Reinforcement learning for robotic assembly of fuel cell turbocharger parts with tight tolerances

Franziska Aschersleben¹ · Rudolf Griemert¹ · Felix Gabriel¹ · Klaus Dröder¹

Received: 31 March 2020 / Accepted: 9 June 2020 / Published online: 18 June 2020
© The Author(s) 2020

Abstract

The efficiency of a fuel cell is not only dependent on the stack, but also to a large extent on the turbocharger, which is responsible for providing the required airflow. Since the individual components, especially those of the rotor, are subject to high demands on manufacturing accuracy, it is crucial to ensure a precise and robust assembly. In order to achieve a scalable assembly process, this paper presents a method for a robot-based assembly of the rotationally symmetric components of the rotor. The assembly task has been reduced to the two essential problems: search and insertion. On this basis, a system was developed, which is able to learn the joining process independently and compensate for positioning inaccuracies with the help of reinforcement learning in combination with a position-controlled robot. The applied reinforcement learning strategy is based on the measurement data of a 6-axis force/torque sensor, with which the current contact state can be evaluated and a decision for the next step can be made. The experimental verification shows that an automation of the assembly process is possible with the proposed strategy. The robot is able to perform the search operation successfully, whereas limitations to the achievable accuracies of the insertion process could be found.

Keywords Fuel cell · Assembly · Peg-in-hole · Reinforcement learning · Industrial robot

1 Introduction

The fuel cell technology has a great importance regarding the reduction of emission in the automotive sector. While approximately one-fifth of global CO₂ emissions are caused by vehicles with combustion engines [1], fuel cells only emit water as reaction product. The energy originates in an electrochemical cell by converting hydrogen and oxygen [2]. A turbocharger provides the required oxygen in the form of air with a specific volume and pressure and has a major impact on the efficiency of the system [3]. Therefore, accurate manufacturing and assembly are necessary. In comparison to a turbocharger used in cars with conventional engines, it consists of a compressor as well as a turbine and an electric engine in between. The rotor of the examined turbocharger at hand consists of six parts that are rotationally symmetric and manufactured with tight tolerances regarding the diameter,

coaxiality and radial run-out to meet the dynamic requirements even at high rotational speed. The connection of these parts plays an important role and involves the application of a shrink fit, which requires an exact positioning of joining partners. The effective joining diameters varies between 12.5 and 19.5 mm and the minimal clearances lie between 6 and 60 µm. The chamfers are acute-angled with radii between 2 and 5 mm. Since a large-scale production of the turbocharger is intended, a high grade of automation is desired, also for joining the rotor parts. Usually, a high-precision robot is utilized for this assembly, which is associated to high cost and low flexibility, because of the high effort for setup and configuration as well as high susceptibility regarding geometric deviation of parts to be mounted.

In order to meet the challenges of rotor assembly, we reduced the problem to the essential: the joining of rotationally symmetric components with small clearance, also known as peg-in-hole process. If a successful and reliable solution with the greatest possible flexibility is found, the results can be transferred to any specific rotor component.

The peg-in-hole process was already investigated by several researchers [4, 5]. It is the basis of many joining techniques and thus fundamentally important for several

✉ Franziska Aschersleben
f.aschersleben@tu-braunschweig.de

¹ Institute of Machine Tools and Production Technology,
Technische Universität Braunschweig, Langer Kamp 19 b,
38106 Braunschweig, Germany

assembly processes. One of the main strategies is the application of force control with the idea to detect the state of the peg in the hole based on the measured force. This can be conducted by a lightweight robot with integrated force/torque sensors to identify the current contact state and assign them to previously defined states [6]. Here, an accuracy of 6.25 mm was achieved. This strategy also works with external force/torque information [7], the alignment of position and orientation can be done separately or in the same stage. Instead of solely using the force/torque data, a vision system can be integrated into the set up to identify the rough position of the mounting parts [8, 9]. The achievable accuracy lies between 40 and 100 μm . Another method is the implementation of a compliant motion [10, 11], so that the robot can actively control the compliance in different axes, which can simplify the programmed paths and requires a lower precision of the robot, but the motions themselves are less controlled.

