Available online at www.sciencedirect.com

ScienceDirect

Procedia Manufacturing 51 (2020) 11–18

Procedia MANUFACTURING

www.elsevier.com/locate/procedia

30th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2021)
15-18 June 2021, Athens, Greece.

A Machine Learning Approach for Collaborative Robot Smart Manufacturing Inspection for Quality Control Systems

Thadeu Brito[a,*], Jonas Queiroz[a], Luis Piardi[a], Lucas A. Fernandes[b], José Lima[a,c], Paulo Leitão[a]

[a]*Research Centre in Digitalization and Intelligent Robotics (CeDRI), Instituto Politécnico de Bragança, Campus de Santa Apolónia, 5300-253 Bragança, Portugal*
[b]*Federal University of Technology - Paraná (UTFPR), Curitiba, Brazil*
[c]*INESC TEC - INESC Technology and Science, Porto, Portugal*

Abstract

The 4th industrial revolution promotes the automatic inspection of all products towards a zero-defect and high-quality manufacturing. In this context, collaborative robotics, where humans and machines share the same space, comprises a suitable approach that allows combining the accuracy of a robot and the ability and flexibility of a human. This paper describes an innovative approach that uses a collaborative robot to support the smart inspection and corrective actions for quality control systems in the manufacturing process, complemented by an intelligent system that learns and adapts its behavior according to the inspected parts. This intelligent system that implements the reinforcement learning algorithm makes the approach more robust once it can learn and be adapted to the trajectory. In the preliminary experiments, it was used a UR3 robot equipped with a Force-Torque sensor that was trained to perform a path regarding a product quality inspection task.

Keywords: Collaborative Robots; Quality Control Systems; Reinforcement Learning; Human–robot Interaction; Actor-Critic; Robot Learning

1. Introduction

Collaborative robots, or Cobots, are becoming more and more used since these robots can interact with humans in a shared space according to ISO 15066:2016. Example of that are industrial manipulators that are designed to handle, process or manipulate the material. On the one side, the human has the ability of a hand that can perform thorough work, with high degree of dynamism. On the other side, the robot is well-known on its excellent repeatability, force and indefatigable. Combining both strengths is the main objective of the Cobots applications. Due to the collision detection and avoidance based on joint motors and sensors, humans and robots can work side by side, getting the best out of each world. This approach is one of the most topics addressed by the Industry 4.0 concept, and a massive number of manufacturers is accepting this technology.

One of the benefits that Cobots bring to the industrial community regards its programming that for many tasks can be easily achieved without write any line of code, instead the robot can be taught by guiding it along a given path. If several years ago, the manipulators require expert programmers to develop the code for the robot, nowadays this task is facilitated since the user can move the robot arm manually, just like a teaching method for a new person. The main Cobot manufacturers allow programming the robot in this way, like a teaching process. Different Learning from demonstration (LfD) methods can be done with robotic manipulators, evidenced by [1], however, each of these methods requires an adaptation to the work that will be demonstrated. For example, using the Kinesthetic method, the robots can learn by primitive motor actions. On the other hand, if the objects being manipulated are large, distant or dangerous, Kinesthetic orientation can be problematic [1]. Moreover, there is still a gap in this approach once the robot controller uses all joint torque information to handle the teaching movement, and it stores the posture of the manipulator in the desired positions of the end-effector. It is truth that adding a force sensor to the end-effector helps this teaching procedure (since the controller doesn't need to estimate the force that the user is applying to the robot). Moreover, there are some cases that it is crucial to store

* Corresponding author. Tel.: +351-273-303-200 ; fax: +351-273-325-405.
E-mail address: brito@ipb.pt (Thadeu Brito).

10.1016/j.promfg.2020.10.003

the path between way-points. An example of this case is the quality control inspection with a camera assembled to the manipulator body, this being an application that requires dynamic trajectories performed by the manipulator, which is difficult to be predicted by the Cobot programmer.

In this context the presented paper addresses a machine learning methodology that, attached to a collaborative manipulator, enables the human that share the space with the Cobot to teach the desired path according to the task requirements. This allows for the continuous and dynamic learning of Cobot, without the need to stop the production process so that Cobot adapts itself to another trajectory in a dynamic way, resulting in a productivity boost. The preliminary results demonstrate that the algorithm enabled the robot to perform the path indicated by the operator through iterations with the force sensor attached to the Cobot tool, and activities such as, for example, quality control, can benefit from this approach.

