IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 14 June 2024, accepted 4 July 2024, date of publication 8 July 2024, date of current version 16 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3424664

RESEARCH ARTICLE

Adaptive Visual Quality Inspection Based on Defect Prediction From Production Parameters

ZVEZDAN LONČAREVIĆ[1], SIMON REBERŠEK[1], SAMO ŠELA[2], JURE SKVARČ[2], ALEŠ UDE[1], (Member, IEEE), AND ANDREJ GAMS[1], (Member, IEEE)

[1]Department of Automatics, Biocybernetics and Robotics, Jožef Stefan Institute, 1000 Ljubljana, Slovenia
[2]SICK d.o.o, 1000 Ljubljana, Slovenia

Corresponding author: Zvezdan Lončarević (zvezdan.loncarevic@ijs.si)

This work was supported in part by the DIGITOP Project funded by the Ministry of Higher Education, Science and Innovation of Slovenia, Slovenian Research and Innovation Agency and European Union—NextGenerationEU; and in part by the Slovenian Research Agency through the program group "Automation, Robotics and Biocybernetics" under Grant P2-0076.

ABSTRACT At the end of a production process, the manufactured products must usually be visually inspected to ensure their quality. Often, it is necessary to inspect the final product from several viewpoints. However, the inspection of all possible aspects might take too long and thus create a bottleneck in the production process. In this paper we propose and evaluate a methodology for adaptive, robot-aided visual quality inspection. With the proposed method, the most probable defects are first predicted based on the production process parameters. A suitable classifier for defect prediction is learnt in an unsupervised manner from a database that includes the produced parts and the associated parameters. A robot then steers the camera only towards viewpoints associated with predicted defects, which implies that the trajectories of robot motion for the inspection might be different for every product. To enable dynamic planning of camera trajectories, we describe a methodology for evaluation and selection of the most appropriate autonomous motion planner. The proposed defect prediction approach was compared to other methods and evaluated on the products from a real-world production line for injection moulding, which was implemented for a producer of parts in the automotive industry.

INDEX TERMS Industrial informatics, injection moulding, machine learning, production parameters, quality inspection, robot motion planning.

I. INTRODUCTION

The required increase in the quality of products, services and processes of the Industry 4.0 paradigms [1] gave rise to the Zero Defects Manufacturing (ZDM) approach [2]. *Per se*, ZDM does not explicitly rely on defect and fault detection, but rather on defect and fault prediction and provision of suggestions on how those can be avoided [3]. However, quality inspection remains an integral step of production processes, sometimes in the form of selective inspections based on the analysis of impact of the action on the overall economic, production logistics and quality performance [4].

In a multi-stage production process, the quality of produced parts is often checked during or after every stage [5]. For example, in production of injection-moulded parts that have added motor windings, which is also the main use-case for this paper, the product gets inspected after the insertion of side windings, after injection moulding, after cooling, after additional machining of parts, etc. In this example, each stage of the production is followed by an intermediate quality inspection process, based on different physical quantities. However, even if the part is deemed good after each intermediate step, the final part, when inspected by a human worker, is sometimes still classified as bad, because defects such as, e.g., porous material, are difficult to detect [6]. Thus, one of the solutions to prevent faulty

The associate editor coordinating the review of this manuscript and approving it for publication was Huiyan Zhang.

products, besides sampling of products and inspecting them manually, is to visually inspect the parts also in the final stage of the production [7].

Visual inspection of all possible aspects of a complex product, however, requires the observation of the part from many different viewpoints. As stated in [8], this process can be automated in two different manners. The first option is to install several cameras so that all the required aspects of the object are observable at the same time. However, the cameras, especially if there are a lot of them, could get in the way of the production process or other means of quality control. In our practical example with an automotive parts producer, inspection with dial indicators was already integrated into the production line and prevented installation of additional cameras.

The second option is to move the camera around the object with a robot using an in-hand camera to sequentially collect the required images [9]. The caveat of the solution with the moving camera is that the process might take more time and thus generate a bottleneck in the production process. A conceptually similar solution of moving the object around the camera is also a possibility [8], but often not a viable one because the object is partially occluded when grasped by the robot.

Machine learning approaches are increasingly being applied to improve the reliability and robustness of quality control and even production processes themselves [10], including injection moulding [11]. They can detect anomalies that are difficult or even impossible to identify for a production-line engineer. In this paper, we advance the application of machine learning approaches to deal with such challenges.

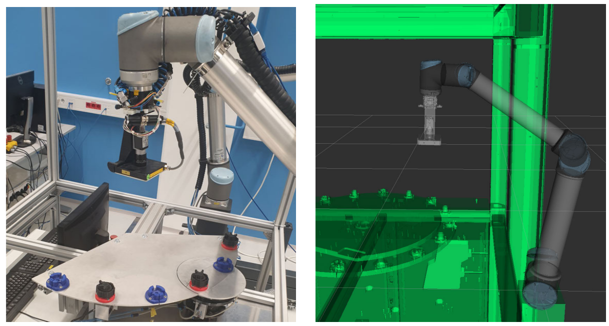
A. PROBLEM FORMULATION

The problem can be formulated as follows. A visual quality inspection with an in-hand camera needs to be added to an existing production line, and it should not create a bottleneck. This implies that only the most likely defective aspects of the object should be inspected. However, such aspects need to be first determined, and later the robot needs to move the camera so that they can be observed. This further implies that the motion of the robot might be different for every workpiece, and thus the motion of the camera needs to be generated dynamically as a part of the inspection cycle.

B. CONTRIBUTION

The contribution of this paper is a methodology for adaptive robot-aided visual quality inspection, where not all aspects of the product but only the most likely predicted areas with defects are inspected. To achieve this goal, we predict the possible defects using a database of the available production parameters and perform visual quality inspection only from viewpoints that provide clear images of the possible defects.

The methodology for the prediction of defects is based on a database of production parameters. Starting with the analysis of several classification methods, we propose an approach based on clustering of production parameters projected into a latent space of a deep autoencoder, which is trained with the parameters of all available good and faulty parts. The latent space projections of faulty parts are clustered based on different types of defects, which can be checked from specific viewpoints. For each new part, the system checks the posterior probability of its latent space projection belonging to the clusters, and the robot then steers the camera to the appropriate viewpoint with sufficiently large posterior probabilities in descending order to check the area where the predicted defect is located.



(a) (b)

FIGURE 1. Physical implementation of the visual quality inspection cell (a) and its visualization in RViz (b).

Because the camera path is not determined in advance but dynamically defined by the sequence of potential viewpoints, the robot motion cannot be pre-programmed. We therefore rely on a motion planner to generate the required motion trajectories. While many robot motion planners exist, we outline and evaluate a methodology to test and select the best motion planner using the planner arena motion planning evaluation tool [12]. In our experiments, motion planners available in the Open Motion Planning Library (OMPL) [13] were evaluated.

All aspects of the proposed adaptive visual quality approach, the prediction of defects and the planning of robot camera steering trajectories were evaluated on data from a real-world injection-moulding production of a parts producer in the automotive industry. A simulated image of the experimental environment and its physical twin are depicted in Fig. 1.

II. RELATED WORK

This paper deals with the application of machine learning methods for quality prediction, robot-aided visual quality inspection, and robot motion planning. The related works in each of these areas are discussed separately.

