The International Journal of Advanced Manufacturing Technology
https://doi.org/10.1007/s00170-024-13609-5

ORIGINAL ARTICLE

Exploiting image quality measure for automatic trajectory generation in robot-aided visual quality inspection

Atae Jafari-Tabrizi[1,3] · Dieter P. Gruber[1,3] · Andrej Gams[2]

Received: 29 November 2023 / Accepted: 5 April 2024

Abstract
Currently, the standard method of programming industrial robots is to perform it manually, which is cumbersome and time-consuming. Thus, it can be a burden for the flexibility of inspection systems when a new component with a different design needs to be inspected. Therefore, developing a way to automate the task of generating a robotic trajectory offers a substantial improvement in the field of automated manufacturing and quality inspection. This paper proposes and evaluates a methodology for automatizing the process of scanning a 3D surface for the purpose of quality inspection using only visual feedback. The paper is divided into three sub-tasks in the same general setting: (1) autonomously finding the optimal distance of the camera on the robot's end-effector from the surface, (2) autonomously generating a trajectory to scan an unknown surface, and (3) autonomous localization and scan of a surface with a known shape, but with an unknown position. The novelty of this work lies in the application that only uses visual feedback, through the image focus measure, for determination and optimization of the motion. This reduces the complexity and the cost of such a setup. The methods developed have been tested in simulation and in real-world experiments and it was possible to obtain a precision in the optimal pose of the robot under 1 mm in translational, and 0.1° in angular directions. It took less than 50 iterations to generate a trajectory for scanning an unknown free-form surface. Finally, with less than 30 iterations during the experiments it was possible to localize the position of the surface. Overall, the results of the proposed methodologies show that they can bring substantial improvement to the task of automatic motion generation for visual quality inspection.

Keywords Industrial robotics · Robot learning · Robotic quality inspection · Visual quality inspection

1 Introduction

In modern industrial environment, minimizing the manual work and replacing them with machines performing automated tasks is constantly in demand [1]. Recent developments and breakthroughs in the field of machine vision and artificial intelligence accelerate this transformation even further [2]. Quality inspection, as one of the essential processes in manufacturing, is no exception. If done manually, quality inspection is a highly repetitive and tedious task. Moreover, exposure to long hours of manual labor makes the inspecting operators prone to mistakes in their judgements [3]. New trends in manufacturing require fast and flexible quality inspection processes with reliable outcomes. In this regard, a necessary step is employing industrial robots and AI-based quality inspection methods. An example of a possible robotic inspection scheme is presented in Fig. 1.

However, manual programming of the robot can be a bottleneck in reaching the goal of "minimal human intervention", as it is highly time-consuming [4]. In today's industrial environment, the trend is towards manufacturing components in lower volumes with higher flexibility and adaptability depending on the wishes of the customers [5]. This has led to development of adaptive and reconfigurable robot cells [6], which are suitable to work on several different work-

✉ Atae Jafari-Tabrizi
atae.jafari@pccl.at ; atae.jafari-tabrizi@stud.unileoben.ac.at

Dieter P. Gruber
dieter.gruber@pccl.at

Andrej Gams
andrej.gams@ijs.si

1 Polymer Competence Center Leoben GmbH, Roseggerstraße 12, Leoben 8700, Austria

2 Jožef Stefan Institute, Jamova cesta 39, Ljubljana 1000, Slovenia

3 Montanuniversität Leoben, Franz-Josef Straße 18, Leoben 8700, Austria

Published online: 26 April 2024

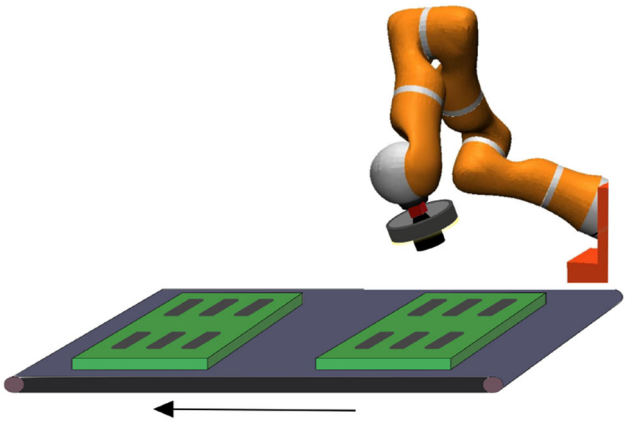


Fig. 1 A conceptual quality inspection setup, in which a robot is scanning the parts which come out of the production on a conveyor belt. Unlike the case depicted here, in some manufacturing facilities it might not be possible to locate the workpieces under inspection in predefined positions, yet a thorough inspection of their surfaces might be needed

pieces. In such a fast-paced environment, the task of manual programming of the robot is unimaginable. For this reason, a methodology enabling the robot to automatically learn a new behavior or adapt itself to the changes and the uncertainties of the environment offers a valuable contribution to the field of manufacturing and quality inspection.

The work presented in this paper is a proof-of-concept to autonomously generate a robot motion in order to scan an unknown free-form surface in a dynamic inspection environment. It has been assumed that an industrial robot, equipped with a camera and a light source on its end-effector, is used to scan the surface of workpieces which arrive to the inspection cell from the production line. The main challenges in this work are: (a) if CAD data of the component are not available, or for any reason not processable, the shape of the surface is not known, (b) even if a trajectory for the robot to scan the shape is available, in many inspection cases the surface is not always located at a fixed, predefined location. Depending on how it has arrived to the inspection station, it can be located at a random point on the inspection table. The proposed approach in this paper introduces solutions to these challenges. A robotic arm, equipped with an industrial camera and a ring light source was used for this work. In all the experiments, the camera was the only external sensor available. Focus measure, an image processing operator, has been used as a metric for measuring the quality of the robot motion [7].

The approach in the paper is separated into three sub-tasks, aligned with the above-listed challenges. In the first sub-task, presented in Section 5, it is shown that the focus measure can be used to obtain the optimal pose of the camera and robot from the surface of the inspected component. In the second sub-task, introduced in Section 6, a trajectory generation method to scan an unknown surface is presented. In the third sub-task, introduced in Section 7, the object's location and orientation with respect to the robot is determined and used to execute a known inspection trajectory. The approach and the subsequent sub-tasks were implemented in a simulation environment and also evaluated with a real-world experimental setup, employing a 7 degree of freedom Kuka LWR-4 collaborative robot. The results of this proof-of-concept work indicate that the proposed methodology has potential to be implemented in a relevant industrial environment at a higher technology readiness level (TRL).

In this work we show how automatic trajectory generation with a robot and a camera can be implemented by only the sensory feedback from the camera and no additional sensors. Autonomous robot movement minimizes manual labor time, and using a single visual camera saves the costs and reduces the complexity of the robotic inspection. Another contribution of this work is employing the focus measure, which is traditionally used for autofocus applications.

2 State of the art and proposed approach

The work presented in this manuscript covers different topics. In this section, the state of the art in each of these areas is shortly summarized.

2.1 Focus measure

The focus measure (FM) is an integer value, representing the image quality, and acting as a figure-of-merit [8]. As there are different image processing techniques available in the literature to calculate the FM, e.g., methods based on calculating the central moments of the images [9], wavelet transformation [10], discrete Chebyshev moments [11], energy of the image [12], singular value decomposition [13], probability coefficients and entropy [14], and texture and structural information of the image [15], there are as well works which compare their performance for different applications [8, 16–18].

After assessing different FM functions, "thresholded absolute gradient" has been chosen for this work. The criteria for choosing it was that it requires substantially less computational time, as also shown in [18]. This function is based on image differentiation, where the calculation in horizontal direction for an image of size $M \times N$ pixels is given as follows [16].

$$F_{\text{horizontal}} = \sum_{i=0}^{M-1} \sum_{j=0}^{N-2} |g(i, j+1) - g(i, j)|, \quad \text{while } |g(i, j+1) - g(i, j)| \geq v \tag{1}$$

Here, $g(i, j)$ is the grey level intensity at the pixel (i, j), and v is the threshold. Equation 1 is extended to calculate the FM both in horizontal and vertical directions [16]. This function reaches its maximum value when the image is the sharpest. The relation between the distance and the angle of the camera from the surface and the FM is shown in Fig. 2. As it can be seen, the model fitted to the angle variation has a higher standard deviation compared to that fitted to the distance variation.

During the simulations the behavior of the FM in vicinity of the optimal distance from the surface was estimated with a Gaussian function (see Eq. 2). In this equation, x is the actual distance of the camera from the surface, and the mean μ represents the optimal distance of the camera from the surface [18]. We illustrate the applicability of modeling the FM variation as a Gaussian distribution with the results depicted in Fig. 2.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}. \tag{2}$$

2.2 Robot-based quality inspection

Vision-based quality inspection is increasingly receiving attention, while there is still relatively large room for further development in the robot-based automation of such systems [19]. Some challenges in implementing fully robotic in-line inspection systems to the manufacturing lines include high initial costs, incompatibility with the current machines, and traditions [20]. Despite these challenges, the developments in the industry and even more so in the computational powers of the computers enable AI-based robotic inspection lines to become widespread.

A robotic inspection setup has been introduced in [21] for detecting geometric and other non-conventional defects. A camera and a light source were installed on the robot arm. For quality inspection of forged metallic components, which are still at high temperatures, a robotic setup has been introduced in [22]. The setup was intended to be used in high-temperature environment. To counter the negative effects of high temperatures, a blue light filter and infrared cut filter were added to the lens of the imaging system, along with a cooling system. The authors of [23] have proposed a hyper-redundant robotic system to inspect the quality of additive manufacturing components. It comprised a mobile omni-drive robot, an industrial robot, and a bionic continuum arm attached to the end-effector of the robotic arm. The authors have also developed a trajectory control strategy. A different solution than optical inspection has been proposed in [24], where a device that gathers acoustic and accelerometer data from the surroundings during the assembly tasks has been introduced. These data are processed with convolutional neural networks for detection of faulty actions in real time.

Here we present a robotic setup which is aimed to be employed at a flexible production line, for manufacturing several components with different shapes in small quantities. This is in contrast with conventional high-volume production lines, where a single component can be manufactured for years. Additionally, it is assumed that in the robotic inspection cell there is no fixed position for the components. Therefore, the robot's starting pose for the scan is not known. Our approach can also be applied for visual inspection of the components where their surfaces are not deterministic after production. Therefore the robot might always need adjustments to be able to inspect them.