Another method is the so-called learning from demonstration method (LfD), where the human transfers its motor skills to the robot by leading the robot with hands. A complex framework is necessary to recover the important parts of the motion and to execute the task successfully [12, 13]. To avoid the need of a human operator, self-learning methods are becoming more popular in the last century. It is not necessary to develop precise models of the reality with respect to the exact task; instead, a computer is modeling the relationship between environment and the available information about it. Different approaches to create this model, such as Gaussian process regression [14] or support vector machines [15], has been applied to solve the problem. Adapted to the human behavior, the investigation of the unknown environment with reinforcement learning (RL) is performed even autonomously, so that the robot is able to learn the process through trial and error [16]. A robotic system with a force controller can be used in combination with a so called Q-Learning-framework (artificial neural network with long-short-term-memory-layers) to maximize the flexibility of the assembly [17]. More complex strategies that utilize DDPG-algorithms (deep deterministic policy gradient) together with supervised learning show good results, but an elaborate set-up is necessary [18, 19].

Since the traditional approach for the peg-in-hole problem of developing a control scheme based on force/torque information is time-consuming for humans and needs to be adjusted for each task specifically, other methods such as LfD or RL are interesting for the use in industrial environments because of their flexibility. Especially the application of RL, which does not require the interaction with a human after the setup is complete, offers great potential even for assembly processes with tight tolerances. Regarding the joining process of the rotor for the turbocharger, it is necessary to save cost by using a traditional industrial

robot without integrated sensors or a cost-intensive force controller.

The objective of this paper is to investigate, if it is possible to develop a simple and cost-effective system that is able to join rotor components of a turbocharger. For this, we focus on RL in order to achieve a high accuracy as well as a great flexibility, so that the initial configuration is simplified and a deviation from the positioning of parts does not result in a failure of the assembly, which would lead to undesirable delays and unusable components.

2 Methodology

As shown in the previous chapter, reinforcement learning has a great potential for the flexible assembly of parts with small clearance by a conventional industrial robot. Therefore, an RL-supported method for assembly and an application framework are developed and presented in the following sections.

2.1 Fundamental strategy for peg-in-hole process with tight tolerances

To investigate the joining of the peg into the hole, the process is divided into two steps: search and insertion (Fig. 1). This subdivision is made so that the objectives of adjustments are clearly separated from each other: During the searching, the peg is centered in x- and y-direction above the hole. This is independent of the starting position. The insertion ensures an alignment of the peg's orientation so that it is in accordance with the orientation of the hole. The deeper the peg is inserted, the more precise the alignment must be in order to move the peg downwards without interfering points of contact.

Fig. 1 The joining process is divided into the search with the horizontal alignment (left) and the insertion part with the alignment of the orientation (right)

For the use of a conventional industrial robot with a position repeatability that usually lies between 40 and 60 μm , the described method is a great opportunity to be able to successfully join the components. With reinforcement learning it is possible to minimize the required human interference for configuration and teaching of the robot cell, so that the robot learns to develop a strategy for the assembly on its own. Regardless of the two different steps, for both a similar structure of the reinforcement learning process is applied in order to keep the system simple.

2.2 Application of Q-Learning as method of reinforcement learning

Reinforcement learning is a model-free method to solve different tasks by independently learning a strategy to maximize a reward. The aim is to learn the relation between states and actions by predicting the influence of an action a_t at a time t on a system. The belonging general principle is displayed in Fig. 2.

After a_t is executed, the state s_{t+1} of the system is observed by the supervisor and transferred to an agent, who computes a reward r_{t+1} to rate the action-state pair. The policy chooses the following action to maximize the cumulated reward. For example, it is possible that a chosen action in the beginning of the process leads to a greater cumulated reward than another action with a greater (current) single-reward. This principle itself is based on the Markov-Decision-Process (MDP), a model for decision-making problems, which means that the states are independent of each other and the environment is completely detectable. The solution of the MDP leads to a policy function.