This paper is organized as follows: after a brief introduction, section 2 presents the related work that stresses the reinforcement learning, the Actor-Critic and the Deep Q-Learning algorithms. Section 3 presents the system architecture where the Universal Robot UR3, the 6 DoF sensor and the Learning strategy are exhibited. The results are presented and discussed on section 4, whereas section 5 rounds up the paper with conclusion and points out the future work.

2. Related work

Smart production is just one of the topics addressed to Industry 4.0, through these new concepts the industry can leverage its production, and also take a step ahead of its competitors in the fight for the leadership of the global market [2]. The digitization of an entire production line changes the value-added of the final product and also the working environment. Among the various technological changes, and the impacts of the new industrial paradigm pointed out by [3], what stands out is the interaction of human with the machine in Cyber-Physical Systems (CPS). This type of interaction promises to bring to the inside of the factories a series of economic and social opportunities, work organization, business models and others. Therefore, this industrial trend also has a substantial impact on academic research, in which researchers try to find innovative and creative solutions applying the entire theoretical concept acquired over the years.

Aware of the impacts that this transformation can trigger, in [4] is demonstrate the potential of these global machine networks in the shop-floor environments. Considering the warehouse, transport and logistics, procedures and fulfilment functions, machines can work autonomously and fully collaboratively between humans and also between themselves. These tasks are only possible when all sensors, devices, machines and other enterprise assets are connected and fully synchronized. In [5], there is a necessity to convert conventional machines (those that are already working in factories) into self-aware and self-learning machines. In this way, the companies can update in the trends of the new industrial age, and consequently, remain competitive in the global marketplace.

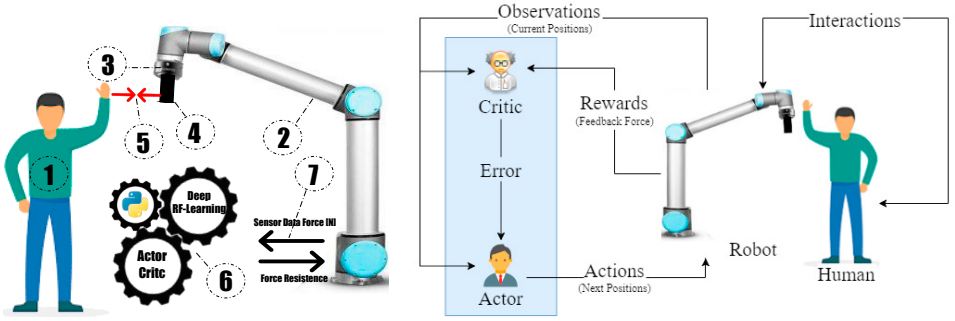
The use of industrial robotic applications in the handling of tools or workpieces is an established practice in the industrial sector [6]. However, many of these applications cannot work adapting to the events that may arise. In other words, many of the applications inherited from the third industrial revolution are configured to be repetitive and fixed, which makes them practically inflexible in terms of action (or motion) changes. Described by [7], collaborative robots with learning techniques can enhance even the most complex and stable markets, such as the food industry, automotive, textile, among others. Integrating human flexibility into robotic applications with automation efficiency has several approaches and techniques, among which the Reinforcement Learning (RL) algorithms stand out.

2.1. Reinforcement Learning

Making collaborative robots capable of learning how to move to pour liquids without knowing the final objective, that is, free of fixed movements, in [8] reports two RLs methods that exemplify the process of recognizing the object's position and approximate the tool to the endpoint. In this way, it is possible to identify the object shape quickly and make the best decision to move dynamically or pour the liquid. Applying RL is not always a simple approach, it is necessary to consider a series of policy conditions, mainly for industrial manipulators with the configuration of their joints in series. A work that addresses the learning of the manipulator's trajectory by RL through individual instructions is described in [9], where the approach uses virtual reality sensors to instruct the robot's movements, different from the method described in this paper that uses touches of the human in the force sensor to guide the manipulator. In [10], two policy search algorithms are presented to make RL in industrial manipulators more applicable, that is, without the need for complex sensors. Through policy representations by Dynamic Movement Primitives (DMPs), the learning based on banned actions that the robot should not learn. Hence, avoid obstacles in the work environment while still calculating paths within the established restrictions. It is also possible to determine the RL by reference methods, demonstrated in [11], where the manipulator must detect as quickly as possible deviations that the applied model has concerning the environment model. Therefore, inspect/correct robots may be able to learn what to do according to the tasks of the scenario. Such situations, where manipulators configured with RL are used to control the quality of products through inspection, may have an infinite number of input spaces, and consequently, will generate an infinite number of output actions. As a solution to this problem, the Actor-Critic Algorithm could be a suitable approach to improve the robot performance and enable it to continuously adapt to dynamic environments subject to a variety of possibles change and tasks like those envisioned in the 4^{th} industrial revolution mass customization production paradigm.