2.3 Automatic trajectory generation

Automatic or semi-automatic generation of robot trajectory depending on changes in its environment (commonly known as robot learning) is an attractive research field due to its broad range of applications [25]. A popular methodology for generating the robot trajectory is to employ the information from the CAD data of the component. It has been used for application such as generating a trajectory for painting surfaces [26, 27], laser cladding [28], visual quality inspection [29], and heterogeneous ensemble of car-painting robots [30]. Learning-by-demonstration is another popular method, which has been addressed in [31] for assembly tasks, and in [32] for quality control systems. The authors in [33] have developed a method focusing on energy consumption minimization of the robot. They have reported energy savings up to 10%. An online trajectory generation method has been introduced in [34], which calculates the velocity, acceleration and jerk profiles of the generated trajectory, and also taking the changes in the desired trajectory on-the-fly into account. A method for point-to-point trajectory generation using model predictive control has been introduced in [35]. The data to estimate the position of the ball are sent from the vision system during the motion of the robot, meaning that there is no predefined path at the start of the motion. A Markov decision process-based method to create robotic path for surface inspection has been proposed in [36], whose aim was to scan the surface of the component within the production cycle time. An offline programming method with the capability of overcoming the uncertainties in the environment (e.g., misplacement of the workpiece) has been introduced in [37]. The authors have used a three-dimensional vision system to localize the workpieces, and generate the trajectory accordingly. A method for detecting the welding groove and generating a robotic welding trajectory has been introduced in [38]. For grinding the welded free-form surfaces a method has been proposed in [39]. A hybrid position-force control was used to apply the generated trajectory.

In our work, the automatic trajectory generation is done online. The robot, equipped with a camera and a light source,

optimizes its trajectory episodically based on the shape of the scanned surface. For learning or optimization of the trajectory we employed different algorithms: Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [40], and Reward Weighted Policy Learning with importance sampling (RWPL) which is a simplified version of Policy learning by Weighting Exploration with the Returns (PoWER) [41].

For updating the trajectories, the objective function was the summation of the integer FM values.

$$\text{Objective} = \sum_{i=1}^{K} F_i. \tag{3}$$

Here, K is the total number of images taken during a single scan episode, and F_i is the FM value of the area of interest in image i. In an experimental environment, during each scan episode images are acquired from the surface, and the FM values are calculated for each image. As the scan episode terminates, the sum of these values is calculated. This final value gives an indication of overall quality of the episode. The objective function reaches maximum only (in the ideal case), when the camera is able to acquire the sharpest images during the whole scan episode (see Fig. 2).

2.4 Dynamic movement primitives in robotics

Research on Dynamic Movement Primitives (DMPs) helps develop solutions for sophisticated robotic applications, and facilitate their optimization. We in this work utilize them to encode the trajectories of motion. Here some of the recent advances are mentioned. A task-parametrized DMP methodology has been introduced in [42], which can adapt the learned robotic movement to different environmental situations. The authors also introduce Deep-DMP (D-DMP), which uses deep learning methods as function approximators. A learning-by-demonstration-based robotic trajectory generation and tracking scheme has been developed in [43], which uses DMP for trajectory definition. In [44], the difficulties of applying deep reinforcement learning method to the robotic contact applications (e.g., assembly tasks) has been addressed, and a neural network-based DMP has been proposed. Deep Deterministic Policy Gradient (DDPG), a reinforcement learning method, has been used for learning. This method incorporates also a force controller. Another DMP implementation for contact-rich assembly applications has been introduced in [45]. The authors employ Proximal Policy Optimization (PPO), a reinforcement learning algorithm. In [46] a method for motor skills learning using reinforcement and statistical learning has been introduced which takes advantage of low dimensional latent spaces. The authors have shown that this method can be applied to the real-world robots, by experimenting with a robot throwing a ball to a target. In [29], a different formulation of DMPs, namely Cartesian DMPs (CDMPs), has been used. This formulation is advantageous for considering end-effector orientations and speed, which are essential factors for visual quality inspection of surfaces.

In this work we use DMPs because of their relatively low dimensional encoding of motion trajectories. The trajectory optimization algorithms we apply update the weights of the DMP formulations based on the visual feedback obtained from the scan episodes.

3 Methodology

In this work we aim to introduce a method for generating a robot trajectory for scanning an unknown free-form surface. The exact position of the surface is also unknown. The purpose of scanning the surface is to perform visual inspection. Therefore it is assumed that a camera is installed on the end-effector of the robot. Here, we utilize the same camera for automatic robot path generation. To achieve this, intermediate steps can be defined. These steps act as milestones for reaching the goal. In this context, we introduce three subtasks, which belong to the same general settings, as illustrated in Fig. 3.

The first sub-task was to automatically determine the optimal distance and angle of the visual sensor installed on the end-effector of the robot from the inspection surface. This

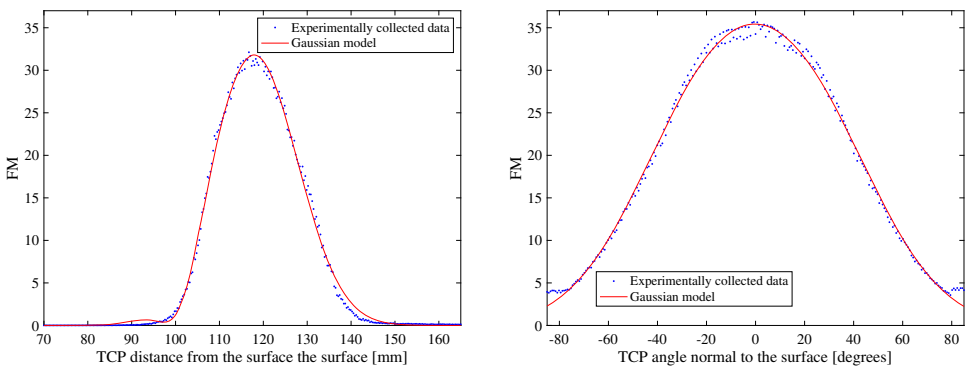


Fig. 2 Variation of the FM function "threshold absolute gradient" with respect to the distance (left) and angle (right) of the camera from the surface, with the Gaussian model fitted to the data

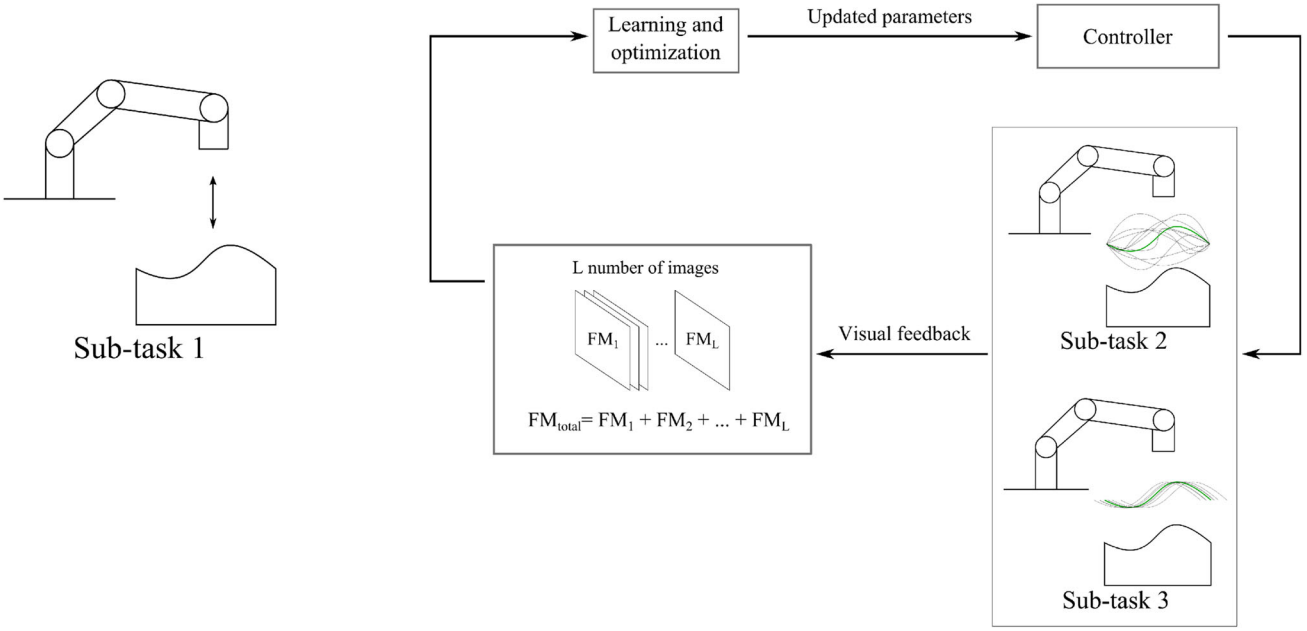


Fig. 3 An illustration of the three sub-tasks to achieve automatic robot trajectory generation for the purpose of visual quality inspection of unknown free-form surfaces. Sub-task 1 (on the left) describes the determination of the optimal distance of the robot end-effector from the surface under inspection (Section 5). Both sub-tasks 2 and 3 are different from sub-task 1 in that they are episodic. Total visual feedback after each episode is used to calculate the total FM value. Sub-task 2 is for generation of the optimal trajectory to scan an unknown surface (Section 6). Sub-task 3 is for determination of the starting pose of the robot for scanning a known surface, given that the position of the surface is unknown (Section 7)

was the most rudimentary step that the robot must be able to perform. Additionally, it served as a proof to the following claim: that only visual feedback from the camera is enough for a robot to generate an optimal trajectory to scan an unknown surface. As a quantifying metric of the visual feedback, the focus measure (FM) was used. As explained more in detail in Section 5, the Fibonacci search method [47] was used to determine the optimal distance and angle of camera placement.

As the capability of the robot to determine its optimal pose with respect to the surface is demonstrated in sub-task 1, the next step was to generate a path used to scan an unknown surface (sub-task 2 in Fig. 3). In this intermediate step, it was assumed that the position of the unknown surface relative to the robot's base was known. Due to the nature of the task, a learning algorithm was adopted. The visual feedback (i.e., FM) was used as a reward to update the robot motion throughout the learning iterations. As already mentioned in Section 2, and discussed in detail in Section 6, DMPs were used as building blocks of the overall robot trajectory. Two optimization algorithms, Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [40], and a simplified version of PoWER algorithm [41] were chosen to be tested.