A. MACHINE LEARNING METHODS FOR QUALITY PREDICTION

Machine learning methods have been applied in different manners for quality prediction [10]. In [14], the authors show that four examined machine learning (ML) algorithms adequately predict quality in injection moulding even with very little training data. Pressure data from the mould cavity was used for quality prediction. A similar analysis was performed in [15], where tree-based algorithms, regression based algorithms, and autoencoder were compared. Using a plethora of collected data, the paper shows that autoencoder outperforms other examined tree-based machine learning algorithms in accuracy, precision, recall, and F1-score. Artificial neural networks (ANN) were compared to decision trees in [16] for prediction of parts quality in thermoplastics injection moulding, showing over 99% accuracy rates. On the other hand, Silva et al. [17] show that only a combination of ANN and support vector machines (SVM) allowed them to reach accuracy above 99%. That work was extended to identify the relevant parameters [18]. Since many injection moulding machines are not prepared to access their data on a real-time basis, digitalization of such devices was explored in [19].

Given that in the real-world the datasets are often uneven or imbalanced, different methods of dealing with such datasets have been proposed [20]. Supervised methods are known to need large quantities of data [21]. Undersampling of the majority class [22] or oversampling of the minority class with repeated or synthetic data are often used [23]. Both can lead to overfitting or loss of characteristics. On the other hand, unsupervised classification methods can be applied for uneven databases [24]. These include, among others, clustering and autoencoders. The latter can be used to detect difference through the reconstruction error [25]. Clustering and deep autoencoders can be combined with deep embedded clustering (DEC) [26].

In this paper we compare several supervised classifiers and evaluate our proposed unsupervised approach based on clustering in the latent space of a deep AE.

B. ROBOT-AIDED VISUAL QUALITY INSPECTION

Robot-aided visual quality inspection has been applied in different manners, either by moving the camera around the object, or by moving the object to be inspected in front of the camera [8]. In both cases, achieving a proper spatial relation between the camera and the object is critical for obtaining high quality images for subsequent processing and detection of defects, for example by using advanced deep learning methods [27], [28]. Thus, camera location [29], [30], including a robot-supported autofocus mechanism [31], and camera motion [9] are often the subject of optimization for quality inspection. Even so, the space in production lines is usually rather confined due to the production and inspection equipment and the robot may not be able to achieve optimal viewpoints for all the aspects of the object that need to be inspected. Thus, robot in-hand camera motion is sometimes combined with a rotation of the object to be inspected [8]. In this paper we use an in-hand camera that is moved around the product, but the motion of the robot is not predefined – it is planned on-line using a motion planner.

C. ROBOT MOTION PLANNING

In recent years, considerable progress was made to realize fast and robust algorithms for robot motion planning [32]. Their goal is to enable robots to automatically compute their motions from descriptors of tasks and models acquired from sensors [33]. As per the definition, the task of a path-planner is to "find a collision-free motion between an initial (start) and a final configuration (goal) within a specified environment" [34]. Various software packages offer readily available planning algorithms, one notable example being MoveIt! [35]. There exist several planning libraries that offer different types of planners. For instance, the Open Motion Planning Library (OMPL) [13] provides geometric and control-based planners and also includes algorithms for constrained planning [36]. In addition, alternatives such as Stochastic Trajectory Optimization for Motion Planning (STOMP) [37], Gradient Optimization Techniques for Efficient Motion Planning (CHOMP) [38], and Search-based Planning Library (SBPL) [39] are available.

Tools are available within the Robot Operating System (ROS) framework to evaluate the performance of different motion planners. This includes the aforementioned MoveIt!, RViz for visualization, and Planner arena [12] for benchmarking of different planners. Open Motion Planning Library (OMPL) [13] is an integral part of available motion planning tools. We used it in the context of visual quality inspection to find planners that can generate collision-free motion trajectories in a fast and reliable manner when steering the camera to the desired viewpoints.

III. METHODOLOGY

In this section we provide the methodology of the proposed defect prediction algorithm.

The proposed defect prediction approach is based on the assumptions that 1. similar production parameters produce a similar output and 2. there is a subset of most common defects, which cover the large majority of all cases. Therefore, we use the production parameters from previously produced parts to predict the outcome of the production process for each new produced part, i.e., we compute the most probable defects based on its production parameters.

As discussed in Section II, there exist different classification approaches. The most important factors in deciding which of these classifiers can be applied effectively are the type and amount of production parameters gathered in the training database. In this section, we outline our unsupervised approach that relies on clustering of faulty parts in the latent space of a deep autoencoder (AE), which is trained on all available production data. In Section IV we show why other methods, such as for example a random forest classifier [40] or a deep neural network classifier with or without a softmax layer, cannot be effectively applied in our case.

In the proposed approach, we first train a deep AE using the complete database of production parameters. We then project the production parameters of parts onto a low dimensional latent space of a deep AE. The autoencoder passes its input

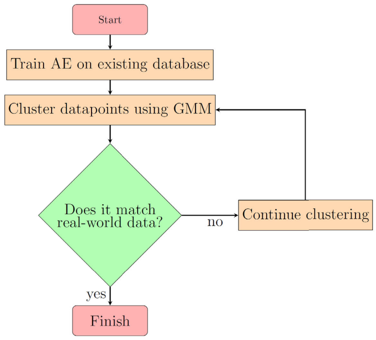


FIGURE 2. Diagram of data clustering.

data to the output so that the output matches the input with the highest possible precision. The data are pushed through differently sized layers, including through the lowest-dimensional layer, known as bottleneck or latent space. The projected data in the latent space now encodes the data with a small number of parameters. To discover the set of most common defects, the production parameters of available faulty parts are projected onto this latent space and clustered using GMM clustering. It is important to note that the real-world data is used to match the clusters and the defects. This process is illustrated in Fig. 2. Each defect type is associated with a camera viewpoint or a set of viewpoints from where its occurrence can be inspected.

At production time, the production parameters of each new part are projected onto the AE latent space. The posterior probabilities of Gaussian mixture components are then computed to predict the most probable defects. They determine the sequence of viewpoints, where the robot should place the camera to check for the most probable defects, as detailed in Algorithm 1. Note that only potential defects where the posterior probability is larger than a cut-off posterior probability p_c, determined empirically based on the producer's requirements, are checked. Thus, time is saved by performing less quality checks, testing only the defects that are likely to occur.

IV. EVALUATION OF CLASSIFICATION APPROACHES

In the following section, we introduce a functional database containing production parameters derived from the injection-moulding process used in manufacturing car parts. We then discuss and analyze design choices and the effectiveness of the proposed approach by utilizing the data obtained from this production process.

A. PRODUCTION PARAMETERS IN INJECTION MOULDING

During the production process, hundreds of production parameters are stored for every product (part). However, a large portion of the parameters is not relevant for the process itself (server name, user name, folder, ...). In collaboration with a production engineer working at the manufacturer, we identified 22 relevant production parameters. They are listed in Table 1. Some can easily be understood, while

Algorithm 1 Procedure for Inspecting the Viewpoints of One Product Based on the Process Parameters

procedure Check viewpoints
  project new part θ (Table 1) into the latent space θ^{AE} of a previously trained AE;
  compute posterior probabilities $\mathbf{P}$ of the established c GMM clusters associated to c most common defects;
  arrange $\mathbf{P}$ in descending order;
  set $i = 1$
  for probabilities $p \in \mathbf{P}$:
    if $p > p_c$:
      choose viewpoint(s) $\mathbf{V}_i$ corresponding to p_i;
      plan trajectory τ between current robot posture and $\mathbf{V}_i$;
      execute trajectory τ and set $i = i + 1$;
      if vision check (Fig. 11) returns error:
        discard product;
        break
  end

TABLE 1. Injection moulding parameters θ used for prediction of defects.

cycle time	switch after volume
injection time	machine cycle counter
max injection pressure	feeder housing temperature
switch after pressure	tool 1 heating circuit
dozing time	tool 2 heating circuit
material cushion	tool 4 heating circuit
tool 7 heating circuit	blasting nest
tool 10 heating circuit	nest insertion
tool 11 heating circuit	nest withdraw
cooling nest	PD_DIA minimum value
marking nest	PD_DIA average value

the others, such as PD_DIA average and minimum values, were proposed by the engineer. Normalized data from the production process are, without identifiers, made available at https://github.com/abr-ijs/production_parameters.