2.2. Actor-Critic

To overcome the uncertainties and spaces for continuous state-action spaces, policy gradient methods can be applied.



(a) General System Architecture.

(b) System Algorithm Architecture.

Fig. 1: System Architecture of proposed approach.

This method has the technique of creating a baseline together with the RL, making the algorithm dependent and with slow convergence. Through a general algorithm to estimate the natural gradient, it is possible to generate learning in robots by temporal difference or start-state reinforcement learning [12]. This general algorithm works with two methods, the Critic which, in a way, monitors the research choices made by Action. However, generating actions for manipulators using this method requires a study of the building block of movement generation, commonly called Motor Primitives [13]. Demonstrated by [14], motor primitives can be an initial point for the robot to start its learning, as exemplified in the task of imitating a baseball player. The anthropomorphic manipulator manages to hit the ball correctly (hit the point so that the ball reaches the greatest possible distance) after some movements that imitate the motor primitives inserted.

The Actor-Critic can be refined through feedbacks inserted by a human, which makes the control of the robot more simplified. When using angular increments or decrements techniques, the system developed by [15], selects new actions that are verified by the critic through the analysis of the error generated by the reward (TD-Error). In this way, robotic agents can be trained by signal impulses from humans. The RL techniques with Actor-Critic can also be inserted directly into the controller of a manipulator, demonstrated by [16], where the feedback signal from the controller of a UR5 manipulator is compensated by learning an Actor-Critic without having to generate the system model. It can also be used in cases of observability, where the actor obtains images from a camera as input and the critic is trained through states connected to a 3-layer neural network. Thus, it is possible to train the manipulator in simulated mode and then transfer the learning to the real robot which eliminates the dangerous and expensive process that deep reinforcement learning can cause [17]. In another real case of industrial manipulator performing processes in food manufacturers with Deep Reinforcement Learning is addressed by [18], where the robot needs to identify the correct locations to inject brine into the bacon pieces. It is possible, by applying the techniques of Deep Deterministic Policy Gradient (DDPG), which is based on the Deep Q-Networks (DQN), both uses two neural networks to estimate the action-value function [19].

3. System Architecture

In order to teach the robot the necessary path to perform a collaborative activity with the human, it was developed the system architecture shown in Figure 1. The hardware used in this work integrates the necessary equipment to teach and communicate with the manipulator. The software, based on RL, aims to enable the robot to learn the path desired by the user through positive or negative reward policies.

The system used can be summarized in 7 different parts represented in Figure 1a. The first part (1), denotes the human responsible for collaboratively guiding the manipulator through the desired trajectory within the $\mathbb{R}^3$ (Cartesian space) acting on the robot tool. The operator will indicate to the robot, through movements using end-effector, the path he/she wants the robot to take. In (2) is the robot itself, which is a collaborative manipulator. This manipulator is equipped with a Force-Torque sensor FT-300 (3) capable of measuring force in the X, Y, and Z directions of the Cartesian plane. Coupled with the force sensor (4) is a cone-shaped 3D printed tool that the user can hold and perform movements with the robot. This printed piece has high rigidity, able to withstand the force exerted by the user on the robot, this iteration of forces between the robot and the user is represented by (5). Then, (6) represents RL algorithms, which, depending on the movement performed by the human, measured by the force sensor, will recognize the trajectory that the manipulator must follow. Finally, (7) indicates the Modbus TCP communication between the robot and the computer running the application. The robot sends the force data applied to all axes, which is processed by the script, which will return the desired trajectory, resulting in a path without the need to use a human-machine interface or some level of programming, just through the collaboration between the robot and the user.