Finally, in sub-task 3 the goal was to optimize the starting position and orientation of the scan trajectory, depending on where the unknown surface was positioned relative to the robot's base (sub-task 3 in Fig. 3). As the work in this paper proposes a solution for robot visual inspection of unknown free-form surfaces in a flexible inspection environment, approaching the challenge of workpiece position uncertainty is necessary. In this step, two different algorithms were utilized for simulation and real-world experimentations, namely the Nelder-Mead simplex method [48] for simulations and a simple line search for the experiments with the real robot.

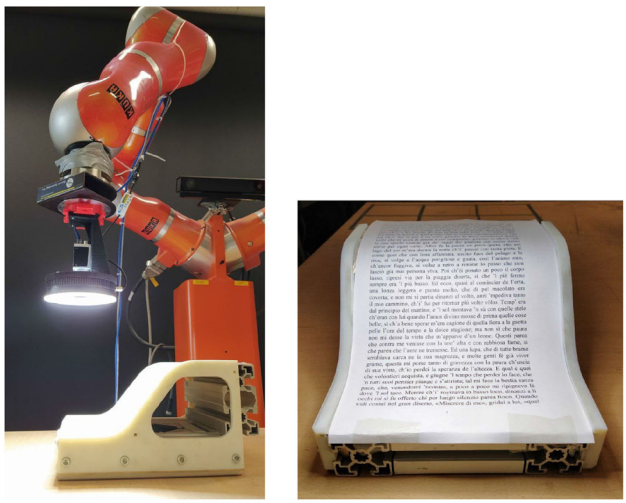
As it can be seen in Fig. 3, the main difference between the sub-tasks is that sub-tasks 2 and 3 have an episodic nature. In sub-task 1, the robot continuously updates its position depending on the continuous visual feedback obtained from the surface, until the optimal position is reached. On the other hand, in sub-tasks 2 and 3 the robot must perform the scan, while the images through the visual feedback are accumulated, and the total FM value is obtained from these images. In the end of each episode, the total FM value is sent to the learning and optimization algorithm, which calculates the updated parameters. The controller uses these parameters to respectively update the motion of the robot.

The premise of the work presented in this paper is that although the methodology and the overall approach are unique, different algorithms and learning methods can be applied. Moreover, factors such as type of the surface, the properties of the camera deployed, the inspection environ-

ment and the hardware make it difficult to determine a single set of best performing algorithms. The aim of this work is not to determine the best algorithms for specific tasks, but to demonstrate to the reader the possibility of applying different optimization approaches for the overall, general task.

4 Hardware and software

The simulations were done in MuJoCo HAPTIX [49] environment. The robot employed was a Kuka LWR-4 collaborative robot for both simulations and experiments. During the experiments Fast Research Interface (FRI) [50] was used to control the robot. All the programming was done in Matlab. The camera used for the experiments was a Basler acA1300-60gm, with a resolution of 1.3 MP, and a frame rate of 60 fps. The lens used had a fixed focal length of 8.5 mm, and an aperture of f/1.3. The aperture was set at the widest possible, to achieve the lowest possible focus range. The exposure time was set to 2000 μs. The light source used was a ring light, HPR2-100SW by the company CCS. A custom-designed and 3D-printed bracket was used to mount the camera and the light source on the robot end-effector, as shown in Fig. 4a. The object used to showcase the applicability of the proposed methodology for automated robot trajectory generation had a curved top surface with an arbitrary text. In the experiments it was located on a stable table near the robot, as shown in Fig. 4a. The curved surface of the object of inspection is shown in Fig. 4b.



(a) (b)

Fig. 4 An overview of the experimental setup used. In (a), the camera and the light source installed on the end effector of the robot along with the 3D object under inspection can be seen. In (b), the curved surface of the object can be seen

5 Finding the optimal pose of the camera from the surface in static mode

5.1 Problem definition

This section presents how the optimal distance and angle from the inspection surface were determined. For fixed-focus cameras (typically used in industrial settings), the distance and angle from the object are often determined manually based on the camera and lens properties. In our case this was done automatically, ensuring optimal image quality.

5.2 Solution approach

We tested the approach with real-world experiments. The Fibonacci search method was used (see Appendix A). It is a suitable search method for this work, as there exists only one optimal solution as shown in Fig. 2. As mentioned in Section 1, the FM is a means to quantify the sharpness of the images. Therefore, here it was used as the optimization objective for the algorithm. The initial search step and direction were chosen empirically.

The motion of the robot was controlled in task space using inverse kinematics. The pseudoinverse method was used to solve the inverse kinematics problem [51]. Given that the required change in position of the end-effector is $\mathbf{e} \in \mathbb{R}^3$, the Jacobian matrix of the robot is defined as $J \in \mathbb{R}^{3\times 7}$, and the change in the joint angles of the robot which would move the robot's end-effector to the target position is $\Delta\mathbf{q} \in \mathbb{R}^7$ (the robot employed in this work has 7 degrees-of-freedom),

$$\Delta\mathbf{q} = J^{+}\mathbf{e}. \tag{4}$$

Here J^{+} is the pseudoinverse of the Jacobian matrix. This approach was used throughout this work for motion control of the robot.

5.3 Distance and angle — results

As the first step, the optimal distance of the camera from the surface has been obtained. During the experiments the camera was positioned normally to the surface. The results of these experiments are shown in Fig. 5a.

Next, keeping the distance constant the angle of the camera with respect to the surface was optimized. The camera was set to an arbitrary angle, and at the end of the experiment it had converged to zero, i.e., perpendicular to the surface. The

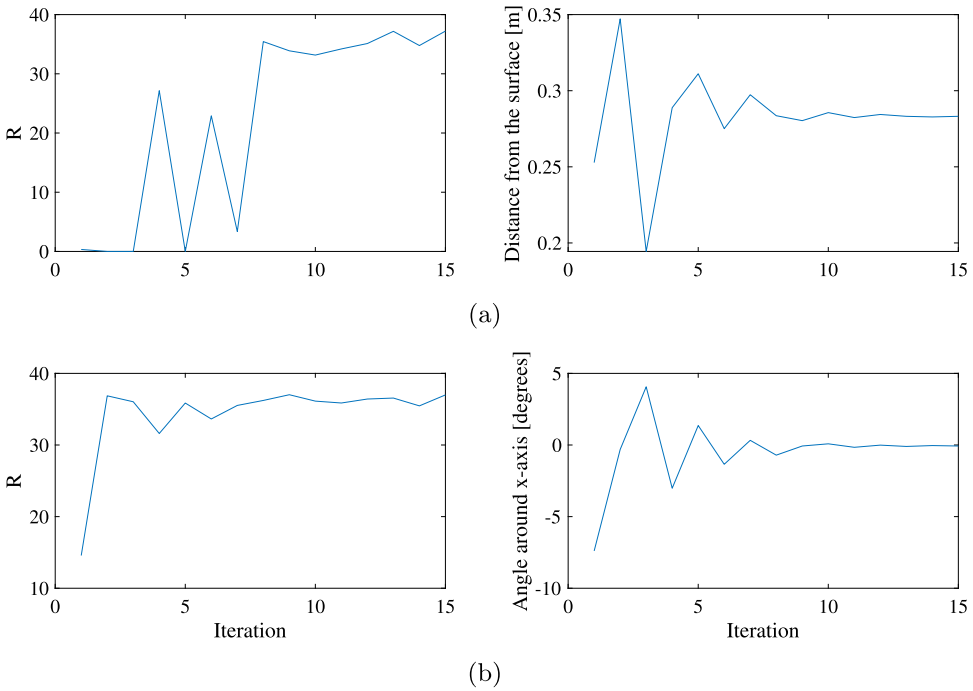


Fig. 5 Results of the experiments (a) to find the optimal distance, (b) to find the optimal angle from the surface. As it can be observed, searching for optimal position and angle stabilizes after approximately 10 iterations

results of this experiment can be seen in Fig. 5b. For both distance and angle optimization experiments the number of iterations was limited to 15. It can be seen that the highest rewards for both position and angle optimizations have been reached in iteration 15. A schematic depicting the axes of motion for determining the position and angle of the camera is shown in Fig. 6a. The image taken from the optimal position can be seen in Fig. 6b. A clear degradation of image sharpness can be observed in Figs. 6c and d where the camera was offset for 3 mm and 20 degrees respectively.

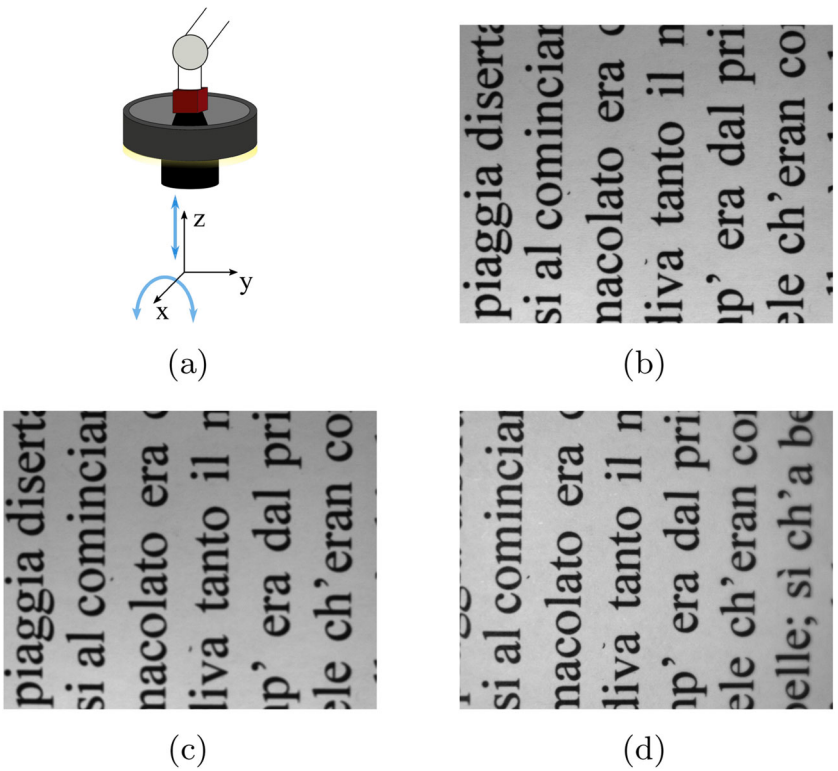


Fig. 6 (a) Translation and rotation performed during the simulations and experiments to find the optimal position and orientation of the robot, (b) image taken at the optimal distance and orientation from the surface, (c) image taken with 3 mm translational offset from the optimal position, (d) image taken with 20° rotational offset from the optimal position

6 Obtaining an optimal scanning trajectory for an unknown curved 3D surface

6.1 Problem definition

After determination of the optimal camera distance and angle from the surface in previous step (Section 5), in this step, the aim was to generate a trajectory for the robot such that it could scan a curved surface, depicted in Fig. 7, while maintaining the highest possible image sharpness. Assuming that the y-axis of the coordinate system $o(x, y, z)$ defined for the surface is aligned with that of the robot base's coordinate system $O(X, Y, Z)$, the distance that the robot should travel along Y-axis is known. This is not a simplification, as this distance can easily be measured or even simply determined — it defines the area one wants to inspect. However, as the shape of the surface was not known, the problem was to determine the motion of the end-effector along the Z-axis, and its rotation around the X-axis, as highlighted in Fig. 7.

