used for the learning algorithm in different cases. During the analysis of post-learning results, it was examined how many injection molding cycles were necessary for the algorithm to make a good product for the first time in the new environment. During the analysis, the performance of the learning algorithm was investigated with 100 learning scenarios for each product combination. From these 100 learning cases, a distribution from the required number of learning steps (injection molding cycles) of the algorithm was made to produce a product with the needed quality. In this chapter, a quick comparison is shown between the different post-learning results and the results of different distributions for better interpretation.

3.3.1. Comparison of the effect of different prior knowledge for filling phase optimization

For the filling phase optimization, prior knowledge was provided by pre-learning on a plate product with 1.2 mm thickness and a fan gate design (Fig. 1 e). The performance of the learning algorithm was examined when it used prior knowledge in the production of the same product or in the production of parts with different geometries and material. The different metrics of the distribution derived from the learning scenarios can be seen in Table 5.

The results show that the performance of the algorithm was the best when the same product was used for pre- and post-learning. This is not surprising since prior knowledge contained information related to the optimal decisions for the given product. The second best case was when the geometry of the product was not changed between the two learning processes, only the material. In such a case, the difference occurring during the filling phase depends mostly on the rheological properties of the materials. The learning that took the longest (93 cycles) was when the algorithm had to use prior knowledge for a completely new geometry (lid product). In this case, the in-mold pressure, the required filling time and the different features of the product (ribs, tubes) may have caused large differences in the effect of individual injection molding parameters. It is also important to note that this value of 93 cycles is an outlier in the distribution (Fig. 19) and therefore does not represent the differences in learning well by itself. The mean or median values of the distributions show that there are no large differences between the individual use of prior knowledge.

3.3.2. Comparison of the effect of different prior knowledge for holding phase optimization

Prior knowledge was provided for the holding phase optimization by pre-learning the ABS lid product (Fig. 1 b). In this optimization, two methods can be distinguished: post-learning starting from 0 bar holding pressure, 0 s holding time, and learning with a starting point defined from pre-learning.

In the former case (Table 6), it is clearly visible that post-learning was the fastest in the case of the small lid product. This is not so surprising as it is in this search space that the widest combination of settings can be used to produce a good product. This is due to the fact that the injection molding machine fills 16 cavities with the same geometry at the same time, so the effect of the melt is divided between these cavities.

Similar results can be seen in the case when the starting point was determined from pre-learning (Table 7). The number of cycles here is significantly smaller since the algorithm started the exploration closer to the optimal settings. However, it is important that the starting point determined in this way is not always closer to the optimum in a new environment. Even with this starting point, there is a learning case that requires a large number of cycles (91 cycles). This may be due to the fact that the range of optimal settings for the plate product (Fig. 12 c) is narrower compared to the one in pre-learning (Fig. 12 a), so the algorithm probably skips this range during exploration. It is clear from the other metrics of the distribution that this is not the typical required cycle number here either; it is also just an outlier.

4. Conclusion

This study shows that using prior knowledge to optimize the filling and holding phases for injection molding is worthwhile. However, the presented method only works for an actor-critic algorithm with state aggregation and discrete actions. The learning method consists of two main steps: a pre-learning step, during which the algorithm learns how to set up the injection molding machine, and a post-learning step, during which the algorithm uses so-called prior knowledge (from the pre-learning step) to set up the machine for a new product.

A method based on pressure measurement and image processing was presented for filling phase optimization. In the analysis, it was shown that the algorithm can learn to set the injection molding machine from products with different gate designs, materials, part thickness, or geometry. With the use of prior knowledge, the learning algorithm can set up the injection molding machine to produce parts with the required quality despite these changes (gate, material, etc.). This study concludes that for the learning algorithm, the changes in geometry are more challenging to handle than changes in material for filling phase optimization (Table 5). However, the method can be applied with good performance in all cases, and the use of this method is recommended in industrial environments.