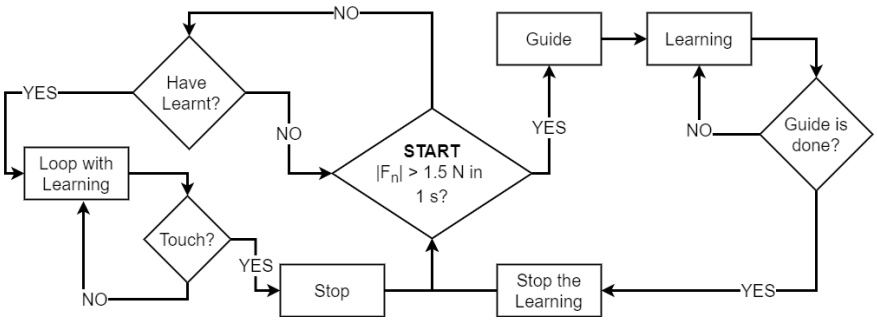


Fig. 2: The logic sequence of control and dynamic learning developed to perform quality inspections.

3.1. UR3 and sensor

The developed application is composed of several components. The manipulator robot that is learning through forces applied by the user refers to a Universal Robots UR3, a collaborative robot. Considering the characteristics of this robot, it represents a potential tool to assist an operator in assembling or inspecting a product. It is a small collaborative table-top robot for light assembly tasks and automated workbench scenarios, equipped with a control box and a programming user interface. The robot weighs 11 kg, with a payload of 3 kg, 360-degree rotation on all wrist joints, and infinite rotation on the end-joint. These unique features make UR3 a flexible, lightweight, collaborative table-top robot to work side-by-side with humans in a safe way.

Table 1: Specifications of the FT300 Force-Torque sensor

Feature	Value	Unit
Measure Range FX, FY, FZ	+/- 300	N
Measure Range MX, MY, MZ	+/- 30	N.m
Data Output Rate	100	Hz
Weight	0.3	kg

The UR3 robot is equipped with a Force-Torque sensor FT300 of 6 DoF. Table 1 presents the specifications of this sensor. The data FX, FY and FZ represents the measure of force in each direction. The variables MX, MY and MZ are the moments that can be measured.

3.2. Learning strategy

In inspection tasks to determine the quality of a product, unforeseen events can sometimes arise in the workspace of a manipulator's movements. Causing a series of problems and generating significant losses for industrial production. In this case, a user can guide the effector of the manipulator to teach which movement would be necessary to inspect each occasion. Using the sensor described in section 1, the operator can transfer his knowledge to the manipulator using only a single sensor, as shown in Figure 1b.

By reading the force data from the sensor and knowing the Cartesian position of the tool, the developed learning algorithm can collect information, and based on that, generates the points that the robot must inspect. Considering that the operating space is continuous, use a Neural Network represents a suitable solution to learn the states-actions in this space (i.e., a given position - a point in XYZ coordinate, or a path represented by a sequence of positions). The performed path will comprise a set of sequential points (position in XYZ axis) that will be repeated along the time, according to each kind of task. Based on that temporal dependency, the neural network can use LSTM (Long Short-Term Memory) neurons that are capable of learning temporal patterns and predicting events where it is needed to obtain a certain precision. Therefore, these data must be analyzed first by a neural network, since it is a dynamic environment and has multiple data (Fx, Fy and Fz). In other words, this means that the movement's prediction needs to be performed according to the previous actions, which is how manipulators perform their motions.

The structure developed in this work is controlled only by the touch sensor, that is, the operator guides the robot for new movements and doesn't configure it through the Teach Pendant. Or even when the operator wants the manipulator to stop or start the inspection previously learned. Thus, Figure 2 demonstrates in a flowchart format, the logic of control and dynamic learning developed. Where the starting point defines the robot's wait for some operator action on the touch sensor. At this point, the manipulator may be unlearned or just stopped. If the operator performs a constant force higher than 1.5 N and lasting at least 1 second, the learning algorithm is automatically activated. During learning mode, samples from inspection points are collected for 0.5 seconds. This sampling is only stopped when the operator stops to apply force on the sensor with a result of 1.5 N. Then, these samples are sent to the Neural Network to learn the movements that the operator taught, creating a baseline for the movements/path that the robot should perform, which should be improved with the Reinforcement Learning approach. As soon as the motions are determined, the algorithm returns to the initial state (waiting for force in sensor). At this moment, if the operator wants to make a new Guide, it is possible, and consequently, new learning is performed. But if the operator applies only one touch (resulting force high than 1.5 N and lasting less

than 1 second), the system checks whether there is any learning in the robot. If so, the manipulator starts to perform the learned inspection movement repeatedly until some stop force is applied to the torque sensor. If this happens, the manipulator return to the state's wait (starting point). There is still the possibility that the manipulator is in action waiting mode (commonly when is power-up recent), receiving only the touch and has not learned any inspection. In this case, the robot issues an alert and returns to the standby mode.