6.2 Solution approach

To facilitate effective learning of the motion trajectories, the motions were encoded as DMPs. The two motions (translation and rotation) were solved independently from each other. This resulted in two independent DMP formulations with two independent sets of weights (see Eq. 5). In this equation weights $\mathbf{w}_z \in \mathbb{R}^N$ are for the translation motion, and $\mathbf{w}_\alpha \in \mathbb{R}^N$ for the rotation. In Eq. 5, x_s represents the state of the motion, which is equal to 1 at the start of the motion, and converges to zero as the motion reaches its end.

$$f_z(x_s) = \frac{\sum_{i=1}^{N} \Psi_i(x_s) \mathrm{w}_{zi}}{\sum_{i=1}^{N} \Psi_i(x_s)} x_s$$
$$f_\alpha(x_s) = \frac{\sum_{i=1}^{N} \Psi_i(x_s) \mathrm{w}_{\alpha i}}{\sum_{i=1}^{N} \Psi_i(x_s)} x_s \quad (5)$$

The functions $\Psi_i(x_s)$ are exponential basis functions,

$$\Psi_i(x_s) = \exp\left(-\frac{1}{2\sigma_i^2}(x_s - c_i)^2\right). \quad (6)$$

Here, c_i and σ_i are constants used to define the shape of the individual basis function. Details about the DMP mathematical formulations can be found in Appendix B. The overall learning and optimization process described is visualized in Fig. 8.

Different learning or optimization algorithms can be applied to obtain the desired robot behavior. In this work we tested two different optimization algorithms: Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [40], and Reward Weighted Policy Learning with importance Sampling (RWPL), which is a simplified version of PoWER algorithm [41]. They were chosen because they have shown to be effective in similar tasks [29]. CMA-ES is a stochastic, black box optimization method. Having an initial current solution $\mathbf{w}_m \in \mathbb{R}^N$, the algorithm generates several other candidate solutions (offspring) around it. The samples are generated as follows [52]

$$\mathbf{w}_i \sim \mathbf{w}_m + \sigma\mathcal{N}(\mathbf{0}, \mathbf{C}) \qquad \text{for } i = 1, \ldots, \lambda \quad (7)$$

Here, $\sigma \in \mathbb{R}_+$ is the step size, $\mathbf{C} \in \mathbb{R}^{N\times N}$ is the covariance matrix (defining how the samples are distributed around the current solution), and λ is the number of offspring. By proper updating the sampling parameters, the algorithm converges to the global optimum.

As an alternative, PoWER can be used to learn the DMP weights. It was developed specifically for the reinforcement learning framework applied to learning motor primitives in

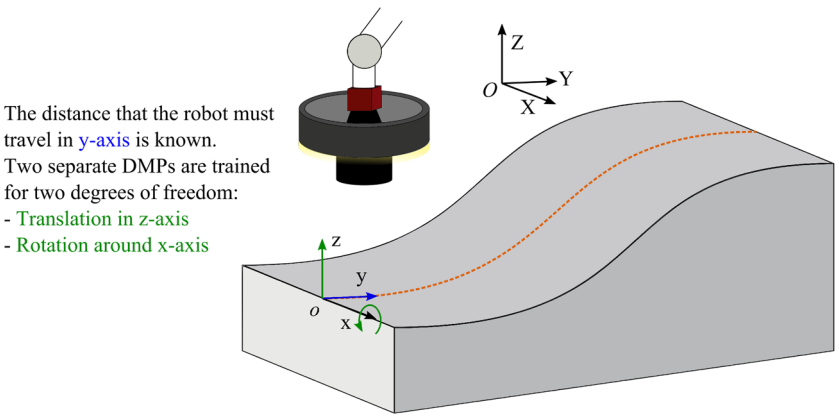


Fig. 7 A simplified visualization of the experimental setup

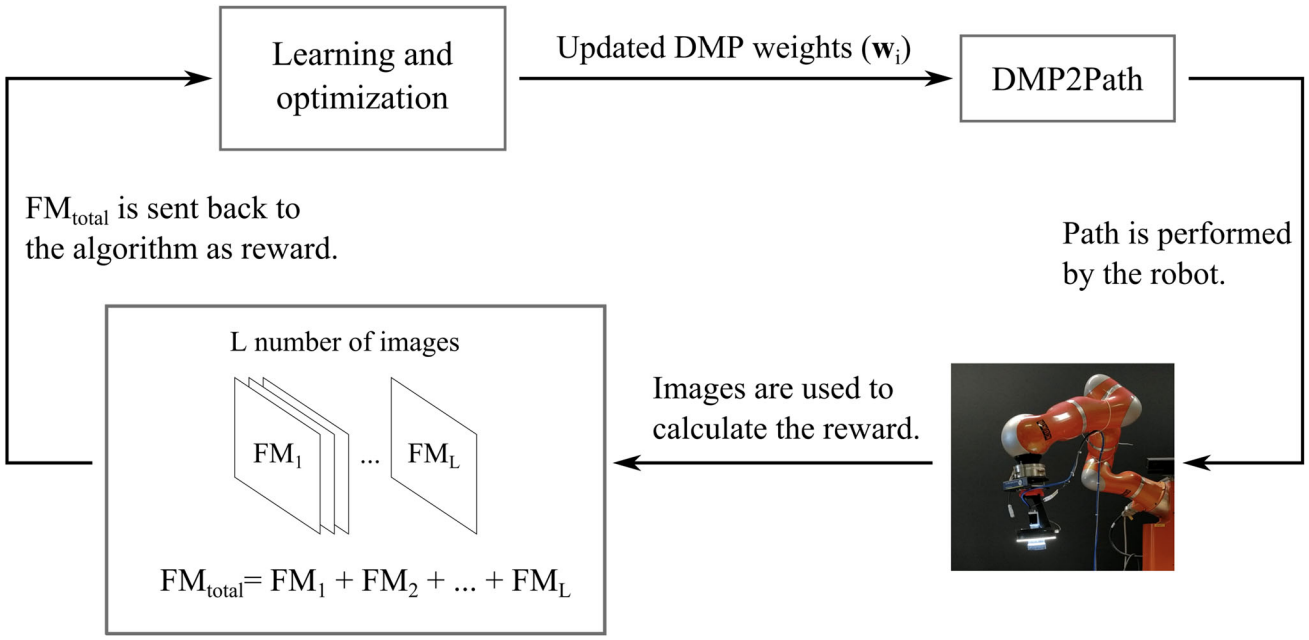


Fig. 8 Overall learning and optimization process of the DMP weights, leading to the convergence to an optimal overall trajectory, utilizing the FM values obtained from the camera as rewards

robotics. In RWPL, a simplified form of the PoWER algorithm, the weights $\mathbf{w} \in \mathbb{R}^n$ are updated as follows

$$\mathbf{w}_{i+1} = \mathbf{w}_i + \frac{\sum_{j=1}^{L} R_{b,j} \times (\mathbf{w}_{b,j} - \mathbf{w}_n)}{\sum_{j=1}^{L} R_{b,j}} \tag{8}$$

In this equation, L is the importance sampling length, $R_{b,j}$ and $\mathbf{w}_{b,j}$ are respectively the reward and weights of the j^{th} best rollout obtained up to n^{th} iteration. See Appendices C and D for more details. The feedback for the algorithms was the sum of the FM values obtained from the images (see Eq. 3).

Figure 8 depicts the main loop for optimizing the motion of the robot. Seen in this figure, as the new weights for the DMP were calculated by the optimization algorithm, they were used in operation to generate the actual robot trajectory ("DMP2Path"). A low-level robot controller was used to implement the motion.

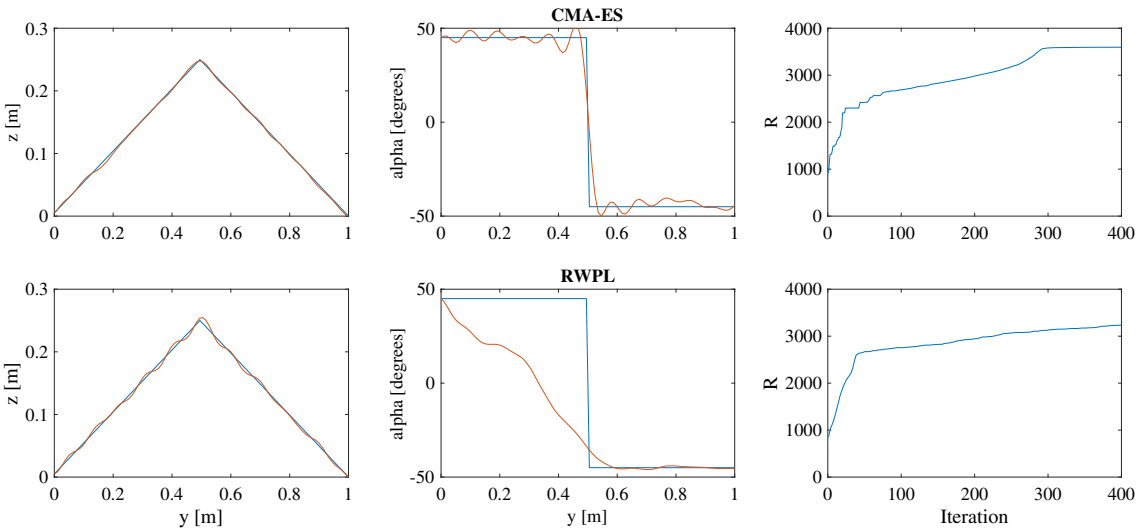


Fig. 9 Simulation results with CMA-ES (top row) and RWPL (bottom row)

6.3 Scanning trajectory — results

In simulations an object with a triangular shape was used to test the approach. The results are shown in Fig. 9. For the CMA-ES algorithm, λ was chosen as 50. For the RWPL algorithm, the number of rollouts was 140, the standard deviation σ of the exploration noise was 25, and the importance sampling length was 5. The number of weights to be optimized for both algorithms was 30. As mentioned before, during the simulations the FM was modeled as a Gaussian distribution (see Eq. 2). We can see from the results that the position aspect of the motion was determined accurately with both algorithms, however for the orientation CMA-ES performed better.