2.3 Mathematical background of Q-learning

Q-learning focuses on the computation of the quality of the action-state pair and aims to maximize it. The optimal policy π^*

$$\pi^*(s) = \arg \max_a Q^*(s, a) \quad (1)$$

decides, which action is the best to maximize the Q-value. This optimal Q-value

Fig. 2 General principle of reinforcement learning

$$Q^*(s_t, a_t) = R(s_t, a_t) + \gamma \max_{a \in A} Q(s_{t+1}, a_{t+1}) \quad (2)$$

is composed of the expected reward R and the maximum prospective Q-value based on the next state-action-pair, which is reduced by the discount factor γ (initial steps greater impact than later steps). The update of the Q-function

$$Q^{\text{upd}}(s_t, a_t) = (1 - \alpha)Q(s_t, a_t) + \alpha \left[r(s_t, a_t) + \gamma \max_{a \in A} Q(s_{t+1}, a_{t+1}) \right] \quad (3)$$

is the core of the related algorithm and can be interpreted as the weighted average between the previous $Q(s_t, a_t)$ and the new Q-value with α as the learning rate. Since the state space of applications such as peg-in-hole is non-determined regarding the position and orientation, it is necessary to define a function that maps the Q-values. A simple solution is the use of an artificial neural network (ANN). In the feed-forward network, the state parameters are assigned to the input values and the ANN calculates the Q-values for each discrete action as the output parameters. The training is performed by backpropagation.

3 Implementation of algorithm for peg-in-hole

Algorithm 1 shows the pseudo code of the utilized Q-learning algorithm. As already mentioned in the previous section, the same algorithm is applied for searching and insertion, however the parameters and the ANN differ.

Algorithm 1: Q-learning

```

Initialize ANN
Initialize RL-Parameter:  $\alpha, \gamma$ 
for i ← 0 to #Episodes do
     $s_0 \leftarrow \text{ReceiveCurrentContactState}()$ 
    for t ← 0 to #Steps do
         $a_t \leftarrow \text{ChooseActionMotion}(s_t, \text{ANN})$ 
        based on Q-function
        SendAction( $a_t$ ) to robot
        ExecuteAction()
         $s_{t+1} \leftarrow \text{ReceiveNewContactState}()$ 
    end
     $r \leftarrow \text{ComputeReward}(s, a)$ 
     $M_i \leftarrow \text{SaveData}(s, a, r)$ 
    ResetProgram()
end
UpdateQValues( $M, \text{ANN}$ )
  
```

First, the ANN is initialized with randomly chosen Q-values and weight parameters. Then, the outer loop runs through episodes, these include a previously defined number

of trials of a complete searching/insertion procedure. The contact state is obtained from a force/torque sensor and saved as the current state. The instructions in the inner loop are executed as long as the maximum number of steps is not exceeded. For each step, the ANN is applied with the state parameters as input. The system executes the chosen action, which leads to a new contact state. After each episode, the reward that depends on every state and action is computed and all information are stored in memory. Finally, after several searching/insertion processes, the ANN is trained with the help of the memory.

3.1 Application of Q-learning in present assembly process

The Q-learning algorithm is based on the evaluation of a state-action pair, which suits well for the presented joining process, because an evaluation of the success is possible. It is intended to use the same structure with independently trained networks in the background, so that the effort of implementation can be reduced. The application of the algorithm is displayed in-depth in Fig. 3 and explained in the following paragraphs.

A fully connected ANN (Fig. 3, top) is suggested for the presented Q-learning, because after each action a new state follows immediately. Since the output of the Q-learning is discrete, the number of actions, which can be selected, is limited. These actions are position controlled, which means that a motion is performed based on the drive information. In this case, eight actions (equally split into each direction) are defined: for searching a change in x- and y-direction, for insertion a change in the orientation.

During the application of the peg-in-hole-process (Fig. 3, left), the starting positions are randomly chosen

in the beginning to simulate small deviations regarding the positioning of the components in the gripper or the mounting during the production of the turbocharger and also to simulate inaccurate positioning of the robot. Then, the robot moves stepwise downwards until a specific defined force is reached. Because of using a position control instead of a compliant force control, force needs to be applied stepwise and carefully to avoid damages. Force/torque information are transferred to the ANN as the current state. For the search process, it is intended that the network has two inputs (torques at x- and y-axis). The network for the insertion, however, has four inputs (torques and forces at x- and y-axis) because a more precise contact state evaluation is necessary due to potential jamming. The action is selected based on the Q-values. After the execution of the chosen action, the cycle starts again. This process breaks, if a threshold for the position of the hole or a maximum number of steps is reached. The latter leads to a failure of the joining process and the application must be repeated, starting with another initial position.

