Computing and Informatics, Vol. 41, 2022, 271–287, doi: 10.31577/cai_2022_1_271

AUTOMATING THE DATASET GENERATION AND ANNOTATION FOR A DEEP LEARNING BASED ROBOT TRAJECTORY ADJUSTMENT APPLICATION FOR WELDING PROCESSES IN THE AUTOMOTIVE INDUSTRY

Mohamed Slim Werda

Institut Pascal
UMR 6602, Campus des Cézeaux des Landais BP 80026
Aubière Cedex 63171, France
&
Faurecia Automotive Seating
Brières-les-Scellés, France
e-mail: mohamed_slim.werda@sigma-clermont.fr

Theodor Al Saify

University Bourgogne Franche-Comté
FEMTO-ST Institute
CNRS, Belfort, France
&
Faurecia Clean Mobility, Bavans, France
e-mail: theosaify7@gmail.com

Khalid Kouiss

Institut Pascal
UMR 6602, Campus des Cézeaux des Landais BP 80026
Aubière Cedex 63171, France
e-mail: Khalid.Kouiss@sigma-clermont.fr

Jaafar Gaber

University Bourgogne Franche-Comté
FEMTO-ST Institute
CNRS, Belfort, France
e-mail: jaafar.gaber@utbm.fr

Abstract. Industrial companies are more and more interested in the use of artificial intelligence (AI) in the control and monitoring of their processes. They try to take advantage of the power of this technology in order to increase the level of automation and to build smarter machines with new capabilities of self-adaptation and self-control. Especially, the automotive industry, with their high requirements in productivity and diversity management, are eager to adapt AI concepts to their processes. However, the training of Deep Learning (DL) models requires an important effort of data preparation, providing a dataset of all possible configurations. Indeed, this dataset must be collected and then annotated. Considering the fact that automotive industry deals with a huge number of references and that it often and quickly needs to modify their products, it is very difficult, if not impossible, to gather sufficient datasets for each produced reference and to have the time to train DL models in the plants with the traditional methods. This paper presents an innovative methodology to prepare the dataset by creating virtual images instead of collecting real ones and then automatically annotating them. It will demonstrate that this method will reduce the efforts and the time of the preparation of the dataset significantly. The paper will also present how this method was deployed for the quality control of welding operations in the automotive industry.

Keywords: Industry 4.0, automatic annotation, dataset generation, automotive industry, welding, artificial intelligence, deep learning

1 INTRODUCTION

New industrial machines are now more and more equipped with computer vision applications that are based on artificial intelligence (AI). In the context of the automotive industry these machines are now designed to be reconfigurable to produce a vast amount of parts references. Computer vision modules equipped in these machines need to have a set of trained models corresponding to the different references that are produced by each machine in order to be able to work properly. To train these different models, the process engineers need to collect datasets for each produced reference on each machine and then annotate them (i.e. describe what is present in each image). These datasets are tricky to create as they must cover all the possible configurations for each reference and all the possible variations to the production environment (e.g. changes to the lighting or to the background). Not only is this process time-consuming but also extremely prone to annotation mistakes especially in large datasets. Furthermore, this process must be repeated for each reference making it extremely repetitive. Reducing the time needed for the collection of these datasets becomes a priority for the viability of the industrial applications.

The main contribution of this paper is to propose a methodology to automate the preparation of the datasets that will be used to train Deep Learning (DL) models for new parts references by creating realistic synthetic images and automatically annotating them.

This methodology is being deployed in the plant of a tier one automotive supplier for the quality monitoring and trajectory adjustment of welding operations.

The paper is organized as follows: In Section 2, we present the use-case of a welding operation, its quality challenges and how the use of a DL based approach allows us to improve the quality of welding operations, and we expose the problem of dataset preparation for new references. Then in Section 3, we will discuss our proposed methodology to generate the datasets by using synthetic images rendered in a virtual environment instead of real ones and by the automation of the entire workflow. Finally, we will present considered future work.

2 THE DEVELOPED DEEP LEARNING SOLUTION FOR ROBOT TRAJECTORY ADJUSTMENT IN WELDING PROCESSES AND ITS LIMITATIONS

The case study concerns an assembly operation by welding in an automotive supplier plant. The welding operation is caried out by robots. For this case study the target was to locate a welding edge of a set of identical parts. Since this edge is specific and unique in terms of thickness and length, as a part of the whole welding process, the involved robot could be trained with a Deep Learning approach to classify the images to locate the edges to weld parts. Therefore, an image classification approach could be used as an alternative to complex edge detection methods.

2.1 The Deep Learning Solution

A neural network based on the MobileNets [1] model definition was used to decide if the edges that need to be welded together are present in 3 distinct areas in each Region of Interest. These areas are called zones A, B and C (see Figure 1). Thus, based on the presence or absence of the edge in each different zone, we will determine the robot's welding trajectory adjustment.

To train the deep learning model we needed at least 2 000 photos of the parts that are going to be welded (equal to 2 weeks of production, as each day the plant produces 250 parts). After collecting our dataset, the regions of interest representing the welding seams had to be cropped. Then, these crops were annotated manually

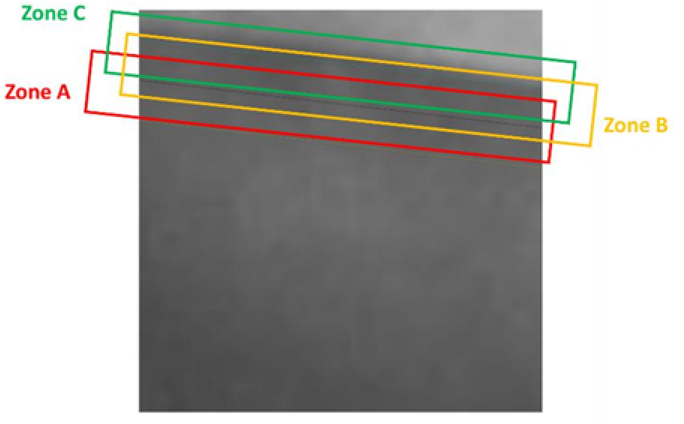


Figure 1. Layout of the overlapping zones A, B and C

to indicate whether a welding seam is present in one or in multiple defined zones in the image.