Next, the results of real-world experiments are presented. During the experiments with both algorithms (CMA-ES and RWPL), the initial motion of the robot was random (except its motion in x-direction). Total number of iterations of the CMA-ES algorithm was 38, and that of the RWPL algorithm was 145. The value of λ used in the CMA-ES algorithm was 5. Results can be seen in Fig. 10. The translational motion of the end-effector in z-direction, and the tilting motion of the end-effector obtained by both algorithms can be seen. The slight oscillations of the orientation motion were a consequence DMP formulation, caused by the combination of weighted kernel functions. Also, another reason was the lower influence of the orientation motion on changes in the image quality. As it can be seen, the CMA-ES algorithm required considerably less iterations to reach our predefined threshold. A series of snapshots from the optimized trajectory of the robot can be seen in Fig. 11. Here, from the first image on the left to the next the scanning motion of the robot can be seen sequentially.

7 Obtaining the starting point and scan angle of a free-form surface with known scan trajectory

7.1 Problem definition

A practical problem in the surface quality inspection of a free-form component can arise when it is required for the components to be positioned accurately on a fixture. However, it is not always possible to fix the positions of components on a production line. It may not be economically feasible, or it might be too time-consuming. For example, it is the case when the components fall on a platform from a conveyor belt. In such a case, a manually programmed robot cannot inspect the surface of the component, as it requires accurate positioning of the component for acquiring meaningful data. Therefore, a solution to automatically adapt the scan trajectory to the position of the component can save considerable resources, as it mitigates the need for special hardware fixtures that ensure repeatable positioning.

7.2 Solution approach

In this section we propose a solution for such a problem by optimizing and modifying the starting pose of the scan trajectory with respect to the positioning of the object of inspection. For the simulations and the experiments, the same curved-shape surface used in Section 6 was used.

Assuming that the main coordinate system in this work is located as shown in Fig. 12a, random positioning of the inspection object means that, as shown in Fig. 12b, its position will vary in x- and y-directions (with variations shown as Δx and Δy respectively), and also around z-axis (shown as $\Delta\gamma$) with respect to the base coordinate system. Here it is assumed that the workpiece is located on a flat surface, such that translation in z-axis, and rotational variations around x- and y-axes are always zero.

In the simulations the variation Δx (in x-direction) was always zero. Additionally, assuming the surface under the component was flat, Δz was also equal to zero. The two variables, namely Δy and $\Delta\gamma$ were estimated. Therefore the problem was approached as a learning/optimization problem in a 2-dimensional space. The results obtained can be analogously extended to the case when all three variations are present.

To solve this optimization problem the Nelder-Mead simplex method [48] has been used. In a 2-dimensional space, this method uses a triangle to search for the optimum. The triangle can move around in the search space, and it can change its size. At each iteration, the vertices of the triangle are updated with function evaluations at those vertices, and finally the triangle converges to the optimum, and it can also shrink around it [47]. The details of the algorithm are provided in Appendix E.

During the real-world experiments the workpiece was deviated only in y-direction; that is, there was only one unknown parameter (i.e., Δy), to demonstrate proof-of-concept behavior. A simple line search (see Algorithm 1) was used to find the optimal position of the robot.

7.3 Obtaining the starting point — results

The results of the simulation are shown in Fig. 13. The initial deviations were arbitrarily chosen as 0.2 m in the y-axis (Δy), and 5° around the z-axis ($\Delta\gamma$). In case of correct positioning of the workpiece both of the variables were zero. As it can be seen in Fig. 13, both Δy and $\Delta\gamma$ start to converge rapidly already in the initial iterations. At around 30^{th} iteration both parameters reached their respective expected values. Also the optimization function already reached a value close to the optimum before 25^{th} iteration (Fig. 13).

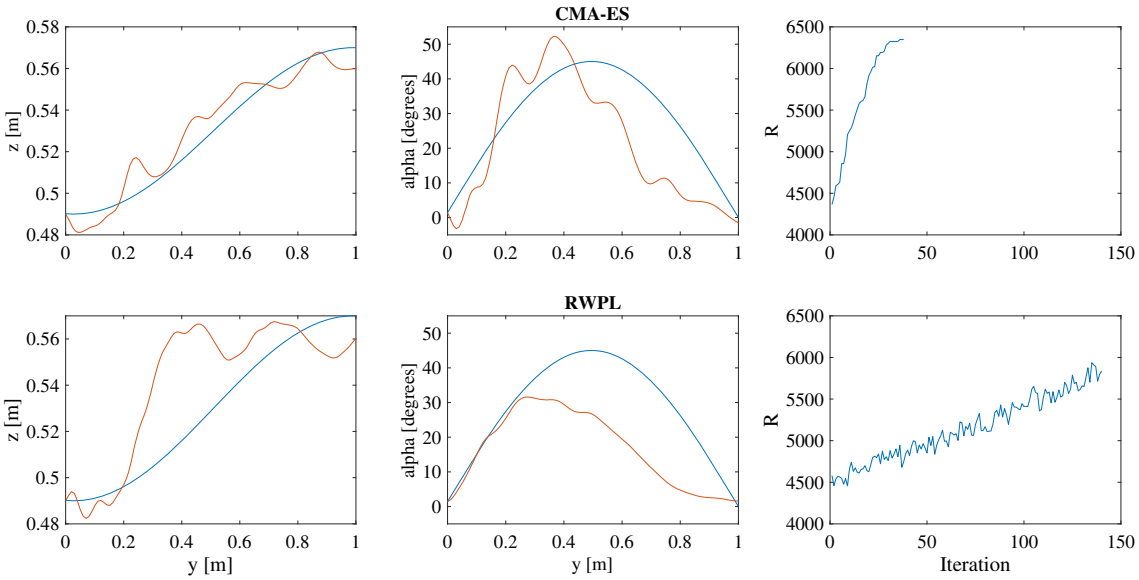


Fig. 10 Experiment results with CMA-ES (top row) and RWPL (bottom row)

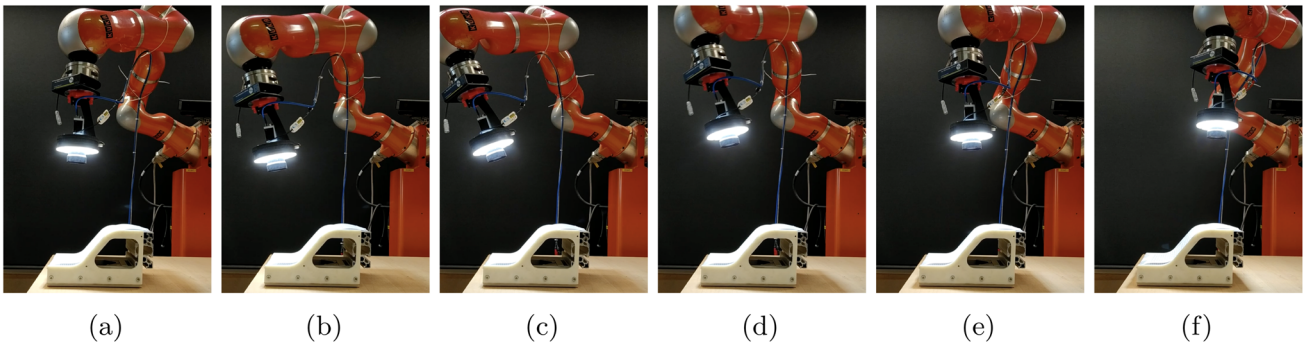


Fig. 11 Snapshots from the optimized trajectory of the robot. The progression of the robot movement while scanning the surface can be seen beginning from the image (a) to the image (f)

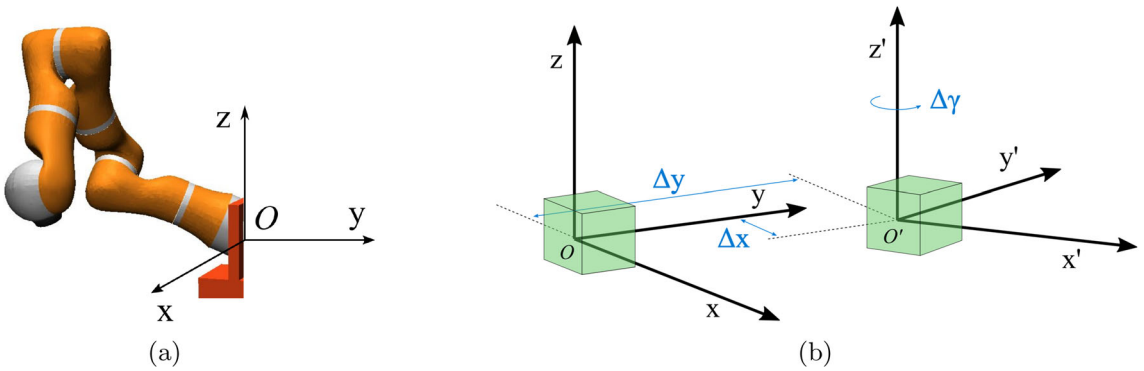


Fig. 12 (a) Coordinate system of the robot, $O(x, y, z)$, (b) Possible positional variations of the workpiece with respect to the robot coordinate system