After a specific number of episodes, the data for each step (force, torque, position, action, reward) are saved in a global memory (Fig. 3, bottom), from which the training algorithm selects random data points to avoid dependencies among each other. The training (Fig. 3, right) is provided by application of backpropagation based on the true Q-values and the size of the reward. After saving the new parameters, the application of the ANN starts again. The training steps are repeated even if good results regarding the searching or insertion are already obtained to be able to react to changing circumstances of the environment such as a new batch in production or changing machine parameters.

Fig. 3 Details of the applied reinforcement algorithm for assembly process

3.1.1 Exploration of the environment

The main advantage of RL is the autonomous exploration of the environment. It is the aim to have a high exploration rate in the beginning to discover state space sufficiently. However, the rate should be increased later on, so that the system applies, what was learned before. Therefore, a decay factor γ was introduced in the algorithm, which reduces the exploration rate ϵ stepwise (starting with $\epsilon = 1$). This describes the probability for a random action to be applied instead of the suggested action. The differentiation is implemented by two cases

$$\epsilon = \begin{cases} \epsilon \cdot \gamma, & \epsilon > 0.1 \\ 0.1, & \epsilon \leq 0.1 \end{cases} \quad (4)$$

so that a minimum of exploration is still given in case that the circumstances changes during the application process.

3.1.2 Reward computation

The computation of the reward has a significant influence on the success of the Q-learning. It is not possible to assign a reward for each step immediately, because it strongly depends on the fact, if the searching or joining is successful, which leads to a reward between 0 and 1. Otherwise, it becomes negative. The total reward

$$r_{\text{Total}, t_{\max}} = 1 - \frac{n}{n_{\max}} \quad (5)$$

is calculated in the end of an episode for the last step $t_{\max}$ with the help of the needed number of steps n and the maximum number $n_{\max}$. Then, the result is computed back by multiplying the reward of the next step $t + 1$ by a factor p , so that the total reward

$$r_{\text{Total}, t} = r_{\text{Total}, t+1} \cdot p \quad (6)$$

is assigned to each step. For the search, the step reward

$$r_{\text{Step}, t} = r_{\text{Total}, t} + T_{\text{Step}} \cdot 2 \quad (7)$$

is also charged with the depth T_{Step} to reward the steps that lead to an insertion. If the assembly is not successful, the reward depends on the distance between the peg in the end and the hole. Instead, the step reward is equal to the total reward per step for the insertion.

4 Experimental evaluation

The assembly of a peg into a hole with small clearances is a great challenge because of the required accuracy and the high demands regarding the investment/running costs that result from the large-scale production of the turbocharger.

Table 1 Overview of the dimensions of the different holes

Hole	Tolerance	Diameter
1	22	–
2	21	–
3	20 D10	20.097 mm
4	20 F8	20.033 mm
5	20 H7	20.012 mm
6	20 M7	19.994 mm

Figure 4 a Anti-collision device, b force/torque sensor, c mounting, d peg, e holes with increasing diameter

Moreover, the duration of the process must be as short as possible. The experimental setup to validate the developed method in order to meet these requirements are presented in the following sections.

4.1 Setup

For this experimental evaluation, the parts have been generalized by a peg with a tolerance of 20h6 and a panel with holes of different sizes (Table 1).

To avoid additional structures for the joining, a conventional industrial robot is used (ABB 2400 16, controller: S4C Plus M2000, position repeatability $\pm 60 \mu\text{m}$), it is able to grab, handle, transport and mount the parts. As shown in Fig. 4, the setup also contains an end effector, which includes a 6-axis force/torque sensor. Furthermore, an anti-collision device is used to avoid damages of the sensor.