3.3. Control with Learning

In the "Loop with Learning", the algorithm will continuously collect the sensor information. Based on the current tool position, it will use the trained Neural Network to predict the next position and derive the distance that the robot needs to move. A `movel` command (move to position - linear in tool-space) is sent to the robot, and after its execution, the loop starts again. During this loop, the data from the torque sensor is also collected and analyzed in order to correct/improve the trajectory based on the user feedback. In this context, the Reinforcement Learning algorithm should be applied, where the next movement needs to include the predicted value plus an increment given by force applied by the user.

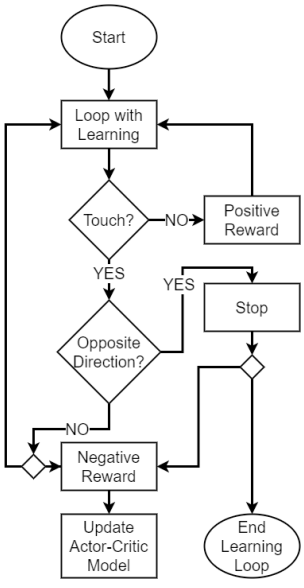


Fig. 3: Update Actor-Critic Model.

In this context, the Actor-Critic algorithm should follow the approach illustrated in Figure 3. The state that the Actor-Critic must receive is defined as the end-effector positions, that is, the Cartesian positions of the UR3 tool. Therefore, the Actor will determine the actions according to the generated distances (robot motion). It is up to Critic to evaluate the actions taken (the error) to receive the reward. Thus, it is essential to always update the Actor-Critic model while the manipulator is carrying out the inspection process.

The update of the Actor-Critic model must be done while the robot is inspecting the parts. Then, if the user touches the robot during its movement, and the applied force is not in the opposite direction of the movement, the next action should receive an increment, and the model should be updated. The update will consider the state that leads to the action that was being performed. The Critic model will be updated with a small negative reward, while the Actor model will be updated with the state incremented according to the applied force. The same happens when a force is applied in the opposite direction, but in this case, the robot should stop and wait for a guide (update learn) or touch (continue with the next step), and a big negative reward should be applied, while the state should be updated with a smaller distance to move in the given state.

4. Results

The preliminary experiments focused on the development of the main control and learning strategy (Figure 2) and its integration with the robot. In this context, the efforts focused on the development of the Actor Neural Network of the Actor-Critic Reinforcement Learning algorithm, i.e., the one that should be responsible for predicting the next action to be performed based on the current observed state.

4.1. Experiment Settings

In this experiment, the proposed system architecture was implemented using Python. The data processing and Machine-Learning was performed based on widely used libraries, like NumPy and Keras (TensorFlow as the backend). The interface with the UR3 robot was developed based on the URX library (from Universal Robots - https://github.com/SintefManufacturing/python-urx). Another interface was developed to connect and retrieve the data stream from the Robotic Force-Torque sensor (FT-300). However, the FT-300 provides six DoF, only three DoFs were used for this experiment: the force on the X, Y and Z axis to measure the force of human during the procedure of interacting with the robot to teach the path. In this sense, the end-effector orientation is fixed to avoid complex measurements from Force-Torque sensor. Although the robot is connected with an industrial PC, the prototype and all the experiments were executed in a external PC with Windows 10, i7-8750H, 16GB RAM, GeForce GTX 1050Ti.

The focus of the preliminary experiments is to develop the Actor model. In this context, a Neural Network was chosen to support the representation of the continuous space and action. Furthermore, given the temporal characteristics of the environment, i.e., the execution of a path requires to perform a sequence of move actions, it was chosen an architecture based on LSTM neurons. Table 2 describes the architecture used for the Actor model. This configuration was reached after some interactions, and does not represent a fine tuned choice for the complete problem solution.