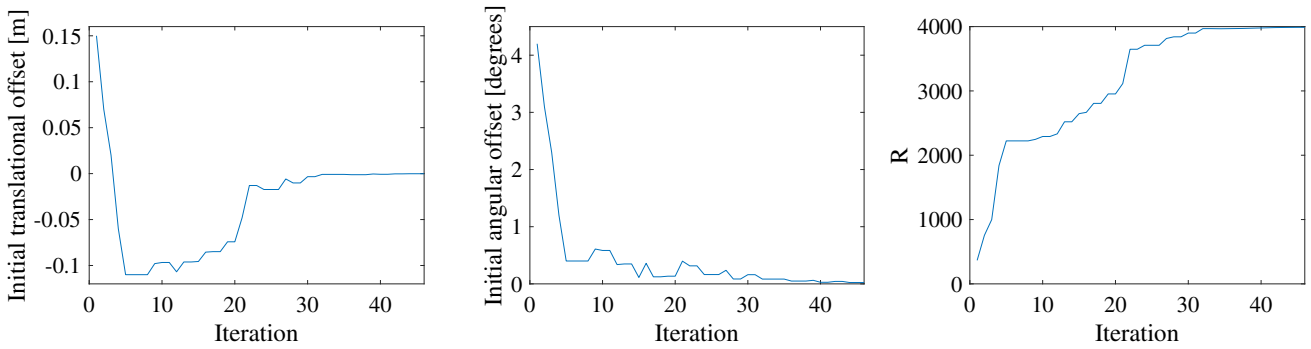


Fig. 13 Simulation results. With increasing iterations both translational offset (left image) and angular offset (center image) stabilize, and approach to zero. Additionally, the increase and final stabilization of the rewards can be observed on the right image

Algorithm 1 Line search algorithm for finding the optimal initial position of the robot

Define the threshold as T.
Define the step as S.
Define initial position of the robot.
Initialize R_{new} and R_{old} such that: $R_{new} >> R_{old}$.
while $|R_{new} - R_{old}| > T$ **do**
  Move robot with S in corresponding direction.
  Follow the defined trajectory.
  Calculate R_{new} as sum of the FM of the images taken during the movement.
  if $R_{new} > R_{old}$ **then**
    Continue.
  else if $R_{new} < R_{old}$ **then**
    Make the step smaller.
    Reverse the direction.
    Continue.
  end if
  $R_{old} = R_{new}$
end while

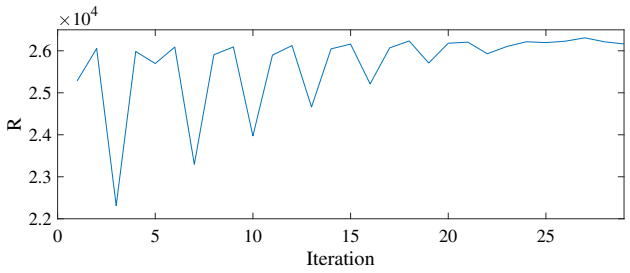


Fig. 14 Reward improvement during the experiments with the real robot. By observing the increase and stabilization of the reward curve with each iteration, the convergence to the optimal starting point can be seen

For the real-world experiments, the linear search method was shown to be able to optimally modify the trajectory to scan the whole surface. The reward improvement during the iterations is shown in Fig. 14.

8 Discussion

Employing additional sensors (e.g., laser distance sensors) could be a solution to both challenges of generating trajectories for unknown surfaces and of localizing the surfaces under inspection. Employing CAD data (as a popular research field, already addressed in Section 2) could provide information to generate robot trajectory in both off- and online manners. Likewise, a mechanical solution (for instance, a fixture) could guarantee the proper positioning of the component. However, as discussed in the introduction, every additional sensor or hardware upgrade increases the cost and complexity of the robotic cell, and introduces new elements that have to be adapted if production is changed to a different component.

Since an industrial camera is already present for visual quality inspection, it can be used to adapt the robot motion any time when a new series or product line is started. Additionally, it is not always possible to use additional sensors due to many reasons. The nature of the components under production, or their surface properties might not always allow the use of additional sensors, or the workpieces might not be identical and therefore the required sensor reading might not be foreseeable.

In this paper we present a solution which can be useful under such exact circumstances. The aim of this work was to assess the feasibility of automatically generating and optimizing the trajectory of a robot, and be as flexible as possible concerning the uncertainties in the environment, while the amount of sensory information available is at a minimum.

In an inspection environment, where it is not possible (due to any foreseeable reasons) to employ additional sensors, such an approach can be the only solution for this challenge. Additionally, for small and medium enterprises (SMEs), where low-volume production is common, such an automated approach can make robotization economical by reducing the re-programming work of the operator and reducing the initial investment by reducing the number of external sensors.

In Section 6, the algorithms CMA-ES and RWPL were used to demonstrate the possibility of applying evolutionary or RL algorithms. Their applicability in industrial setting increases the adaptability of the solution and shows their potential in increasing the productivity. The main reason for choosing Nelder-Mead algorithm in Section 7 for the optimization task was that it converges rapidly in the initial iterations. In an industrial environment where time plays an important factor this is a desired property. Another advantage is that it works well for the problems with noisy parameters. Therefore it is a suitable method for problems, which do not have an accurate solution [53]. In the work presented here, and as shown in Section 2, solutions which lie within the depth of field of the camera/lens were acceptable. Therefore, there was some room for tolerance.

The task of calculation of the focus measure from the taken images was a substantially fast process, so that it never caused a bottleneck regarding the overall duration of the experiments. For determining the optimal distance and angle of the camera from the surface, each iteration was finished within 1 s (including the motion of the robot and the calculation of the focus measure). Thus, the overall process of optimization was done within approximately 15 s. During optimization of the scanning trajectory, and also localization of the free-form surface, the duration of the robot scan trajectory was measured to be on average 5.3 s. The total iteration duration (including the time for the robot to move back to the starting pose) was approximately 10.4 s. Thus, the CMA-ES and the RWPL algorithms took approximately 7 and 25 min to converge, respectively. And the line search algorithm needed approximately 5 min to localize the position of the surface.

As this work is based on a novel approach with relatively simple simulation and experimental implementations, there is room for improvement and optimization. The focus measure function is highly dependent on the surface texture and illumination characteristics. Therefore depending on different practical applications, different textures and different illumination strategies can be examined. The shape of the surface used during the experiments was rather basic. A system whose aim is to be used on a production line must be able to deal with complex shapes. The presented approaches, however, can also deal with more complex shapes, although one must reckon with higher number of iterations. We will also test different optimization algorithms that are relatively more sample efficient for acquiring the desired robot motion. This would enable us to test the method with more complex free-form surfaces.

9 Conclusion

In this work, a method for automatic adaptation of an industrial robot to different uncertainties in its environment, based only on the focus measure feedback quantity is presented. In a robot-based surface quality inspection scenario, the robot's end-effector is equipped with a camera and a light source. Here, the same camera was used for robot motion generation and adaptation. The task could be divided into three subtasks: (1) Obtaining the optimal distance and orientation of the end-effector from the surface, (2) Generating a trajectory to scan an uneven surface, which is not previously known, (3) Adapting the starting point of the scanning motion of the end-effector, given that the exact position of the surface is not known. The first sub-task was tested with real-world experiments. Fibonacci search method was used to obtain optimal distance and orientation. The second and third subtasks were implemented both in simulations and in the real world. CMA-ES and RWPL algorithms were used to generate a trajectory to scan the unknown surface. For the third subtask, in the simulations the Nelder-Mead search method was used. While as a proof-of-concept during the experiments only a simple line search method was used, more complex problems can also be solved with the proposed methods. For the first sub-task using the focus measure precisions under 1 mm in translational, and 0.1^{o} in angular directions were obtained. In the experiments carried out for the second subtask it is shown that the optimized trajectory was obtained within 50 iterations for scanning an unknown free-form surface. Finally, in the framework of the third sub-task it was possible to localize the position of the surface with less than 30 iterations during the experiments.

The algorithms used in this work were chosen based on their simplicity to be implemented and be applied. In the future, further algorithms will be tested. With the abovementioned improvements and assessments, the current work has the potential to provide solutions to real-world challenges in the field of manufacturing.

This approach of generating a desired behavior for a robot can also be applied to other novel inspection and characterization methods which employ different sensor technology than an imaging sensor, such as a tactile sensor.

Appendix A. Fibonacci search

Fibonacci search is a bracketing method [47]. Bracketing methods are suitable for univariate functions. Given that a minimum exists in a selected interval, these methods shrink around the minimum. In Fibonacci method the number of function evaluations is limited. Defining the parameters of the search method as follows,

$$s = \frac{1-\sqrt{5}}{1+\sqrt{5}}$$

$$\phi = \frac{1+\sqrt{5}}{2}$$

$$\rho = \frac{1}{\frac{\phi \times (1-s^{n+1})}{1-s^{n}}}$$

$d = \rho \times b + (1 - \rho) \times a$
the search algorithm is given below.

Algorithm 2 Fibonacci search method (adapted from [47])

Initialize:
$\epsilon \leftarrow 0.01$
$y_d \leftarrow f(d)$
for $i = 1 : (n - 1)$ **do**
  if $i = n - 1$ **then**
    $c \leftarrow \epsilon \times a + (1 - \epsilon) \times d$
  else
    $c \leftarrow \rho \times a + (1 - \rho) \times b$
  end if
  $y_c \leftarrow f(c)$
  if $y_c < y_d$ **then**
    $b, d, y_d \leftarrow d, c, y_c$
  else
    $a, b \leftarrow b, c$
  end if
  $\rho \leftarrow \frac{1}{\frac{\phi \times (1 - s^{n-i+1})}{1 - s^{n-i}}}$
end for

Appendix B. Dynamic movement primitives

DMPs are used to model complex motor skills in robotics with a well-known dynamical system which also includes a forcing term [54]. Using the forcing term enables it to modulate the dynamical system to achieve various behaviors, which later can be transferred to a robot. The mathematical formulation to model the behavior of the robot has the following form [54]

$$\begin{aligned} \tau \dot{z} &= \alpha_z(\beta_z(g - y) - z) + f, \\ \tau \dot{y} &= z. \end{aligned} \tag{9}$$

In this formulation τ is a time constant, α_z and β_z are positive constants, y converges to the goal g, and f is the forcing term, given as

$$f(x) = \frac{\sum_{i=1}^{N} \Psi_i w_i}{\sum_{i=1}^{N} \Psi} x(g - y_0). \tag{10}$$

Here, x is a replacement of explicit dependence of the system on time. In the beginning of the motion it is 1 and it converges to 0 as the motion reaches its end. The independence from time enables the individual dynamical systems to be easily coupled with other systems. The term $(g - y_0)$ contributes to the amplitude of the motion, to make it scalable.