B. DATABASE

For evaluation we gathered a database that consists of production parameters for 4436 good (OK) parts and 45 faulty (NOK) parts. The production parameters for good parts were collected in 5 days. We also had 32 faulty physical parts with associated parameters available. The faulty parts were collected by workers at the production line by manually inspecting samples of products over the course of three months. We refer to this complete database as DB1.

The main reason for many more entries for good parts is that a typical industrial production process produces significantly more good than bad parts. In addition, the parts that reach the end of the production line have already passed intermediate checks, and thus the number of bad parts is relatively small [10]. In the use-case of this paper, the parts that are deemed faulty at the intermediate checks are discarded and their production data is not stored or is incomplete. In any case, the data about faulty parts identified

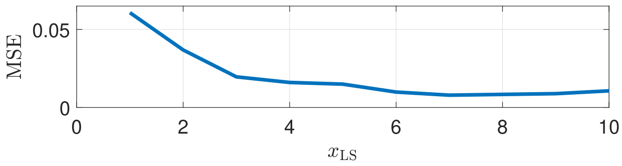


FIGURE 3. Mean Squared Error (MSE) of samples after pushing the data through AEs with different latent space dimensions.

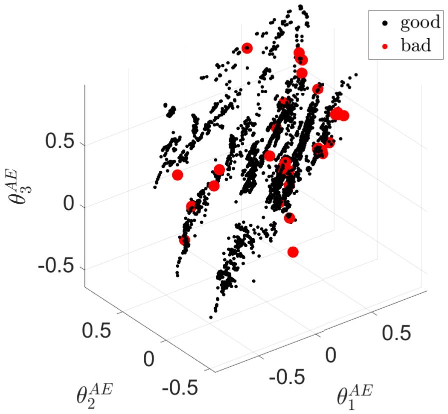


FIGURE 4. Database DB1 datapoints projected into the 3-dimensional latent space of a deep AE with red dots marking datapoints for faulty parts and smaller black dots for good parts.

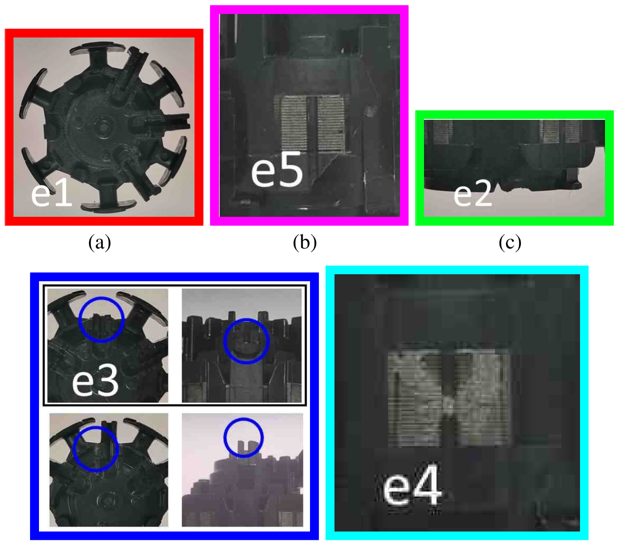
at the end of the production line can initially only be collected by assigning a human worker with sufficient knowledge [41] to manually inspect the parts. Thus, in general, the data for faulty parts is difficult to obtain.

To obtain the production parameters for each part, their QR code identifier is scanned and used to extract the data from the complete production line database. In our practical example, the discrepancy between the number of faulty physical parts and the number of data for faulty parts in database DB1 is due to some parts having the (small) QR code identifier not readable.

C. DEFECT PROBABILITY PREDICTION USING DEEP AE AND CLUSTERING

As explained in Algorithm 1, we pass all the parameters gathered in DB1 through a deep autoencoder (AE). The projected data is then normalized for each parameter. The autoencoder was designed empirically, with a 22 dimensional input and output layers and 5 hidden layers of the size 15, 10, x_{LS}, 10, 15, respectively. The hyperbolic tangent sigmoid activation function was used for the neurons and mean squared error (MSE) for the cost function. We first checked the size of the latent space that still preserves most of the information. As shown in Fig. 3, the mean squared error remains consistent after $x_{LS=3}$. An illustrative example projections of parameters of parts onto the latent space of a deep AE for DB1 is depicted in Fig. 4.

From the manufacturer we know that there are 5 typical errors that occur with the considered parts. These include porous material, faulty/broken sides, incomplete bottom edge, fault/broken pins and faulty windings as a result of



(a) (b) (c) (d) (e)

FIGURE 5. Five typical defects: (a) porous material — e1, (b) faulty/broken sides – e5, (c) incomplete bottom edge –e2, (d) faulty/broken pins – e3, and (e) faulty windings as a result of the injection moulding process –e4.

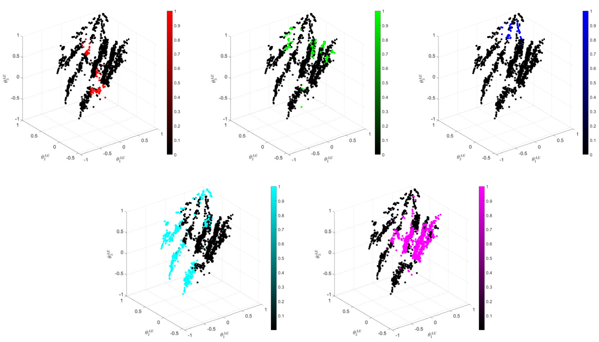


FIGURE 6. Probability of defect determined by posterior distribution of GMM components. The colors of the highest defect probability are associated with defects in Fig. 5.

the injection moulding process. The defects are depicted in Fig. 5. We therefore partitioned the data into $c = 5$ clusters. In order to exclude the effect of the selected clustering method on the result of clustering, we tested two methods; k-means and Gaussian Mixture Model (GMM) clustering, with 5 target clusters for each. The former, which is the most commonly used clustering approach, forms spheres around the centers of the clusters and performs hard classification, while the latter can handle non-round shapes of clusters and perform soft classification [42]. To check if the clustering depends on the autoencoder, we also trained the deep AE with different random initial parameters of the deep AE network. The results of clustering using both methods in latent spaces of 6 different AEs are for illustration shown in Fig. 7. Clusters are marked with different colors. Except for one object, both algorithms clustered the data in the same manner in all shown cases. This shows that training of deep AE with different initial parameters will change the latent space, but not necessarily the clustering. Due to its soft classification