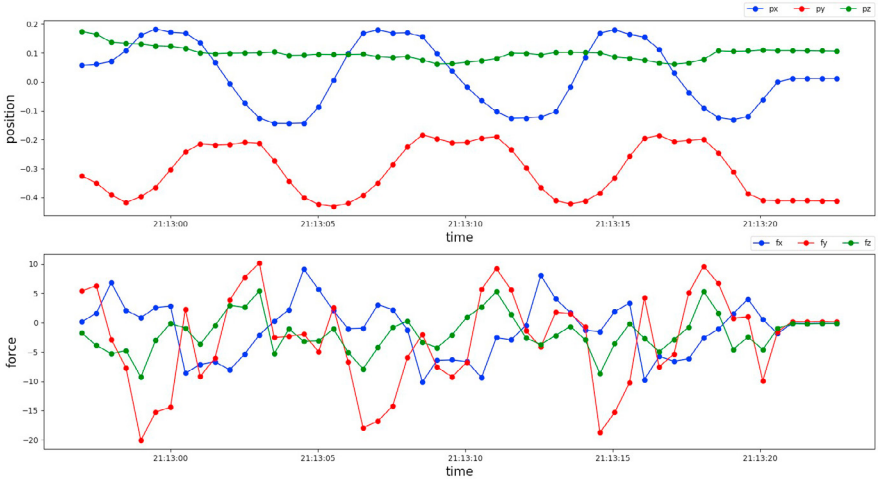


Fig. 4: Dataset obtained with a square-shaped path guided by a human. Position (m) and force (N).

Table 2: Specifications of the actor model neural network.

Feature	Value
LSTM layers	2
Optimizer	Adam
Learn Rate	0.001
Loss Function	Mean Squared Error
Input	3
Output	3
Neurons per layer	100
Epochs	100
Functions Activation	relu, relu,sigmoid

The test consists of simulating movements of a quality inspection task, through an operator moving the end-effector of the UR3 to any points. As shown in the Figure 5, the objective of the experiment is to train the robot to perform quality inspection tasks according to the points determined by the operator. For instance, an inspection task may consist in the robot to carry a product to be inspected in a given position, where there is an inspection equipment, then rotate this object to inspect different angles. The same object can be moved to another position, to another inspection equipment, and finally, show it to the operator for some final check.

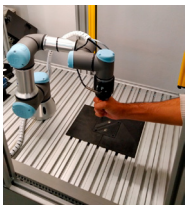


Fig. 5: Force training with the user holding the Force-Torque sensor.

4.2. Training baseline Neural Network

The approach in this work considers the possibility of the robot learning a path based on human supervision, instead of let the robot to find an optimal path by itself based on a set of goal points and unlimited number of trial-and-fail interactions. In this context, a baseline path should be provided by the user that will work with the robot in the same workspace. This means that, at the beginning, the robot will wait for the user to guide it to perform a path, one or few times, creating a baseline that the control algorithm will use during operations, while the learning algorithm will continuously improve based on user feedback.

In this situation, when the Guide mode (Figure 2) is enabled, the system starts to collect several points along the path. These points represent the states (i.e., the observations) that at the end of the Guide mode will be used to train the Actor model. Figure 4 (top) illustrates the collected points, where the robot is guided to perform a squared-shape path (3 times) along X-Y axes. The chart in the bottom of the Figure 4 illustrates the force applied in the three axes along the path and can be used to determine the move speed in different segments. Therefore, the blue line (movement guided on the X-axis) represents the points collected from one of the faces of the created square. In duality

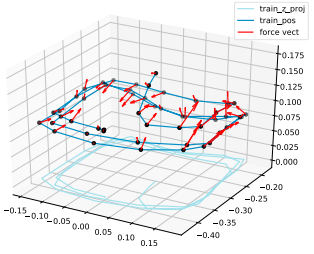


Fig. 6: The square-shaped path used for training.

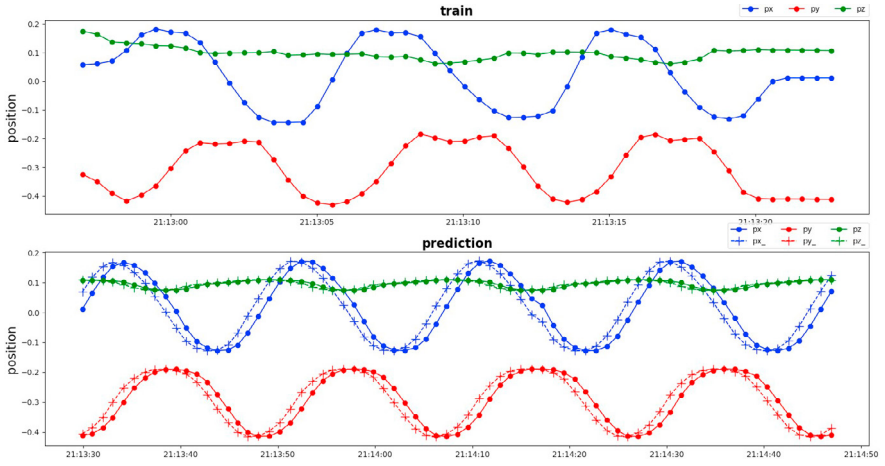


Fig. 7: Trained dataset and prediction results of the baseline model for a square-shaped path.