Moreover, the presented method is able to fine-tune product weight by optimizing the holding phase. The results show that the algorithm can be used for products with different types of gates, part geometry, and materials. In this investigation, two different settings were used for the learning algorithm: one with the original starting point (with 0 bar holding pressure and 0 s holding time) and one with a starting point derived from pre-learning. The former settings were the original settings for safety reasons (these results can be seen in Table 6). However, it was also investigated how much it can help the optimization process if the starting point is derived from pre-learning (it can also be found as prior knowledge). These results (Table 7) showed that the starting point had a significant effect on learning, and the performance of this method is comparable to that of a good technician.

Funding

This work was supported by the National Research, Development and Innovation Office, Hungary (2020–1.2.3-EUREKA-2021-00010, 2019–1.1.1-PIACI-KFI-2019-00205). This research was funded by the Horizon Europe Framework Programme and the call HORIZON-WIDERA-2021-ACCESS-03, under the grant agreement for project 101079051 – IPPT_TWINN. The research was done under the scope of the Project no. RRF-2.3.1-21-2022-00009, entitled “National Laboratory for Renewable Energy” which has been implemented with the support provided by the Recovery and Resilience Facility of the European Union within the framework of Programme Széchenyi Plan Plus. Project no. TKP-6-6/PALY-2021 has been implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the TKP2021-NVA funding scheme.

Table 5
The summary of the post-learning results for filling phase optimization.

Pre-learning data	Post-learning data	Distribution of cycle number until the first acceptable product is made					
		Minimum	Q1 (best 25 % of the cases)	Mean	Median	Q3 (best 75 % of the cases)	Maximum
1.2 mm thick plate (fan gate) from ABS	1.2 mm thick plate (fan gate) from ABS	3	7	10.11	9	12	30
	1.2 mm thick plate (double gate)	4	9	17.21	13	22	77
	1.2 mm thick plate (fan gate) from PLA	4	8	12.46	11	17	41
	1 mm thick plate (film gate)	4	9	13.55	12	16	45
	2.5 mm thick plate (film gate)	5	9	14.98	13	18	69
	Lid (ABS)	3	9	15.68	13.5	20	93

Table 6
The summary of the post-learning results for holding phase optimization with the default starting point.

Pre-learning data	Post-learning data	Distribution of cycle number until the first acceptable product is made					
		Minimum	Q1 (best 25 % of the cases)	Mean	Median	Q3 (best 75 % of the cases)	Maximum
Lid from ABS	Lid from ABS	5	9	13.33	12.5	17	26
	Lid from PLA	5	10	15.73	14	19	44
	1 mm thick plate with a film gate	4	8	15.31	13.5	20	56
	Small lid	5	9	12.67	11.5	16	29

Table 7
The summary of the post-learning results for holding phase optimization with a starting point derived from pre-learning.

Pre-learning data	Post-learning data	Distribution of cycle number until the first acceptable product is made					
		Minimum	Q1 (best 25 % of the cases)	Mean	Median	Q3 (best 75 % of the cases)	Maximum
Lid from ABS	Lid from PLA	1	1	2.96	2	3	33
	1 mm thick plate with film gate	1	2	9.79	4	10	91
	Small lid	1	1	1.49	1	2	11

Ethical approval

This article does not contain any studies with human participants or animals performed by any of the authors.

CRediT authorship contribution statement

Richárd Dominik Párizs: Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Dániel Török:** Writing – review & editing, Supervision, Resources, Project administration, Funding acquisition.

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.

Data availability

Data will be made available on request.

Acknowledgments

We wish to thank Arburg Hungária Kft. for the ARBURG Allrounder 470 A 1000–290 injection molding machine, and Tool-Temp Hungária Kft., Lenzkes Gmbh and Piovan Hungary Kft. for the accessories. The authors would also like to thank Attila Balaskó for his help in checking the grammar of the article.