And Ψ_i is defined as

$$\Psi_i(x) = \exp\Big(-\frac{1}{2\sigma_i^2}(x - c_i)^2\Big). \tag{11}$$

The terms σ_i and c_i are constants of the basis function. There are N basis functions. By choosing certain N number of weights (w_i where $i = 1, ..., N$), the forcing term can make y in Eq. 9 to follow a certain trajectory. Therefore the ultimate goal is to find the correct weights corresponding to the behavior that is desired from the robot.

Algorithm 3 CMA-ES (adapted form [40])

Define the parameters λ, $w_{i=1\ldots\lambda}$, c_σ, d_σ, c_c, c_1, and c_μ.
Initialize $\mathbf{p}_\sigma \leftarrow \mathbf{0}, \mathbf{p}_c \leftarrow \mathbf{0}$, covariance matrix $\mathbf{C} \leftarrow \mathbf{I}$, $g \leftarrow 0$, mean $\mathbf{x}_m \in \mathbb{R}^n$, and step size $\sigma \in \mathbb{R}_+$.
while termination criteria is not met **do**
  Sample λ new points: $\mathbf{x}_i \sim \mathbf{x}_m + \sigma\mathcal{N}(\mathbf{0}, \mathbf{C})$ for $i = 1, \ldots, \lambda$
  Update the mean: $\mathbf{x}_m \leftarrow \mathbf{x}_m + \sigma \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda}$, where $\mathbf{y}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{C})$, and $\sum_{i=1}^{\mu} w_i = 1$, $w_i > 0$ for $i = 1 \ldots \mu$.
  Update the step size: $\sigma \leftarrow \sigma \times \exp(\frac{c_\sigma}{d_\sigma}(\frac{||\mathbf{p}_\sigma||}{\mathbb{E}||\mathcal{N}(\mathbf{0},\mathbf{I})||} - 1))$, where $\mathbf{p}_\sigma = (1 - c_\sigma)\mathbf{p}_\sigma + \sqrt{c_\sigma(2 - c_\sigma)\mu_{eff}}\mathbf{C}^{-\frac{1}{2}}(\sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda})$.
  Update the covariance matrix: $\mathbf{C} \leftarrow \mathbf{C} + c_1\mathbf{p}_c\mathbf{p}_c^{\mathrm{T}} + c_\mu \sum_{i=1}^{\lambda} w_i^{\mathrm{o}}\mathbf{y}_{i:\lambda}\mathbf{y}_{i:\lambda}^{\mathrm{T}}$, where $w_i^{\mathrm{o}} = w_i \times (1 \text{ if } w_i \geq 0 \text{ else } \frac{n}{||\mathbf{C}^{-\frac{1}{2}}\mathbf{y}_{i:\lambda}||^2})$, and $\mathbf{p}_c = (1 - c_c)\mathbf{p}_c + h_\sigma\sqrt{c_c(2 - c_c)\mu_{eff}}(\sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda})$.
  (Note: h_σ is the Heaviside function.)
end while

Appendix C. Covariance Matrix Adaptation Evolution Strategy (CMA-ES)

CMA-ES is a black box, stochastic search method. Thorough explanation of this method can be found in [40, 52]. The algorithm is given below (Algorithm 3).

Algorithm 4 RWPL (adapted from [41] and simplified)

Define initial policy parameters $\boldsymbol{\theta}_0$, number of rollouts N, importance sampling length L, and exploration noise σ.
while convergence $\boldsymbol{\theta}_{k+1} \approx \boldsymbol{\theta}_k$ is not met **do**
  Perform N rollouts: $\boldsymbol{\theta}_i \leftarrow \boldsymbol{\theta} + \epsilon_t$, $i = 1 \ldots N$, where $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$.
  Collect: $(i, \boldsymbol{\theta}_i, R_i)$ for $i = 1 \ldots N$ where R_i is the reward.
  Sort the rollouts based on their rewards: $(l, \boldsymbol{\theta}_l, Q_l)$ for $l = 1 \ldots L$.
  Update the policy using $\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \frac{\sum_{l=1}^{L}(\boldsymbol{\theta}_l - \boldsymbol{\theta}_k)Q_l}{\sum_{l=1}^{L} Q_l}$.
end while

Appendix D. Reward weighted policy learning (RWPL)

RWPL is a simplified form of the algorithm PoWER [41]. It is a policy learning method which focuses on the motor primitives (see Algorithm 4).

Algorithm 5 Nelder-Mead triangle search method (adapted from [47])

while convergence criteria is not met **do**
  Compute the reflection: $\mathbf{x}_r \leftarrow \overline{\mathbf{x}} + \alpha(\overline{\mathbf{x}} - \mathbf{x}_h)$, where $\alpha > 0$ and is typically equal to 1.
  if $y_r < y_l$ **then** compute the expansion: $\mathbf{x}_e \leftarrow \overline{\mathbf{x}} + \beta(\mathbf{x}_r - \overline{\mathbf{x}})$, where $\beta > \max(1, \alpha)$ and is typically set to 2.
    if $y_e < y_r$ **then** replace $\mathbf{x}_h$ with $\mathbf{x}_e$.
    else replace $\mathbf{x}_h$ with $\mathbf{x}_r$.
    end if
  else
    if $y_r \geq y_s$ **then**
      if $y_r < y_h$ **then** replace $\mathbf{x}_h$ with $\mathbf{x}_r$.
      end if
      Compute the contraction: $\mathbf{x}_c \leftarrow \overline{\mathbf{x}} + \gamma(\mathbf{x}_h - \overline{\mathbf{x}})$.
      if $y_c > y_h$ **then** shrink by replacing all $\mathbf{x}^{(i)}$ with $\frac{\mathbf{x}^{(i)}+\mathbf{x}_l}{2}$.
      else replace $\mathbf{x}_h$ with $\mathbf{x}_c$.
      end if
    else replace $\mathbf{x}_h$ with $\mathbf{x}_r$.
    end if
  end if
end while

Appendix E. Nelder-Mead simplex search

Nelder-Mead search is composed of rules, which determine the behavior of the simplex (in the case of this work, the triangle) at each iteration. Let us assume the vertex of the triangle with the highest function value is $\mathbf{x}_h$, $\mathbf{x}_l$ is the vertex with the lowest value, and $\mathbf{x}_m$ has the second highest value. The mean of the two lowest vertices, $\overline{\mathbf{x}}$ is defined as $\overline{\mathbf{x}} = \frac{\mathbf{x}_m+\mathbf{x}_l}{2}$. For any point $\mathbf{x}_\theta$ the function value is $y_\theta = f(\mathbf{x}_\theta)$. For a pseudocode see Algorithm 5.

Acknowledgements The authors gratefully acknowledge Simon Reberšek for his technical support and contributions to this research.

Funding Open access funding provided by Montanuniversität Leoben. Part of the research work of this paper was performed at the Polymer Competence Center Leoben GmbH (PCCL, Austria) within the framework of the COMET-program of the Federal Ministry for Transport Innovation and Technology and the Federal Ministry of Digital and Economic Affairs (with Grant No.: 879785). The PCCL is funded by the Austrian Government and the State Governments of Styria, Lower Austria and Upper Austria. The work was partially funded (A.G.) by ARIS research grant Robot Textile and Fabric Inspection and Manipulation – RTFM (J2-4457), DIGITOP project funded by Ministry of Higher Education, Science and Innovation of Slovenia, Slovenian Research and Innovation Agency and European Union – NextGenerationEU, and ARIS program group Automation, Robotics, and Biocybernetics (P2-0076).

Declarations

Competing interests The authors declare no competing interests.

References

1. Arents J, Greitans M (2022) Smart industrial robot control trends, challenges and opportunities within manufacturing. Applied Sciences 12(2). https://doi.org/10.3390/app12020937. https://www.mdpi.com/2076-3417/12/2/937
2. Zeng R, Wen Y, Zhao W et al (2020) View planning in robot active vision: A survey of systems, algorithms, and applications. Computational Visual Media 6(3):225–245. https://doi.org/10.1007/s41095-020-0179-3
3. Yang J, Li S, Wang Z, et al (2020) Using deep learning to detect defects in manufacturing: A comprehensive survey and current challenges. Materials 13(24). https://doi.org/10.3390/ma13245755. https://www.mdpi.com/1996-1944/13/24/5755
4. Ratiu M, Prichici M (2017) Industrial robot trajectory optimization-a review. MATEC Web of Conferences 126(02):005. https://doi.org/10.1051/matecconf/201712602005
5. Kappassov Z, Corrales JA, Perdereau V (2015) Tactile sensing in dexterous robot hands - review. Robot Auton Syst 74:195–220. https://doi.org/10.1016/j.robot.2015.07.015. https://www.sciencedirect.com/science/article/pii/S0921889015001621
6. Gašpar T, Deniša M, Radanovič P et al (2020) Smart hardware integration with advanced robot programming technologies for efficient reconfiguration of robot workcells. Robotics and Computer-Integrated Manufacturing 66(101):979. https://doi.org/10.1016/j.rcim.2020.101979
7. Subbarao M, Choi TS, Nikzad A (1993) Focusing techniques. Opt Eng 32(11):2824–2836. https://doi.org/10.1117/12.147706
8. Groen FCA, Young IT, Ligthart G (1985) A comparison of different focus functions for use in autofocus algorithms. Cytometry 6(2):81–91. https://doi.org/10.1002/cyto.990060202. https://arxiv.org/abs/https://onlinelibrary.wiley.com/doi/pdf/10.1002/cyto.990060202
9. Zhang Y, Zhang Y, Wen C (2000) A new focus measure method using moments. Image Vis Comput 18(12):959–965. https://doi.org/10.1016/S0262-8856(00)00038-X. https://www.sciencedirect.com/science/article/pii/S026288560000038X
10. Kautsky J, Flusser J, Zitová B et al (2002) A new wavelet-based measure of image focus. Pattern Recogn Lett 23(14):1785–1794. https://doi.org/10.1016/S0167-8655(02)00152-6. https://www.sciencedirect.com/science/article/pii/S0167865502001526