with this blue line, there is the red line (movement guided on the Y-axis). These two opposing lines demonstrate the configuration of the straight lines made by the user in the X-Y plane. Consequently, the green line does not have many variations due to the movement configuration not having components on the Z-axis. The movement's 3D path view representation is illustrated in Figure 6. In this figure, the path starts in the top-centre of the working space, where the robot is initially positioned. Then it is guided down to the corner, where the desired path is repeated three times. The figure also illustrates the force vectors applied during the guiding.

From the dataset created, the learning algorithm based on the LSTM Neural Network starts to process points for the UR3 to perform in the closest possible way to the movements that the user previously taught. These points are indicated with Cartesian coordinates, where the robot increments from one position to another. In this way, the `movel` command is configured with the distance from one point to the other. Figure 7 (bottom) illustrates the performance of the baseline model (along 4 interactions) after being trained with the dataset illustrated in Figure 7 (top). The dashed lines are the predictions performed based on the current state (solid lines, i.e., tool position). The poor performance showed to repeat the guided path is mainly given by the few amounts of samples used for the model training, as well as the need to fine tune the algorithm's parameters. However, the idea is with few supervised samples obtain an approximated model that can be dynamically improved, so the user does not need to repeat the complete path several times. It also enables the dynamic change of segments without the need to retrain all the path. The predicted movement's 3D path view representation is illustrated in Figure 8. There is illustrated that the trained model could approximate the training data points, leading to a path that smooth the path of the corner.

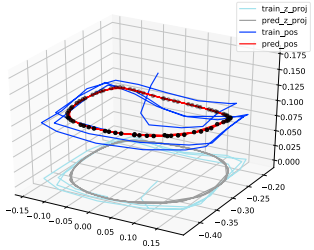


Fig. 8: The square-shaped path used for training and the predicted path.

5. Conclusions and Future work

This paper addressed a machine learning that attached to a collaborative manipulator enabled the user to teach the desired path with significant precision. Unlike the teaching system that is distributed with the robot, the developed system enables dynamic teaching and operation, where the user can interact and change the path on-the-fly. In this way, it allows the human to interact with the robot to guide it on a new path, as soon as unexpected situations arise during the quality inspection, using the force sensor attached to the robot. The reinforcement learning and the Actor Neural Network methodology proved preliminary solutions for the initial problem. It is also noticed at the results section that the proposed system is able to support the smart inspection and corrective actions for quality control systems in the manufacturing process, complemented by an intelligent system that learns and adapts its behaviour according to the inspected part. As future work direction, it is pointed to develop the Critic model and integrate it with the Actor, as well as to use the six DoF from the torque sensor instead of three. Finally, an implementation in a real environment factory should be implemented emphasizing the character of the problem.

Acknowledgements

This work has been supported by FCT – Fundação para a Ciência e Tecnologia within the Project Scope: UIDB/05757/2020.

References

[1] Z. Zhu, H. Hu, Robot learning from demonstration in robotic assembly: A survey, Robotics 7 (2) (2018).

[2] L. Merkel, J. Atug, L. Merhar, C. Schultz, S. Braunreuther, G. Reinhart, Teaching smart production: An insight into the learning factory for cyber-physical production systems (lvp), Procedia Manufacturing 9 (2017) 269 – 274, 7th Conference on Learning Factories, CLF 2017.

[3] A. Pereira, F. Romero, A review of the meanings and the implications of the industry 4.0 concept, Procedia Manufacturing 13 (2017) 1206 – 1214, manufacturing Engineering Society International Conference 2017, MESIC 2017, 28-30 June 2017, Vigo (Pontevedra), Spain.

[4] B. Tjahjono, C. Esplugues, E. Ares, G. Pelaez, What does industry 4.0 mean to supply chain?, Procedia Manufacturing 13 (2017) 1175 – 1182, manufacturing Engineering Society International Conference 2017, MESIC 2017, 28-30 June 2017, Vigo (Pontevedra), Spain.