References

[1] Z. Chen, L.S. Turng, A review of current developments in process and quality control for injection molding, Adv. Polym. Technol.: Journal of the Polymer Processing Institute 24 (3) (2005) 165–182, https://doi.org/10.1002/adv.20046.
[2] L. Gajzlerova, J. Navratilova, M. Polášková, L. Beníček, R. Čermák, The polymorphic composition of long-chain branched polypropylene processed by injection and compression molding, Express Polym. Lett. 17 (10) (2023) 1031–1041, https://doi.org/10.3144/expresspolymlett.2023.77.
[3] A. Mourya, A. Nanda, K. Parashar, R. Kumar, An explanatory study on defects in plastic molding parts caused by machine parameters in injection molding process, Mater. Today: Proc. 78 (2023) 656–661, https://doi.org/10.1016/j.matpr.2022.12.070.
[4] A.L. Kelly, M. Woodhead, P.D. Coates, Comparison of injection molding machine performance, Polym. Eng. Sci. 45 (6) (2005) 857–865, https://doi.org/10.1002/pen.20335.
[5] J. Gomez-Caturla, N. Montanes, L. Quiles-Carrillo, R. Balart, D. Garcia-Garcia, F. Dominici, D. Puglia, L. Torre, Development of biodegradable PLA composites and tangerine peel flour with improved toughness containing a natural-based terpenoid, Express Polym. Lett. 17 (8) (2023) 789–805, https://doi.org/10.3144/expresspolymlett.2023.59.
[6] S. Horváth, J.G. Kovács, Effect of processing parameters and wall thickness on the strength of injection molded products, Period. Polytech. - Mech. Eng. 68 (1) (2024) 78–84, https://doi.org/10.3311/PPme.24068.
[7] S. Krizsma, A. Suplicz, Comprehensive in-mould state monitoring of material jetting additively manufactured and machined aluminium injection moulds, J. Manuf. Process. 84 (2022) 1298–1309, https://doi.org/10.1016/j.jmapro.2022.10.070.
[8] M. Moayyedian, K. Abhary, R. Marian, Gate design and filling process analysis of the cavity in injection molding process, Advances in Manufacturing 4 (2016) 123–133, https://doi.org/10.1007/s40436-016-0138-5.
[9] M. Myers, R. Mulyana, J.M. Castro, B. Hoffman, Experimental Development of an injection molding process window, Polymers 15 (15) (2023) 3207, https://doi.org/10.3390/polym15153207.
[10] M. Czepiel, M. Bańkosz, A. Sobczak-Kupiec, Advanced injection molding methods, Materials 16 (17) (2023) 5802, https://doi.org/10.3390/ma16175802.
[11] U. Roy, B. Zhu, Development of material information model for the injection molding process and product, in: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, 2014 V004T06A058, https://doi.org/10.1115/DETC2014-35200, 46353.
[12] S.M. Mukras, H.M. Omar, F.A. al-Mufadi, Experimental-based multi-objective optimization of injection molding process parameters, Arabian J. Sci. Eng. 44 (2019) 7653–7665, https://doi.org/10.1007/s13369-019-03855-1.
[13] U. Roy, Y. Li, Sustainability assessment of the injection molding process and the effects of material selection, in: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, 2014 V003T06A059, https://doi.org/10.1115/DETC2014-35205, 46353.
[14] P.H.A. Chen, M.G. Villarreal-Marroquín, A.M. Dean, T.J. Santner, R. Mulyana, J. M. Castro, Sequential design of an injection molding process using a calibrated predictor, J. Qual. Technol. 50 (3) (2018) 309–326, https://doi.org/10.1080/00224065.2018.1474696.
[15] M.A. Barghash, F.A. Alkaabneh, Shrinkage and warpage detailed analysis and optimization for the injection molding process using multistage experimental design, Qual. Eng. 26 (3) (2014) 319–334, https://doi.org/10.1080/08982112.2013.852679.
[16] S. Kashyap, D. Datta, Process parameter optimization of plastic injection molding: a review, International Journal of Platics Technology 19 (1) (2015) 1–18, https://doi.org/10.1007/s12588-015-9115-2.
[17] A. Benayad, R. El Otmani, A. El Hakimi, M. Boutaous, A. Touache, K.R. Musa, S. Derdouri, N. Mahfoudi, D. Siginer, Experimental investigation and numerical simulation of the microinjection molding process through an expanding flow configuration, Polym. Adv. Technol. 32 (4) (2021) 1690–1711, https://doi.org/10.1002/pat.5206.
[18] M.W. Wang, F. Arifin, V.H. Vu, The study of optimal molding of a LED lens with grey relational analysis and molding simulation, Period. Polytech. - Mech. Eng. 63 (4) (2019) 278–294, https://doi.org/10.3311/PPme.13337.
[19] S. Krizsma, A. Suplicz, Monitoring and modelling the deformation of an aluminium prototype mould insert under different injection moulding and clamping conditions, Results in Engineering 20 (2023) 101556, https://doi.org/10.1016/j.rineng.2023.101556.
[20] M. Baum, D. Anders, A numerical simulation study of mold filling in the injection molding process, Computer Methods in Materials Science 21 (1) (2021) 25–34, https://doi.org/10.7494/cmms.2021.1.0743.
[21] A.O. Andrisano, F. Gherardini, F. Leali, M. Pellicciari, A. Vergnano, Design of simulation experiments method for injection molding process optimization, in: IMProVe 2011-International Conference on Innovative Methods in Product Design-Proceedings, 2011, pp. 476–486. https://hdl.handle.net/11380/655838.
[22] S. Hwang, J. Kim, Injection mold design of reverse engineering using injection molding analysis and machine learning, J. Mech. Sci. Technol. 33 (8) (2019) 3803–3812, https://doi.org/10.1007/s12206-019-0723-1.