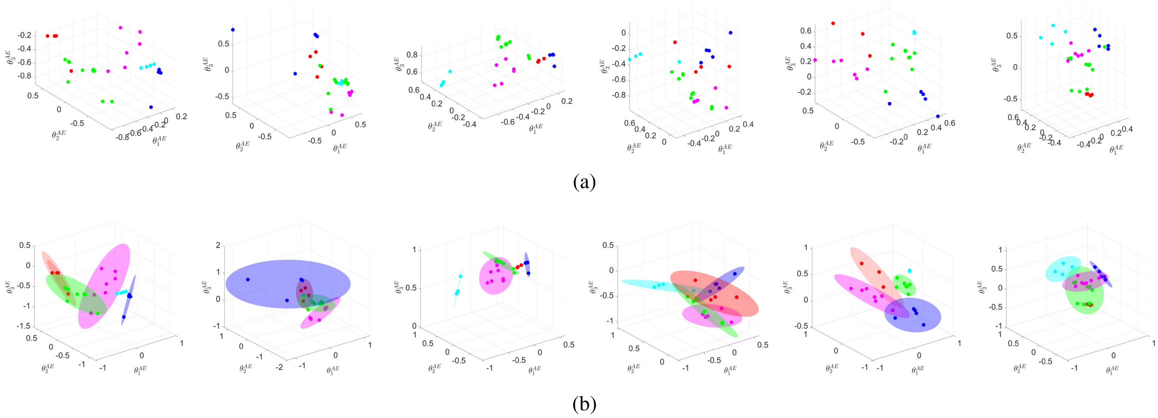


FIGURE 7. Clustering of faulty datapoints in the latent space of six autoencoders trained with different randomly selected initial autoencoder parameters. (a): k-means, (b): GMM clustering. In all cases, the clustering and color coding is the same as in Fig. 5.

FIGURE 8. The defects on physical parts coincide with the clustering of the associated production parameters in the latent space of the autoencoder, highlighted with the same colors in Fig. 7.

outcome, we utilized the GMM clustering method in our methodology.

Thus, for any new data point, we check to which cluster it belongs using the posterior probability for all GMM components in the latent space. Results for the posterior probability of all DB1 data points are shown in Fig. 6.

Note, that clustering starts with random initial seeds. As depicted in Fig. 2, to use the clusters for classification of real data, they have to match. By associating the latent space data points with the physical parts with known defects, shown in Fig. 8, we can see that only one part fits into two clusters (green and purple, shown in Fig. 8 in the 5-th row left). This result confirms that the clustering procedure is appropriate.

D. COMPARISON OF CLASSIFICATION WITH DIFFERENT CLASSIFIERS

As stated in Section II, there exist various classifiers that can be used to classify the data. In this section we compare the classification results of our proposed approach based on clustering in the latent space of the AE with three different supervised classifier methods on the parts with defects from DB1: simple neural network classification, softmax neural network classification and random forest classification. For each of these three methods, we used 70% of the data points of parts with defects for training, 15% for validation and 15% for testing. We additionally tested another unsupervised method, i.e., classification based on AE reconstruction error.

1) DEEP NEURAL NETWORK CLASSIFICATION (DNN)

We defined a deep neural network with the input and output normalized in range [−1,1], with the sizes of hidden layers [15 10 6 3]. When performing the forward pass, the denormalized output was rounded to the closest number in range [1, 5], with the rounded output corresponding to the identity of the defect.

2) SOFTMAX DEEP NEURAL NETWORK CLASSIFICATION (DNN+SOFTMAX)

We used the same deep neural network but added a softmax layer at the end.

3) RANDOM FOREST CLASSIFICATION (RF)

We used 50 estimators that describe how many trees are trained. The minimal number of leaf node observations was set to 1.

4) AUTOENCODER RECONSTRUCTION ERROR (AE-RE)

The error of AE reconstruction can be used, e.g., see [25]. For this case, we trained the autoencoder on the good parts and

TABLE 2. Average classification and clustering results for 100 attempts using different classifiers and the proposed AE LS GMM clustering algorithm.

Classifier	DNN	DNN+Softmax	RF
Accuracy (%)	0.29167	0.353	0.579
Classifier	AE-re	**AE LS GMM**	
Accuracy (%)	0.5545	**0.969**	

then checked the reconstruction of good and bad parts, when passed through the AE. We clustered the reconstruction error to check if it can match with the real data.

5) GMM CLUSTERING IN THE LATENT SPACE OF AN AE (AE LS GMM)

We clustered the projection in the latent space to match with the real data We also tested whether clustering the data points in the latent space of an AE is consistent. To do this, we first obtained the clusters that match the real-world data and then re-clustered the data by changing the initial center of the clusters.

Note, as mentioned above, the choice of classifier highly depends on the available database and potentially, with a more even database, supervised classifier would do much better. Please see also the discussion in Section VII.

Results in Table 2 show that, as expected, the three supervised classifiers do not perform well with a low number of data samples because each data point that is left out for validation makes a large difference. Using all the data, however, would lead to over-fitting. Unsupervised anomaly detection performed better. AR-re could reliably predict that there was a defect for 78.44% of cases, but could not be used to classify the defect, as the defect classification was only 55.45% accurate. GMM clustering in the latent space of an AE (AE LS GMM), on the other hand, led to better classification rates, where 31 out of 32 items are correctly classified (96.9%), and was more consistent with the real-world data, as changing the cluster centers still matched the data in 85% of the time.

Our approach proceeds with checking the parts in the descending order of probabilities of possible defects. This means that the motion of the robot is potentially different for each part that needs to be inspected. We therefore generate the robot motions that place the camera at the required viewpoints online using a suitable motion planner.

V. ADAPTIVE VISUAL QUALITY INSPECTION

The production example presented in the introduction calls for the inspection of parts from 10 different viewpoints, even though there are only 5 most common defects. This is because some of the defects have to be inspected from more than one viewpoint (e.g. pins, porous material). In addition, there is the starting point of the robot. Thus, the robot potentially needs to move between 11 different postures. The physical implementation and simulation of the real visual quality inspection cell are depicted in Fig. 1.

A. INSPECTION VIEWPOINTS

The task of the production engineer is to associate the potential defects with the predefined camera viewpoints (**V**). The viewpoints must be selected in such a way that the critical surface area of the part is visible and that the acquired images are sharp. Additionally, the engineer should make sure that the associated robot postures are reachable in the confined space of the workcell. Some of the robot postures that provide good viewpoints of the observed part are displayed in Fig. 9. Because the part is small, some of the postures are quite close together. In our practical example, the engineer selected two viewpoints from the side of the object, one for the sides and windings and one for the bottom of the product. The other defects were observed from the top.

The issue with adaptive visual quality inspection is that potentially there are a lot of different motion trajectories between viewpoints. If we consider the trajectories from one viewpoint to another and back as two different motion trajectories and there are m different camera viewpoints (plus the initial robot posture), then there are $(m + 1) \times m$ possible motion trajectories between different viewpoints. In our practical example, this results in 110 possible trajectories. They can be either generated in advance and stored or they can be generated online.

B. MOTION PLANNERS

Many different motion planners exist, as presented in the related work section. We used MoveIt! [35] software package for motion planning and Planner arena [12] for benchmarking of different planners for steering the camera in adaptive visual quality inspection. We tested 21 different motion planners available in Open Motion Planning Library (OMPL) [13]. The planners are listed in Table 3 and the number associated with each planner is used to present the results of our evaluations.