The structure of the communication system between the components is shown in Fig. 5. The main part of the control system is running on an external computer in Python. The reinforcement learning framework is programmed with Keras running on top of tensorflow. The control system is connected to the robot controller via FTP (file transfer protocol) to send the calculated offset and to receive the current position as text files. On the robot controller, a Rapid-program controls the motion and the communication. The anti-collision device is included into the automatic stop circuit. The force/torque sensor is attached to the measuring amplifier, which is connected to the controller via a digital 24 V port to forward a trigger signal in case of overloading. Because of the renunciation of the cost-intensive force-controlling unit and the limited communication speed, it is a step-by-step process. There is always the request for the measuring data on the one hand and the waiting for a new motion command on the other hand, which are alternating.

4.2 Preliminary studies

Before running experiments for the assembly process, a preliminary study is conducted in order to prove that reinforcement learning is applicable for the previously presented setup.

It is essentially relevant for machine learning to get to know the underlying data so that it is possible to estimate if the ANN can learn a functional relation. Therefore, it was tried to figure out if there is a relation between a random starting position and the belonging measured torque. The third hole with a clearance of 97 μm is used and starting

position lies in a radius from 1 mm around the center. The robot moves downwards until a contact is detected and the torque in x- and y-direction reaches $|1.5 \text{ Nm}|$. At the starting position, the peg is already placed on the chamfer of the hole because there is an explicit point of force transmission.

The results can be seen in Fig. 6. The data points on the left belong to the data points with the same shade on the right.

It is obvious that there is a relation between the left side and the right side regarding the order. A similar relation can be observed for a random starting orientation and the belonging torque state. These results show that there is a relationship between contact state and the current torque, which can be learned by an ANN.

4.3 Search

For the searching process, the experiments are conducted with the following scheme: The robot makes a contact between the peg and the edge of the hole, measures forces and torques, selects an action based on the output of the ANN (horizontal motion) and applies the action. Then, the cycle starts again until the maximum number of trial is exceeded or the searching is successful. The entire searching process (one episode) is repeated 10 times, afterwards the ANN is trained based on the collected data and the searching process is conducted again 10 times and so on. For each episode, another random starting position is chosen, so that the motion parts towards the center of the hole is different.

The graph of 10 exemplary horizontal paths in Fig. 7 visualizes the motion of the peg by the robot. The center

Fig. 5 Logical structure of the setup and communication channels

Fig. 6 Results of the preliminary study. Random starting position (left) and related torque with the same color-code as for the starting position (right)

Fig. 7 Exemplary paths of the horizontal alignment for the search process. The circle with a radius of 1 mm indicates the area of the starting position around the center of the hole at $x=0$ mm and $y=0$ mm

of the hole is assigned to (0,0), the gray circle indicates the random starting position around the center. The motion increment is fixed at 0.4 mm. As before, the position information are obtained by the robot controller, so the resolution might influence the results. The results show that the motion increments varies between 0.4 and 0.2 mm. Moreover, the movement is mostly not immediately aligned towards the centers. Reasons can be random actions, wrong decisions by the ANN or incorrect measurements.

Fig. 8 Total and average reward of the searching process with hole 3 (clearance of 97 μ m)

4.3.1 Results of search process

The results of the peg-in-hole searching process with hole 3 are indicated by the reward, with which it is possible to evaluate the success. Figure 8 shows that the training is effective since the first training after 40 episodes with 200 data points, the training steps follows each 10 episodes. The higher the reward, the better the process works. The total reward varies between -1 and 0.875 from the first to the 50th episode, after episode 70 it is always greater than 0.55 . The average over 30 values is constantly greater than 0.8 during the last episodes, so that the training for the searching process is successful. However, the searching process fails when joining the peg into the hole with the next smaller

clearance of 33 μm . Because of the data received from the force/torque sensor, the controller can estimate the position of the hole. But the robot is not able to move in such small incremental distances that would be necessary. This is independent of the positions repeatability. Moreover, the evaluation of the reward becomes more complicated. Currently, the computation of the reward is based on the robot's position data, which could be inaccurate because of the manipulator's compliance or the measurement resolution. Further experiments show that it is also possible to apply a pre-trained network for hole 2 to hole 3. This discovery was also transferred to the next transition from hole 3 to hole 4, but the robots still fails to align the horizontal position of the peg correctly.