[23] J. Gim, B. Rhee, Novel analysis methodology of cavity pressure profiles in injection-molding processes using interpretation of machine learning model, Polymers 13 (19) (2021) 3297, https://doi.org/10.3390/polym13193297.
[24] Z. Gao, G. Dong, Y. Tang, Y.F. Zhao, Machine learning aided design of conformal cooling channels for injection molding, J. Intell. Manuf. 34 (3) (2023) 1183–1201, https://doi.org/10.1007/s10845-021-01841-9.
[25] O. Ogorodnyk, O.V. Lyngstad, M. Larsen, K. Wang, K. Martinsen, Application of machine learning methods for prediction of parts quality in thermoplastics injection molding, Advanced Manufacturing and Automation VIII 484 (2019) 237–244, https://doi.org/10.1007/978-981-13-2375-1_30.
[26] Ardestani A. Mollaei, G. Azamirad, Y. Shokrollahi, M. Calaon, J.H. Hattel, M. Kulahci, R. Soltani, G. Tosello, Application of machine learning for prediction and process optimization-case study of blush defect in plastic injection molding, Appl. Sci. 13 (4) (2023) 2617, https://doi.org/10.3390/app13042617.
[27] F.Y. Wu, J. Yin, S.C. Chen, X.Q. Gao, L. Zhou, Y. Lu, J. Lei, G.J. Zhong, Z.M. Li, Application of machine learning to reveal relationship between processing-structure-property for polypropylene injection molding, Polymer 269 (2023) 125736, https://doi.org/10.1016/j.polymer.2023.125736.
[28] H. Cañas, J. Mula, M. Díaz-Madroñero, F. Campuzano-Bolarín, Implementing industry 4.0 principles, Comput. Ind. Eng. 158 (2021) 107379, https://doi.org/10.1016/j.cie.2021.107379.
[29] M.R. Khosravani, S. Nasiri, T. Reinicke, Intelligent knowledge-based system to improve injection molding process, Journal of industrial information Integration 25 (2022) 100275, https://doi.org/10.1016/j.jii.2021.100275.
[30] V. Rousopoulou, A. Nizamis, T. Vafeiadis, D. Ioannidis, D. Tzovaras, Predictive maintenance for injection molding machines enabled by cognitive analytics for industry 4.0, Frontiers in Artificial Intelligence 3 (2020) 578152, https://doi.org/10.3389/frai.2020.578152.
[31] S. Farahani, N. Brown, J. Loftis, C. Krick, F. Pichl, R. Vaculik, S. Pilla, Evaluation of in-mold sensors and machine data towards enhancing product quality and process monitoring via Industry 4.0, The International Journal of Advanced Manufactuing Technology 105 (2019) 1371–1389, https://doi.org/10.1007/s00170-019-04323-8.
[32] A. Benešová, J. Tupa, Requirements for education and qualification of people in Industry 4.0, Procedia Manuf. 11 (2017) 2195–2202, https://doi.org/10.1016/j.promfg.2017.07.366.
[33] C. Combemale, K.S. Whitefoot, L. Ales, E.R. Fuchs, Not all technological change is equal: how the separability of tasks mediates the effect of technology change on skill demand, Ind. Corp. Change 30 (6) (2021) 1361–1387, https://doi.org/10.1093/icc/dtab026.