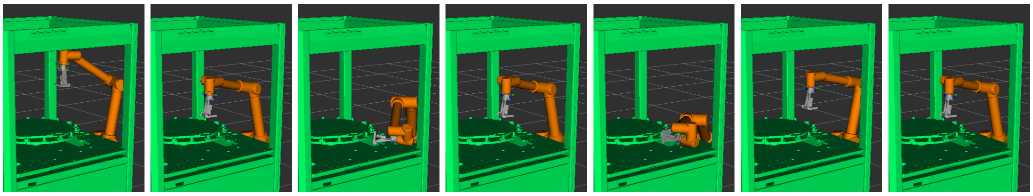
C. EXPERIMENTAL PROTOCOL AND METRICS

Each planner was used to generate trajectories between all possible pairs of 11 robot postures (10 viewpoints + the starting point), 250 times for each pair. Several executions were executed because the planners use random sampling when generating the trajectory points. In general, there is no motion planner that is optimal for every given use-case [12]. Thus, we checked various parameters to determine which planner performed best in our practical example. The calculations were done on a PC with AMD Ryzen Threadripper PRO 5975WX processor and 512GB of RAM.

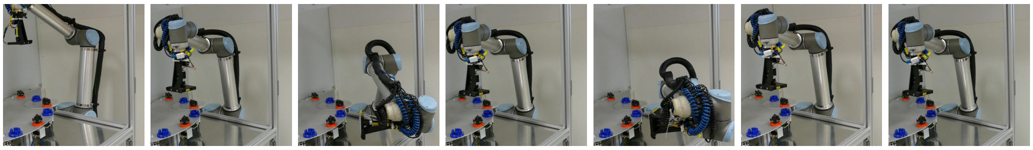
First, one needs to check if the robot motion generated by the selected planner reaches the desired final pose. If the planner does not converge, it is immediately ruled out.

Next, since short cycle times are needed to prevent bottlenecks in the production, the average time required for planning was checked.

The length of motion, as the next evaluation metrics, was computed as the sum of joint angle differences travelled by all



(a)



(b)

FIGURE 9. Seven out of eleven postures for visual quality inspection shown in RViz visualization (a) and in real world (b).

TABLE 3. List of tested OMPL planning algorithms.

Pl. #	Planner name
1	BiEST (Bidirectional version of EST)
2	ProjEST (Projection-based version of EST)
3	LazyPRM
4	LazyPRM*
5	SPARS (SPArse Roadmap Spanner algorithm)
6	SPARS2
7	SBL (Single-query, Bi-directional and Lazy Collision Checking)
8	EST (Expansive Space Trees)
9	LBKPIECE (Lazy BKPIECE)
10	BKPIECE (Bidirectionnal KPIECE)
11	KPIECE (Kinematic Planning by Interior-Exterior Cell Exploration)
12	RRT (Rapidly-exploring Random Trees)
13	RRT Connect
14	RRT*
15	T-RRT (Transition-based RRT)
16	PRM (Probabilistic roadmap)
17	PRM*
18	FMT (Fast Marching Tree algorithm)
19	PDST (Path-Directed Subdivision Trees)
20	STRIDE (Search Tree with Resolution Independent Density Estimation)
21	BiTRRT (Bi-directional Transition-based RRT)

robot joints starting at the initial and final robot configuration $\mathbf{q}_0$ and $\mathbf{q}_n$, respectively

$$L = \sum_{i=1}^{n} \|\mathbf{q}_i - \mathbf{q}_{i-1}\|. \tag{1}$$

Finally, smoothness S of the Cartesian space path, as introduced in [43], was evaluated. It was obtained by averaging the squared angles between consecutive trajectory segments

$$S = \frac{1}{n}\sum_{i=1}^{n-1} \phi_i^2. \tag{2}$$

To compute the angles, we sampled a sequence of $n + 1$ positions $\mathbf{r}_0, \mathbf{r}_1, \ldots, \mathbf{r}_n \in \mathbb{R}^3$ on the Cartesian space path,

TABLE 4. Success rate of planners for 250 planning attempts and 110 movements between the specified camera postures.

#	1	**2**	3	4	5	6	7	8	9	10	11
%	1	1	1	1	0.98	0.66	1	1	1	1	1
#	12	13	14	15	16	17	18	19	20	**21**	
%	0.99	1	0.99	0.98	1	1	1	1	1	**1**	

distributed from the beginning to the end of the path. The angle ϕ_i between each triple of consecutive points $\{\mathbf{r}_{i-1}, \mathbf{r}_i, \mathbf{r}_{i+1}\}$ can be calculated using

$$\phi_i = \pi - \arccos\left(\frac{(\mathbf{r}_{i-1} - \mathbf{r}_i)^\top (\mathbf{r}_{i+1} - \mathbf{r}_i)}{\|\mathbf{r}_{i-1} - \mathbf{r}_i\| \|\mathbf{r}_{i+1} - \mathbf{r}_i\|}\right). \tag{3}$$

VI. MOTION PLANNER EVALUATION

We evaluated the planners listed in Table 3. The results of convergence to the desired posture are shown in Table 4. It can be seen that some planners did not converge to the solution in all 250 planning sessions. These planners were immediately ruled out.

As shown in Fig. 10a, some of the planners took an order of magnitude longer than the others and reached the cut-off time of 2 seconds. These planners were also excluded from further consideration.

Length of motion metrics results, as defined in (1), are presented in Fig. 10b. These were combined with Cartesian path smoothness. Considering the results in Table 4 and Fig. 10, the best result was obtained by planner #21, Bi-directional Transition-based Rapidly-exploring Random Trees (BiRRT). This planner combines the exploratory strength of RRTs with the efficiency of stochastic optimization methods (e.g., Monte Carlo optimization) [44]. The BiRRT method converged to a solution all the time, was very fast, and the resulting trajectories were smooth and short. Variations of RRT, such as T-RRT performed on the same level if not better, but did not converge in all planning attempts. Planners #1, 2, 19 and 20 were excluded because of longer and less smooth trajectories, even though the results were close to the chosen planner.

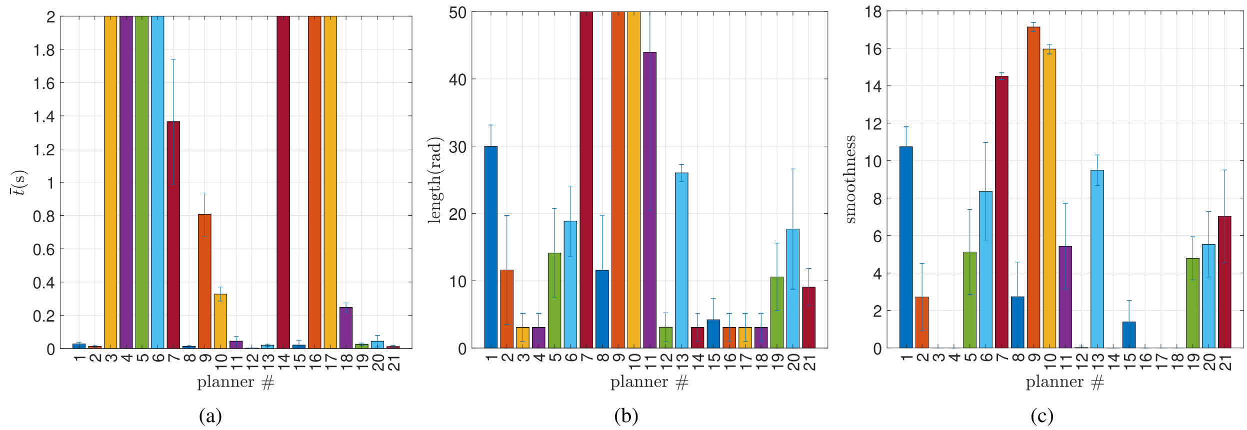


FIGURE 10. Planner evaluation results. (a) planning times. (b) length as defined in (1), cut at 50 rad. (c) smoothness as defined in (2).