[5] S. Vaidya, P. Ambad, S. Bhosle, Industry 4.0 – a glimpse, Procedia Manufacturing 20 (2018) 233 – 238, 2nd International Conference on Materials, Manufacturing and Design Engineering (iCMMD2017), 11-12 December 2017, MIT Aurangabad, Maharashtra, INDIA.

[6] T. Brito, Intelligent collision avoidance system for industrial manipulators, Master's thesis, Instituto Politécnico de Bragança, Escola Superior de Tecnologia e Gestão, http://hdl.handle.net/10198/19319 (2019).

[7] R. Accorsi, A. Tufano, A. Gallo, F. Galizia, G. Cocchi, M. Ronzoni, A. Abbate, R. Manzini, An application of collaborative robots in a food production facility, Procedia Manufacturing 38 (2019) 341–348, 29th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM 2019), June 24-28, 2019, Limerick, Ireland, Beyond Industry 4.0: Industrial Advances, Engineering Education and Intelligent Manufacturing.

[8] M. Tamosiunaite, B. Nemec, A. Ude, F. Wörgötter, Learning to pour with a robot arm combining goal and shape learning for dynamic movement primitives, Robotics and Autonomous Systems 59 (11) (2011) 910 – 922.

[9] N. Arana-Arexolaleiba, N. Urrestilla-Anguiozar, D. Chrysostomou, S. Bøgh, Transferring human manipulation knowledge to industrial robots using reinforcement learning, Procedia Manufacturing 38 (2019) 1508 – 1515, 29th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM 2019), June 24-28, 2019, Limerick, Ireland, Beyond Industry 4.0: Industrial Advances, Engineering Education and Intelligent Manufacturing.

[10] M. Ossenkopf, P. Ennen, R. Vossen, S. Jeschke, Reinforcement learning for manipulators without direct obstacle perception in physically constrained environments, Procedia Manufacturing 11 (2017) 329 – 337, 27th International Conference on Flexible Automation and Intelligent Manufacturing, FAIM2017, 27-30 June 2017, Modena, Italy.

[11] H. Nasereddin, G. M. Knapp, Hybrid robotic reinforcement learning for inspection/correction tasks, Procedia Manufacturing 39 (2019) 406 – 413, 25th International Conference on Production Research Manufacturing Innovation: Cyber Physical Manufacturing August 9-14, 2019 — Chicago, Illinois (USA).

[12] J. Peters, S. Vijayakumar, S. Schaal, Natural actor-critic, in: J. Gama, R. Camacho, P. B. Brazdil, A. M. Jorge, L. Torgo (Eds.), Machine Learning: ECML 2005, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 280–291.

[13] J. Peters, S. Schaal, Applying the episodic natural actor-critic architecture to motor primitive learning, in: ESANN 2007, 15th European Symposium on Artificial Neural Networks, Bruges, Belgium, April 25-27, 2007, Proceedings, 2007, pp. 295–300.

[14] J. Peters, S. Schaal, Natural actor-critic, Neurocomputing 71 (7) (2008) 1180 – 1190, progress in Modeling, Theory, and Application of Computational Intelligenc.

[15] K. W. Mathewson, P. M. Pilarski, Simultaneous control and human feedback in the training of a robotic agent with actor-critic reinforcement learning, arXiv preprint arXiv:1606.06979 (2016).

[16] Y. P. Pane, S. P. Nageshrao, R. Babuška, Actor-critic reinforcement learning for tracking control in robotics, in: 2016 IEEE 55th conference on decision and control (CDC), IEEE, 2016, pp. 5819–5826.

[17] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, P. Abbeel, Asymmetric actor critic for image-based robot learning, arXiv preprint arXiv:1710.06542 (2017).

[18] R. E. Andersen, S. Madsen, A. B. Barlo, S. B. Johansen, M. Nør, R. S. Andersen, S. Bøgh, Self-learning processes in smart factories: Deep reinforcement learning for process control of robot brine injection, Procedia Manufacturing 38 (2019) 171 – 177, 29th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM 2019), June 24-28, 2019, Limerick, Ireland, Beyond Industry 4.0: Industrial Advances, Engineering Education and Intelligent Manufacturing.

[19] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous control with deep reinforcement learning, arXiv preprint arXiv:1509.02971 (2015).