[34] A. de Giorgio, A. Maffei, M. Onori, L. Wang, Towards online reinforced learning of assembly sequence planning with interactive guidance systems for industry 4.0 adaptive manufacturing, J. Manuf. Syst. 60 (2021) 22–34, https://doi.org/10.1016/j.jmsy.2021.05.001.
[35] P. Garrad, S. Unnikrishnan, Reinforcement learning in VANET penetration testing, Results in Engineering 17 (2023) 100970, https://doi.org/10.1016/j.rineng.2023.100970.
[36] F. Guo, X. Zhou, J. Liu, Y. Zhang, D. Li, H. Zhou, A reinforcement learning decision model for online process parameters optimization from offline data in injection molding, Appl. Soft Comput. 85 (2019) 105828, https://doi.org/10.1016/j.asoc.2019.105828.
[37] S. Lee, Y. Cho, Y.H. Lee, Injection mold production sustainable scheduling using deep reinforcement learning, Sustainability 12 (20) (2020) 8718, https://doi.org/10.3390/su12208718.
[38] Y. Qin, C. Zhao, F. Gao, An intelligent non-optimality self-recovery method based on reinforcement learning with small data in big data era, Chemometr. Intell. Lab. Syst. 176 (2018) 89–100, https://doi.org/10.1016/j.chemolab.2018.03.010.
[39] H.I. Ugurlu, S. Kalkan, A. Saranli, Reinforcement learning versus conventional control for controlling a planar bi-rotor platform with tail appendage, J. Intell. Rob. Syst. 102 (2021) 1–17, https://doi.org/10.1007/s10846-021-01412-3.
[40] J.E. Sierra-Garcia, M. Santos, Combining reinforcement learning and conventional control to improve automatic guided vehicles tracking of complex trajectories, Expet Syst. 41 (2) (2024) e13076, https://doi.org/10.1111/exsy.13076.
[41] Z. Wang, T. Hong, Reinforcement learning for building controls: the opportunities and challenges, Appl. Energy 269 (2020) 115036, https://doi.org/10.1016/j.apenergy.2020.115036.

Richárd Dominik Párizs obtained his B.Sc. and M.Sc. degree in Mechanical Engineering at the Budapest University of Technology and Economics (Hungary) in 2018 and 2020, respectively. He is currently a Ph.D. student at the Department of Polymer Engineering, Budapest University of Technology and Economics. His research interest is the use of machine learning for injection molding technology, mainly supervised learning and reinforcement learning.

Dániel Török is an assistant professor at the Department of Polymer Engineering, Faculty of Mechanical Engineering, Budapest University of Technology and Economics (Hungary), and a research fellow at MTA-BME Lendület Lightweight Polymer Composites Research Group, Hungarian Academy of Sciences (Hungary). His research interests include injection molding, polymer testing, polymer materials science, machine learning and image processing.