VII. DISCUSSION

It is important to note that parts for the automotive industry are produced in large series. Any possible defects are problematic for the overall production process because they might cause delays in the time of delivery of the final product or even recall of the vehicles that are already being used in traffic. Since such recalls can be expensive for the manufacturer and cause the loss of trust from the customers, it is important for the manufacturers to invest and implement better methods for visual quality inspection. It is desirable to reduce the number of undetected faulty parts to zero. Note that technological limitations still cannot provide 100% optical control in real-world production environments.

The defect prediction approach presented in this paper mostly depends on the available data. Additional data-points can always be added and the clustering re-trained. Thus, with an increasing amount of data points for good and faulty parts, it is possible to raise the trust in the prediction of possible defects. Manufacturing companies, however, tend to produce orders of magnitude more of good parts than faulty parts. Thus, collecting a large database, especially at the end of the production process where the parts that reached this point have already passed the intermediate quality checks, is not easy. As can be seen in the practical production example considered in this paper, the database of good and faulty parts (DB1) can be extremely lopsided.

With more data available, the choice of classification algorithms may also change. Our proposed approach shows good classification results even though the amount of relevant data was low. Furthermore, the comparisons show that some approaches are really not appropriate in this case, i.e, supervised learning classifiers are known to require a lot of data [21]. The proposed approach combines both AEs and clustering. Using them separately, for clustering in the data space, or for detecting defects through reconstruction error has not shown good results. Conceptually similar to our method is also deep embedded clustering (DEC), where the clustering is implemented as an additional layer of the encoder. Since it can only be trained on the outcome of already implemented clustering, it offers only an additional implementation which will take the data as the input and provide directly the possibility of belonging to a cluster, without the need to check separately. Thus, the results will be the same.

Our approach can be used in two ways: either to i) check all the parts after the final production stage, but using the prediction model to change the order of inspections, or ii): to only checks the most likely defects. Under i), all defects will be found, but it will take the longest, as shown in Table 5, which shows the average times for 100 iterations of the inspections for all the defect combinations. With enough confidence in the prediction model, we can save close to 80% of the time when checking for only one defect, or around 55% of the time when checking for two, as listed in Table 5. Checking for 3 defects saved roughly 36% of the time, while checking for 4 out of the five most likely defects on average saved around 23% of the inspection time.

In our approach, the robot must frequently move between different pairs of viewpoints no matter if we check for all or only some of the defects. We therefore tested which of the available motion planners is best for our practical production process. As stated in [12], there is no motion planner that works best in every situation. For the considered production process, the extensive statistical evaluation has shown that several planners might be appropriate. The best results were obtained with Bi-directional Transition-based Rapidly-exploring Random Trees (BiTRRT) method. In this example, learning 110 trajectories and then recalling them from the database is still viable. Nonetheless, it is easy to imagine visual inspection processes where many more motion trajectories would need to be generated. This is, however, not necessary because the available motion planners can generate trajectories for visual quality inspection in a fast and reliable manner.

For completeness of the adaptive visual quality inspection approach, we implemented a deep neural network classifier

TABLE 5. Time required for inspecting different combinations of defects using the selected BiRRT motion planner. The Table shows the mean and standard deviation of computational time needed for trajectory planning and execution for different defect combinations, as well as the percent of the saved time compared to the inspection from all viewpoints.

Inspected defect	$\mu(\mathbf{s})$	$\sigma(\mathbf{s})$	saved (%)
All	42.21	0.51	0.00 ± 1.21
bottom	2.66	0.40	93.69 ± 1.02
pins	24.92	0.50	40.96 ± 1.91
porous	11.06	0.44	73.80 ± 1.36
windings	2.68	0.38	93.66 ± 0.98
sides	2.67	0.38	93.67 ± 0.98
Average 1	8.80	0.42	79.16 ± 1.25
bottom + pins	29.12	0.85	31.01 ± 2.84
bottom + porous	20.23	1.46	52.07 ± 4.04
bottom + windings	8.34	0.44	80.24 ± 1.29
bottom + sides	8.73	0.22	79.32 ± 0.77
pins + porous	28.08	0.25	33.47 ± 1.39
pins + windings	29.20	0.37	30.82 ± 1.72
pins + sides	29.22	0.38	30.77 ± 1.73
porous + windings	18.13	0.17	57.05 ± 0.93
porous + sides	18.42	0.38	56.37 ± 1.44
windings + sides	2.71	0.38	93.59 ± 0.98
Average 2	19.22	0.49	54.47 ± 1.71
bottom + pins + porous	32.07	0.47	24.03 ± 2.04
bottom + pins + windings	32.97	0.47	21.88 ± 2.06
bottom + pins + sides	32.93	0.43	22.00 ± 1.96
bottom + porous + windings	24.90	2.22	41.02 ± 5.97
bottom + porous + sides	24.79	1.80	41.26 ± 4.97
bottom + windings + sides	8.71	0.19	79.37 ± 0.70
pins + porous + windings	31.91	0.44	24.40 ± 1.95
pins + porous + sides	32.07	0.38	24.03 ± 1.83
pins + windings + sides	29.10	0.47	31.05 ± 1.94
porous + windings + sides	18.18	0.35	56.94 ± 1.35
Average 3	26.76	0.72	36.6 ± 2.48
bottom + pins + porous + windings	35.82	0.32	15.15 ± 1.78
bottom + pins + porous + sides	35.79	0.48	15.22 ± 2.17
bottom + pins + windings + sides	32.83	0.41	22.23 ± 1.92
bottom + porous + windings + sides	24.77	2.55	41.32 ± 6.76
Average 4	32.27	0.84	23.56 ± 2.91

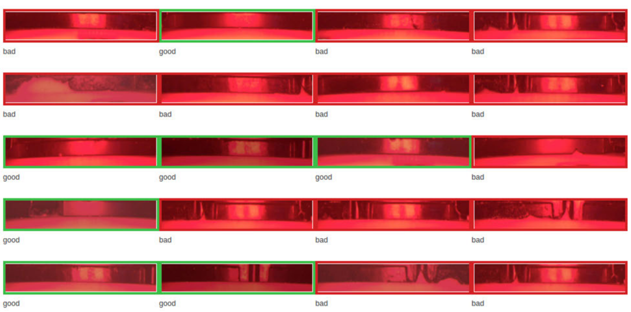


FIGURE 11. Results of final visual quality inspection for the "incomplete bottom edge" (e2) defect of several parts. Faulty parts are marked with a red edge.

that takes the image as the input and classifies the part as good or faulty at the output. The results of this classification for the incomplete bottom edge defect are shown in Fig. 11.

VIII. CONCLUSION

The presented approach, demonstrated on real-world data and objects, takes the production parameters of an injection moulding line to predict potential defects of the manufactured parts. We have shown that both supervised and unsupervised classification methods can struggle with the problem. As is common with learning approaches, a small database prevents the use of supervised learning classifiers, as they either overfit or achieve poor classification results. Our proposed, unsupervised method shows how to use autoencoders and clustering of production parameters to predict what kind of defect might be present.

The evaluation has shown that the clustering and classification are robust against the training of AE networks. The classification results and the comparison to the real data show, that the proposed method can be effectively applied for default prediction. The use of the proposed method can, without having to check the most common defects, reduce their number without introducing a delay in the production. it should be noted, that the proposed approach provides the initial, starting point in the production. Once added to the production line, the database can be potentially increased. Thus, either our approach or even supervised deep learning classification approaches could, at a later stage, be applied more effectively.