11. Yap P, Raveendran P (2004) Image focus measure based on chebyshev moments. IEE Proceedings - Vision, Image and Signal Processing 151:128–136(8). https://digital-library.theiet.org/content/journals/10.1049/ip-vis_20040395
12. Shen CH, Chen H (2006) Robust focus measure for low-contrast images. In: 2006 Digest of technical papers international conference on consumer electronics, pp 69–70. https://doi.org/10.1109/ICCE.2006.1598314
13. Wee CY, Paramesran R (2008) Image sharpness measure using eigenvalues. In: 2008 9th International conference on signal processing, pp 840–843. https://doi.org/10.1109/ICOSP.2008.4697259
14. Rajevenceltha J, Gaidhane VH (2020) A novel approach for image focus measure. SIViP 15:547–555
15. Rajevenceltha J, Gaidhane VH (2022) An efficient approach for no-reference image quality assessment based on statistical texture and structural features. Eng Sci Technol Int J 30(101):039. https://doi.org/10.1016/j.jestch.2021.07.002. https://www.sciencedirect.com/science/article/pii/S2215098621001609
16. Santos A, Ortiz de Solórzano C, Vaquero JJ et al (1997) Evaluation of autofocus functions in molecular cytogenetic analysis. J Microsc 188(3):264–272. https://doi.org/10.1046/j.1365-2818.1997.2630819.x. https://arxiv.org/abs/https://onlinelibrary.wiley.com/doi/pdf/10.1046/j.1365-2818.1997.2630819.x
17. Huang W, Jing Z (2007) Evaluation of focus measures in multi-focus image fusion. Pattern Recogn Lett 28(4):493–500. https://doi.org/10.1016/j.patrec.2006.09.005. https://www.sciencedirect.com/science/article/pii/S0167865506002352
18. Pertuz S, Puig D, Garcia MA (2013) Analysis of focus measure operators for shape-from-focus. Pattern Recogn 46(5):1415–1432. https://doi.org/10.1016/j.patcog.2012.11.011. https://www.sciencedirect.com/science/article/pii/S0031320312004736
19. Babic M, Farahani MA, Wuest T (2021) Image based quality inspection in smart manufacturing systems: A literature review. Procedia CIRP 103:262–267. 9th CIRP Global Web Conference–Sustainable, resilient, and agile manufacturing and service operations : Lessons from COVID-19. https://doi.org/10.1016/j.procir.2021.10.042. https://www.sciencedirect.com/science/article/pii/S2212827121008830
20. Azamfirei V, Granlund A, Lagrosen Y (2023) Lessons from adopting robotic in-line quality inspection in the swedish manufacturing industry. Procedia Comput Sci 217:386–394. 4th International Conference on Industry 4.0 and Smart Manufacturing. https://doi.org/10.1016/j.procs.2022.12.234. https://www.sciencedirect.com/science/article/pii/S1877050922023122
21. Kefer M, Tian J (2016) An intelligent robot for flexible quality inspection. In: 2016 IEEE International Conference on Information and Automation (ICIA), pp 80–84, 10.1109/ICInfA.2016.7831800
22. Han L, Cheng X, Li Z, et al (2018) A robot-driven 3d shape measurement system for automatic quality inspection of thermal objects on a forging production line. Sensors 18(12). https://doi.org/10.3390/s18124368. https://www.mdpi.com/1424-8220/18/12/4368
23. Lakhal O, Melingui A, Dherbomez G et al (2019) Control of a hyper-redundant robot for quality inspection in additive manufacturing for construction. In: 2019 2nd IEEE International conference on soft robotics (RoboSoft), pp 448–453. https://doi.org/10.1109/ROBOSOFT.2019.8722720
24. Sarivan IM, Greiner JN, Álvarez DD, et al (2020) Enabling real-time quality inspection in smart manufacturing through wearable smart devices and deep learning. Procedia Manufacturing 51:373–380. 30th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2021). https://doi.org/10.1016/j.promfg.2020.10.053. https://www.sciencedirect.com/science/article/pii/S2351978920319089
25. Liu Z, Liu Q, Xu W et al (2022) Robot learning towards smart robotic manufacturing: A review. Robotics and Computer-Integrated Manufacturing 77(102):360. https://doi.org/10.1016/j.rcim.2022.102360. https://www.sciencedirect.com/science/article/pii/S0736584522000485
26. Chen H, Xi N (2008) Automated tool trajectory planning of industrial robots for painting composite surfaces. The International Journal of Advanced Manufacturing Technology 35:680–696
27. Andulkar MV, Chiddarwar SS (2015) Incremental approach for trajectory generation of spray painting robot. Ind Robot 42:228–241
28. Zheng H, Cong M, Dong H et al (2017) Cad-based automatic path generation and optimization for laser cladding robot in additive manufacturing. Int J Adv Manuf Technol 92:3605–3614
29. Lončarević Z, Gams A, Reberšek S et al (2021) Specifying and optimizing robotic motion for visual quality inspection. Robot Comput-Integr Manuf 72(102):200
30. Zbiss K, Kacem A, Santillo M et al (2022) Automatic collision-free trajectory generation for collaborative robotic car-painting. IEEE Access 10:9950–9959. https://doi.org/10.1109/ACCESS.2022.3144631
31. Duque DA, Prieto FA, Hoyos JG (2019) Trajectory generation for robotic assembly operations using learning by demonstration. Robot Comput-Integr Manuf 57:292–302
32. Brito T, Queiroz J, Piardi L, et al (2020) A machine learning approach for collaborative robot smart manufacturing inspection for quality control systems. Procedia Manufacturing 51:11–18. 30th International conference on flexible automation and intelligent manufacturing (FAIM2021). https://doi.org/10.1016/j.promfg.2020.10.003. https://www.sciencedirect.com/science/article/pii/S2351978920318588,
33. Hansen C, Öltjen J, Meike D et al (2012) Enhanced approach for energy-efficient trajectory generation of industrial robots. In: 2012 IEEE International conference on automation science and engineering (CASE), pp 1–7. https://doi.org/10.1109/CoASE.2012.6386343
34. Lange F, Albu-Schäffer A (2016) Path-accurate online trajectory generation for jerk-limited industrial robots. IEEE Robot Autom Lett 1(1):82–89. https://doi.org/10.1109/LRA.2015.2506899
35. Ardakani MMG, Olofsson B, Robertsson A, et al (2015) Real-time trajectory generation using model predictive control. In: 2015 IEEE International conference on automation science and engineering (CASE), pp 942–948. https://doi.org/10.1109/CoASE.2015.7294220
36. Jing W, Goh CF, Rajaraman M et al (2018) A computational framework for automatic online path generation of robotic inspection tasks via coverage planning and reinforcement learning. IEEE Access 6:54,854–54,864. https://doi.org/10.1109/ACCESS.2018.2872693
37. Bedaka AK, Vidal J, Lin CY (2019) Automatic robot path integration using three-dimensional vision and offline programming. Int J Adv Manuf Technol 102:1935–1950
38. Peng R, Navarro-Alarcon D, Wu V et al (2020) A point cloud-based method for automatic groove detection and trajectory generation of robotic arc welding tasks. arXiv:2004.12281
39. Feng H, Xukai R, Li L et al (2021) A novel feature-guided trajectory generation method based on point cloud for robotic grinding of freeform welds. Int J Adv Manuf Technol 115. https://doi.org/10.1007/s00170-021-07095-2
40. Hansen N (2016) The cma evolution strategy: A tutorial. https://doi.org/10.48550/ARXIV.1604.00772
41. Kober J, Peters J (2008) Policy search for motor primitives in robotics. In: Koller D, Schuurmans D, Bengio Y, et al (eds) Advances in Neural Information Processing Systems, vol 21. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2008/file/7647966b7343c29048673252e490f736-Paper.pdf
42. Pervez A, Mao Y, Lee D (2017) Learning deep movement primitives using convolutional neural networks. In:

2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids), pp 191–197. https://doi.org/10.1109/HUMANOIDS.2017.8246874
43. Yang C, Chen C, He W et al (2019) Robot learning system based on adaptive neural control and dynamic movement primitives. IEEE Trans Neural Netw Learn Syst 30(3):777–787. https://doi.org/10.1109/TNNLS.2018.2852711
44. Kim YL, Ahn KH, Song JB (2020) Reinforcement learning based on movement primitives for contact tasks. Robot Comput-Integr Manuf 62(101):863
45. Spector O, Zacksenhouse M (2020) Deep reinforcement learning for contact-rich skills using compliant movement primitives. arXiv:2008.13223
46. Pahič R, Lončarevič Z, Gams A et al (2021) Robot skill learning in latent space of a deep autoencoder neural network. Robot Auton Syst 135(103):690. https://doi.org/10.1016/j.robot.2020.103690. https://www.sciencedirect.com/science/article/pii/S0921889020305303
47. Kochenderfer MJ, Wheeler TA (2019) Algorithms for Optimization. The MIT Press
48. Nelder JA, Mead R (1965) A Simplex Method for Function Minimization. Comput J 7(4):308–313. https://doi.org/10.1093/comjnl/7.4.308. https://arxiv.org/abs/https://academic.oup.com/comjnl/article-pdf/7/4/308/1013182/7-4-308.pdf
49. Kumar V, Todorov E (2015) Mujoco haptix: A virtual reality system for hand manipulation. In: 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids), pp 657–663. https://doi.org/10.1109/HUMANOIDS.2015.7363441
50. Schreiber G, Stemmer A, Bischoff R (????) The fast research interface for the kuka lightweight robot
51. Buss S (2004) Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods. IEEE Trans Robotics and Autom 17
52. Auger A, Hansen N (2012) Tutorial cma-es: Evolution strategies and covariance matrix adaptation. In: Proceedings of the 14th annual conference companion on genetic and evolutionary computation. association for computing machinery, New York, NY, USA, GECCO '12, pp 827–848. https://doi.org/10.1145/2330784.2330919
53. Singer S, Nelder J (2009) Nelder-Mead algorithm. Scholarpedia 4(7):2928. revision #91557. https://doi.org/10.4249/scholarpedia.2928
54. Ijspeert AJ, Nakanishi J, Hoffmann H et al (2013) Dynamical movement primitives: Learning attractor models for motor behaviors. Neural Comput 25(2):328–373. https://doi.org/10.1162/NECOa00393