4.4 Insertion

The experiments for the insertion follow a similar scheme as for the search: The cycle consists of making contact between peg and hole by moving downwards, measurement of the torques and forces, realignment of the orientation in x- and y-direction. This is also repeated until a maximum number of trials is reached or the insertion is successful. One complete insertion process is referred to as one episode. It is performed 10 times, then the training step follows.

4.4.1 Results of insertion process

The reward for the insertion process is displayed in Fig. 9, bright background for hole 2 and darker background for hole 3. The average reward for hole 2 is increasing almost constantly from 0.17 to 0.75, a positive effect of the training can be seen, so the adaptation of the orientation is successful. After the successful strategy of applying a pre-trained network to another hole, the same method was chosen for the insertion. However, the graph shows a decrease of the total and hence for the average reward. The total reward is always smaller equal to 0.1, which means that the process was not

Fig. 9 Total and average reward of the insertion process with hole 2 and hole 3 (gray background)

successful. Further experiments without newly initialized network show the same result: It is not possible with the current setup to insert the peg into the hole with the clearance of 97 μm . The probable reason is an ambiguous state of contact, which leads to different force and torque measurements. The incremental alignment of the orientation could have been too large for this clearance, so that a motion leads to jamming and the peg gets stuck with more than one point of contact with the hole. This cannot be predicted on basis of the sensor data, it can only be detected, when the ambiguous state of contact is already present.

4.5 Findings and limitations

The aim of the presented study was to figure out if it is possible to use a simple and low-cost robotic setup in combination with reinforcement learning to join rotor components of a turbocharger. The here presented results of the search and insertion process for joining abstracted rotationally symmetric components with small clearance can be transferred to the real assembly. The data show that it is currently not in every case possible to improve the robot's position repeatability of $\pm 60 \mu\text{m}$, in the sense that the presented system outperforms the position controller itself. The searching fails at hole 4 with a clearance of 33 μm and the insertion at hole 3 with a clearance of 97 μm . Currently, the process takes too much time; it lasts between 1.5 and 2 min for a successful search or insertion episode. Nevertheless, with changing the communication protocol the duration should decrease drastically. The data transfer via FTP, which is the only interface between our controller and computer, is time-consuming due to the processes of successively opening, writing, saving and closing files. With TCP/IP or UDP the transport can be accelerated since these protocols establish a data stream instead of sending files, but the workflow is identical to FTP.

The comparison between the results presented here and the achieved accuracies from the state-of-the-art shows that the results are in the same range [18, 19]. However, some researchers can present a joining process with smaller clearance, e. g. [17], successfully joined peg and hole with just 6 μm of clearance. They use an internal force controller for an external or internal force/torque sensor. Sometimes the controller is supported by a vision system. Instead, in this case, a common position controller with a cheap and rudimentary external computing was applied for reasons of simplicity. The possibilities with this simple setup are quite limited, so that the application of a complex system with complex optimization strategies is not reasonable. Instead, the presented system is straightforward to set up and to use, therefore, it is not necessary to invest much time for integration in contrast to [19]. Our method is supported by the use

of a conventional position controller, thus, the method can be applied even for older robots.

The accuracy for the presented joining process (search and insertion) received with the help of the force/torque sensing system is currently still lower than the accuracy obtained by the pure position controller with the position repeatability of 60 μm . But, the results show that it is able to eliminate deviations regarding the position and orientation and thus increase the flexibility of the production system.

5 Conclusion and outlook

This paper presents the experimental study of reinforcement learning for the assembly process of a turbo charger rotor. A setup with a conventional industrial robot in combination with an external force/torque sensor is introduced to learn the joining process with prototypical rotor parts. The results of the searching process show that the horizontal alignment (search process) of the peg in the center of the hole with a clearance of 97 μm is possible, whereas the insertion process fails at this fitting due to ambiguous contact states. This can probably be optimized by reducing the length of the incremental step, which was not possible in this setup due to the controller restrictions. Because the usual position control of the robot is used, this strategy is easily transferable for many (even older) systems, which enables a technology transfer to new markets.