We have also analyzed different motion planners to generate motions that bring visual inspection cameras to the required viewpoints in a collision-free manner. The timing results show that considerable time can be saved when only some of the most common potential defects are inspected. However, it is the prediction which potential defects need to be checked for, that makes the approach more reliable.

While in our approach we close the feedback loop between the prediction of the defect and the camera viewpoint, for a true ZDM approach the closed loop should consider changing the production parameters based on the prediction of feedback. This remains an open research question and future work. With a larger database, one could improve the confidence in the predictions from the model. Additionally, a larger database would enable one to evaluate the effect of specific models on the outcome of the production, for example with Shapely values [45], and thus influence the production process in a closed-loop manner.

REFERENCES

[1] H. Foidl and M. Felderer, "Research challenges of Industry 4.0 for quality management," in *Innovations in Enterprise Information Systems Management and Engineering*, M. Felderer, F. Piazolo, W. Ortner, L. Brehm, and H.-J. Hof, Eds., Cham, Switzerland: Springer, 2016, pp. 121–137.

[2] F. Psarommatis, G. May, P.-A. Dreyfus, and D. Kiritsis, "Zero defect manufacturing: State-of-the-art review, shortcomings and future directions in research," *Int. J. Prod. Res.*, vol. 58, no. 1, pp. 1–17, Jan. 2020.

[3] A. A. Nazarenko, J. Sarraipa, L. M. Camarinha-Matos, C. Grunewald, M. Dorchain, and R. Jardim-Goncalves, "Analysis of relevant standards for industrial systems to support zero defects manufacturing process," *J. Ind. Inf. Integr.*, vol. 23, Sep. 2021, Art. no. 100214.

[4] F. Eger, D. Coupek, D. Caputo, M. Colledani, M. Penalva, J. A. Ortiz, H. Freiberger, and G. Kollegger, "Zero defect manufacturing strategies for reduction of scrap and inspection effort in multi-stage production systems," *Proc. CIRP*, vol. 67, pp. 368–373, Jan. 2018.

[5] M. Ismail, N. A. Mostafa, and A. El-assal, "Quality monitoring in multistage manufacturing systems by using machine learning techniques," *J. Intell. Manuf.*, vol. 33, no. 8, pp. 2471–2486, Dec. 2022.

[6] M. Gan and J. Wang, "Applications of image processing technique in porous material characterization," in *Advanced Image Acquisition, Processing Techniques and Applications*, D. Ventzas, Ed., Rijeka, Croatia: IntechOpen, ch. 6, 2012.

[7] J. Wang, P. Fu, and R. X. Gao, "Machine vision intelligence for product defect inspection based on deep learning and Hough transform," *J. Manuf. Syst.*, vol. 51, pp. 52–60, Apr. 2019.

[8] N. Herakovic, *Robot Vision in Industrial Assembly and Quality Control Processes*. London, U.K.: INTECH Open Access, 2010.

[9] Z. Loncarevic, A. Gams, S. Rebersek, B. Nemec, J. Skrabar, J. Skvarc, and A. Ude, "Specifying and optimizing robotic motion for visual quality inspection," *Robot. Comput.-Integr. Manuf.*, vol. 72, Dec. 2021, Art. no. 102200.

[10] J. Zipfel, F. Verworner, M. Fischer, U. Wieland, M. Kraus, and P. Zschech, "Anomaly detection for industrial quality assurance: A comparative evaluation of unsupervised deep learning models," *Comput. Ind. Eng.*, vol. 177, Mar. 2023, Art. no. 109045.

[11] O. Ogorodnyk, O. V. Lyngstad, M. Larsen, and K. Martinsen, "Prediction of width and thickness of injection molded parts using machine learning methods," in *EcoDesign and Sustainability I: Products, Services, and Business Models*. Singapore: Springer, 2021, pp. 455–469.

[12] M. Moll, I. A. Sucan, and L. E. Kavraki, "Benchmarking motion planning algorithms: An extensible infrastructure for analysis and visualization," *IEEE Robot. Autom. Mag.*, vol. 22, no. 3, pp. 96–102, Sep. 2015.

[13] I. A. Sucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robot. Autom. Mag.*, vol. 19, no. 4, pp. 72–82, Dec. 2012.

[14] R. D. Párizs, D. Török, T. Ageyeva, and J. G. Kovács, "Machine learning in injection molding: An Industry 4.0 method of quality prediction," *Sensors*, vol. 22, no. 7, p. 2704, Apr. 2022.

[15] H. Jung, J. Jeon, D. Choi, and J.-Y. Park, "Application of machine learning techniques in injection molding quality prediction: Implications on sustainable manufacturing industry," *Sustainability*, vol. 13, no. 8, p. 4120, Apr. 2021.

[16] O. Ogorodnyk, O. V. Lyngstad, M. Larsen, K. Wang, and K. Martinsen, "Application of machine learning methods for prediction of parts quality in thermoplastics injection molding," in *Advanced Manufacturing and Automation VIII*. Berlin, Germany: Springer, 2019, pp. 237–244.

[17] B. Silva, J. Sousa, and G. Alenya, "Machine learning methods for quality prediction in thermoplastics injection molding," in *Proc. Int. Conf. Electr., Comput. Energy Technol. (ICECET)*, Dec. 2021, pp. 1–6.

[18] B. Silva, R. Marques, T. Santos, J. Sousa, and G. Alenya, "Relevant parameters identification in traditional & stretch and blow thermoplastics injection molding," in *Proc. IEEE 9th Int. Conf. Comput. Intell. Virtual Environments Meas. Syst. Appl. (CIVEMSA)*, Jun. 2022, pp. 1–6.

[19] B. Silva, J. Sousa, and G. Alenya, "Data acquisition and monitoring system for legacy injection machines," in *Proc. IEEE Int. Conf. Comput. Intell. Virtual Environments Meas. Syst. Appl. (CIVEMSA)*, Jun. 2021, pp. 1–6.

[20] L. Wang, M. Han, X. Li, N. Zhang, and H. Cheng, "Review of classification methods on unbalanced data sets," *IEEE Access*, vol. 9, pp. 64606–64628, 2021.

[21] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: A review of classification techniques," *Emerg. Artif. Intell. Appl. Comput. Eng.*, vol. 160, no. 1, pp. 3–24, 2007.

[22] D. Devi, S. K. Biswas, and B. Purkayastha, "A review on solution to class imbalance problem: Undersampling approaches," in *Proc. Int. Conf. Comput. Perform. Eval. (ComPE)*, Jul. 2020, pp. 626–631.

[23] A. Sharma, P. K. Singh, and R. Chandra, "SMOTified-GAN for class imbalanced pattern classification problems," *IEEE Access*, vol. 10, pp. 30655–30665, 2022.

[24] M. Alloghani, D. Al-Jumeily, J. Mustafina, A. Hussain, and A. J. Aljaaf, "A systematic review on supervised and unsupervised machine learning algorithms for data science," in *Supervised and Unsupervised Learning for Data Science*. Cham, Switzerland: Springer, 2020, pp. 3–21.

[25] Y. Zhou, "Rethinking reconstruction autoencoder-based out-of-distribution detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 7369–7377.

[26] X. Huang, Z. Hu, and L. Lin, "Deep clustering based on embedded auto-encoder," *Soft Comput.*, vol. 27, no. 2, pp. 1075–1090, Jan. 2023.

[27] J. Villalba-Diez, D. Schmidt, R. Gevers, J. Ordieres-Meré, M. Buchwitz, and W. Wellbrock, "Deep learning for industrial computer vision quality control in the printing Industry 4.0," *Sensors*, vol. 19, no. 18, p. 3987, Sep. 2019.

[28] V. Zavrtanik, M. Kristan, and D. Skocaj, "Reconstruction by inpainting for visual anomaly detection," *Pattern Recognit.*, vol. 112, Apr. 2021, Art. no. 107706.

[29] B. Triggs and C. Laugier, "Automatic camera placement for robot vision tasks," in *Proc. IEEE Int. Conf. Robot. Autom.*, Nagoya, Japan, May 1995, pp. 1732–1737.

[30] C. Zang and K. Hashimoto, "A flexible camera positioning strategy for robot-based visual inspection applications," in *Proc. IEEE Int. Conf. Mechatronics Autom.*, Chengdu, China, Aug. 2012, pp. 527–532.

[31] R. Bevec, T. Gaspar, and A. Ude, "Robot-driven autofocus control mechanism for an in-hand fixed focus camera," in *Advances in Service and Industrial Robotics*, N. A. Aspragathos, P. N. Koustoumpardis, and V. C. Moulianitis, Eds., Cham, Switzerland: Springer, 2019, pp. 551–559.

[32] P. M. Fresnillo, S. Vasudevan, W. M. Mohammed, J. L. M. Lastra, and J. A. P. Garcia, "Extending the motion planning framework—Moveit with advanced manipulation functions for industrial applications," *Robot. Comput.-Integr. Manuf.*, vol. 83, Oct. 2023, Art. no. 102559.

[33] H. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, W. Burgard, L. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge, MA, USA: MIT Press, May 2005.

[34] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni, "Path planning and trajectory planning algorithms: A general overview," in *Motion and Operation Planning of Robotic Systems* (Mechanisms and Machine Science). Cham, Switzerland: Springer, 2015.

[35] D. Coleman, I. Sucan, S. Chitta, and N. Correll, "Reducing the barrier to entry of complex robotic software: A Moveit! Case study," 2014, *arXiv:1404.3785*.

[36] Z. Kingston, M. Moll, and L. E. Kavraki, "Exploring implicit spaces for constrained sampling-based planning," *Int. J. Robot. Res.*, vol. 38, nos. 10–11, pp. 1151–1178, Sep. 2019.

[37] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 4569–4574.

[38] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "CHOMP: Gradient optimization techniques for efficient motion planning," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2009, pp. 489–494.

[39] M. Likhachev, "Search-based planning library (SBPL)," Robot. Inst., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep., 2016.

[40] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, pp. 5–32, Oct. 2001.

[41] B. A. U. Olimov, K. C. Veluvolu, A. Paul, and J. Kim, "UzADL: Anomaly detection and localization using graph Laplacian matrix-based unsupervised learning method," *Comput. Ind. Eng.*, vol. 171, Sep. 2022, Art. no. 108313.

[42] Z. Wang, C. Da Cunha, M. Ritou, and B. Furet, "Comparison of K-means and GMM methods for contextual clustering in HSM," *Proc. Manuf.*, vol. 28, pp. 154–159, Jan. 2019.

[43] B. Cohen, I. A. Sucan, and S. Chitta, "A generic infrastructure for benchmarking motion planners," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 589–595.

[44] D. Devaurs, T. Siméon, and J. Cortés, "Enhancing the transition-based RRT to deal with complex cost spaces," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 4120–4125.

[45] S. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1–10.

ZVEZDAN LONČAREVIĆ received the B.S. degree in mechatronics engineering from the Faculty of Technical Sciences, University of Novi Sad, in 2017, and the M.S. and Ph.D. degrees in information and communication technologies from the Jožef Stefan International Postgraduate School, Ljubljana, Slovenia, in 2019 and 2023, respectively.

Since 2019, he has been a Research Assistant with the Department of Automatics, Biocybernetics, and Robotics (E1), Jožef Stefan Institute, Ljubljana. His research interests include robotics, reinforcement learning, deep learning, and transfer learning. He has worked on the QU4LITY (Digital Reality in Zero Defect Manufacturing) project, where he applied optimization techniques for zero-defect manufacturing and used deep learning for adaptive visual quality inspection. In addition, he contributed to the GOSTOP (The Building Blocks, Tools, and Systems for the Factories of the Future) project, focusing on learning and optimization for industrial robotics.

SIMON REBERŠEK received the M.S. degree in electrical engineering from the Faculty of Electrical Engineering, University of Ljubljana.

He is currently a Research Assistant with the Jožef Stefan Institute in the field of robotics.

SAMO ŠELA received the B.S. degree from the Faculty of Computer and Information Science, University of Ljubljana, in 2016. He is currently pursuing the Ph.D. degree.

From 2012 to 2022, he held various roles in the company Kolektor in machine vision and deep learning. Since 2022, he has been the Head of Software Engineer at SICK d.o.o., Slovenia. He has actively complemented his computer vision and artificial intelligence knowledge by taking courses in the increasingly widespread mass open educational platforms, such as Coursera and Udacity. He contributed to the realization of demanding machine vision solutions in industrial environments and made vital contributions to developing the KiS, Kolektor Imaging Software. His drive and thinking about the future took him on the road of research and development for the factories of the future at Kolektor Group, in 2016.

JURE SKVARČ received the bachelor's and Ph.D. degrees from the Faculty of Electrical Engineering, University of Ljubljana, in 1992 and 1998, respectively.

Since 1998, his primary focus has been on the implementation of complex image-processing systems in industrial environments. Over the last 25 years, he has been involved in more than 1000 projects for users in various industries. In 1998, he started his non-academic career as an Independent Researcher and became the Technical Director at FDS Research. From 2012 to 2022, he was the Managing Director at Kolektor, BU VISION. Since 2022, he has been responsible for the Research and Development Team, SICK d.o.o., Slovenia. He leads the interdisciplinary team specializing in the research, development and implementation of vision systems. His research interests include image processing, machine vision, machine learning, digital signal processing, and integrating new concepts into the value chain structure, focusing on machine vision and artificial intelligence as key technologies of Industry 4.0. He is active in international organizations, particularly EMVA and VDMA, where he contributed to the first version of the international standard OPC UA Vision CS.

ALEŠ UDE (Member, IEEE) received the Diploma degree in applied mathematics from the University of Ljubljana, Slovenia, in 1990, and the Ph.D. degree in engineering sciences from the University of Karlsruhe, Germany, in 1995.

He is currently the Head of the Department of Automatics, Biocybernetics, and Robotics, Jožef Stefan Institute, Ljubljana. His research interests include robot learning, imitation learning, reconfigurable robotic systems, and humanoid robotics. He has been a coordinator and/or a principal investigator of numerous national and international projects in these areas.

ANDREJ GAMS (Member, IEEE) received the Diploma degree in electrical engineering, in 2004, and the Ph.D. degree in robotics from the University of Ljubljana, Slovenia, in 2009.

From 2012 to 2013, he was a Postdoctoral Researcher with the Biorobotics Laboratory, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland. He is currently the Head of the Humanoid and Cognitive Robotics Laboratory, Department of Automatics, Biocybernetics, and Robotics, Jožef Stefan Institute, Ljubljana. His research interests include robot and imitation learning, applied for industrial and household robotics, and handling of deformable materials with robots.