In the future, the assembly approach shall be improved by the application of a vision system which is a promising supplement to the existing experimental setup, in order to achieve even greater accuracies and mainly to optimize the insertion process. With a high-resolution camera, the position of the hole and the peg can be determined at the same time, so that jamming can be predicted before the force/torque sensor detects it.

Acknowledgements Open Access funding provided by Projekt DEAL. This paper is part of the research project “ARIEL”, which is funded by the Federal Ministry of Transport and Digital Infrastructure, Germany in course of National Innovation Program for Hydrogen and Fuel Cell Technology (Grant No. 03B10105D2).

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Availability of data and material Yes.

Code availability Yes.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long

as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. International Energy Agency (2018) World Energy Outlook 2018
2. Sammes N (2006) Fuel cell technology. Springer, London
3. Venturi M, Sang J, Knoop A et al (2012) Air supply system for automotive fuel cell application. In: SAE 2012 world congress & exhibition
4. Laktionev N, Andreev G (1966) Automatic assembly of parts. *Russ Eng* 46(8):40
5. McCallion H, Wong PC (1975) Some thoughts on the automatic assembly of a peg and a hole. *Ind Robot* 2(4):141–146
6. Jasim I, Plapper P, Voos H (2014) Position identification in force-guided peg-in-hole assembly tasks. *Proc CIRP* 23:217–222
7. Luo RC, Chang A, Li C (2017) A novel peg-in-hole approach based on geometrical analysis for inclined uncertainty. In: 2017 IEEE international conference on advanced intelligent mechatronics. IEEE, Piscataway, pp 891–896
8. Abdullah MW, Roth H, Weyrich M et al (2015) An approach for peg-in-hole assembling using intuitive search algorithm based on human behavior and carried by sensors guided industrial robot. *IFAC* 48(3):1476–1481
9. Chen H, Zhang G, Zhang H et al (2007) Integrated robotic system for high precision assembly in a semi-structured environment. *Assem Autom* 27(3):247–252
10. Chen H, Wang J, Zhang G et al (2009) High-precision assembly automation based on robot compliance. *Int J Adv Manuf Technol* 45(9–10):999–1006
11. Park H, Park J, Lee D-H et al (2017) Compliance-based robotic peg-in-hole assembly strategy without force feedback. *IEEE Trans Ind Electron* 64(8):6299–6309
12. Tang T, Lin H-C, Tomizuka M (2015) A learning-based framework for robot peg-hole-insertion. In: Proceedings of the ASME 2015 dynamic systems and control conference: Columbus. American Society of Mechanical Engineers, New York, pp 1–5
13. Ehlers D, Suomalainen M, Lundell J et al (2019) Imitating human search strategies for assembly. In: 2019 IEEE international conference on robotics and automation (ICRA). IEEE, Piscataway, pp 7821–7827
14. Li B, Cheng H, Chen H et al (2014) Modeling complex robotic assembly process using Gaussian process regression. In: Proceedings of the 2014 9th IEEE conference on industrial electronics and applications. IEEE, Piscataway, pp 456–461
15. Jakovljevic Z, Petrovic PB, Hodolic J (2012) Contact states recognition in robotic part mating based on support vector machines. *Int J Adv Manuf Technol* 59(1–4):377–395
16. Kober J, Bagnell JA, Peters J (2013) Reinforcement learning in robotics: a survey. *Int J Robot Res* 32(11):1238–1274
17. Inoue T, Magistris G de, Munawar A et al (2017) Deep reinforcement learning for high precision assembly tasks. In: 2017 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, Piscataway, pp 819–825
18. Fan Y, Luo J, Tomizuka M (2019) A learning framework for high precision industrial assembly. In: 2019 IEEE international

- conference on robotics and automation (ICRA). IEEE, Piscataway, pp 811–817
19. Ren T, Dong Y, Wu D et al (2018) Learning-based variable compliance control for robotic assembly. *J Mech Robot* 10(6):41015

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.