This way the model was trained to determine if the welding seam is present in any of those areas, and depending on the combination of presence/absence predictions, an offset is calculated and transferred to the robot to adjust its trajectory. An application for patent delivery has been submitted for this solution.

2.1.1 First Experimental Results and Limitations

First experimental results. The validation of this solution was done by a proof of concept to study the functionality and to analyze if it returns what it is intended. The System was installed at Faurecia Columbus South in the United States of America, on a line where mislocated welds were the main reason for reworks (over 90 % of reworked parts).

The plant had a rework rate of 7 % and the objective was to reduce this number at least by half. This proof of concept was turned on at the beginning of the year 2020 and we can clearly see a huge impact on the rework rate – from 7 % to approximately 1 % after 4 weeks of production (see Figure 2).

Then, after six full weeks of production, we turned off the system to see the effect of disabling Robot Trajectory Auto Adjustment. We can see that from week 12 the number of reworks has increased to reach 6 %. The algorithm prediction was monitored regularly to keep track of the accuracy. After each cycle, all the values of offset were saved in text file and then analyzed in order to make a comparison between the ground truth and the value predicted.

Real dataset limitations. Data collection is the most important phase when aiming to train a neural network (supervised learning). It consists of gathering

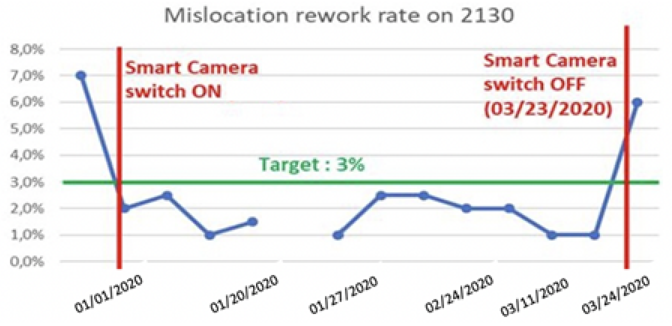


Figure 2. Rework rate before and after switching on the system

a large dataset so it can be annotated and only then the model can be trained. In this industrial application this phase can take up to 2 weeks of production to gather the needed amount. Indeed, on the production line it is not always the same part reference that is produced. In the context of automotive tier one suppliers, a large variety of references are produced by the same machines and the process of gathering images that cover all the possible variations might take a lot of time.

Variation can be either in the geometry, the dimensions or the localization of the welding area which we analyze. We talked about implementing a deep learning method using MobileNets model to detect and calculate if the welding edge has moved and send the necessarily offset to the robot. However, this approach was only effective on simple pieces because we had a rich dataset that contains all the possible variety, but it does not have the same effectiveness when we work on complex parts. This lower efficiency is due to the lack of the variety of images in the training dataset. So, by creating a virtual dataset we can enrich our dataset and create a more accurate model.

3 THE PROPOSED APPROACH

One of the most hindering points in this application is the fact that for every new reference to be added a new Deep Learning model must be trained. And in order to do that new training, a dataset with images of that reference must be generated. The generation of that new training dataset requires approximately 2 weeks in this application. So, in order to reduce this delay, the usage of simulated images in the training phase is being considered as a viable solution.

The training dataset will be generated using a software developed internally (based on the Unity 3D engine). This software takes as input configured CAD models of the parts/systems, their environment/machine and configuration data for the simulation and generates the corresponding annotated image dataset.

3.1 CAD Models

As mentioned by Posada et al. [2] CAD models that are used in industrial applications (solid or shell) are usually overly detailed as they represent the exact shapes of the products. So, in order to optimize the memory usage and the performance of our solution, the models need to be simplified. To do so, and to be able to render realistic images, the CAD models need to be converted to a shell model format that is readable by the Unity 3D engine (.fbx or .obj files in our case). The generated meshes must then be simplified/optimized.

In our application the CAD files are originally created in the CATIA software and are available in the CATProduct/CATPart file formats which need to be first converted to .STL or similar shell file formats (.obj or .fbx in our case) to get the meshes needed for the rendering of realistic images.

3.2 The Considered Variations

To make sure that our dataset covers all possible configurations that can be observed in real-world images and that our trained models will become as robust as possible to these variations, the simulation environment will be randomized before the generation of each image. This domain randomization [3] is completely configurable before launching any dataset generation process. These are the considered variations that we will be randomizing in our generated datasets.

Lighting. We will be randomizing a few settings that will impact the lighting of the scene. These settings include the number of light sources, their types (area, spot, directional, point), intensity, color, range and their pose (position/orientation).

Figure 3. Images of different lighting settings of the same test environment. In this environment we placed a sphere with a background and two light sources to show the effect of variating lighting settings on sample images.

Blurriness. In some situations, the images captured by the camera can get slightly blurry and if not taken into account while creating the dataset this can lead to some overfitting. So, we decided to add a depth of field effect to the simulated cameras to randomly add blurriness to some images.

Noise. Random noise will be added to the rendered images to make sure that the trained model's inference results will be the same regardless of the noise that can be applied to the images. This should make the model more robust to noise

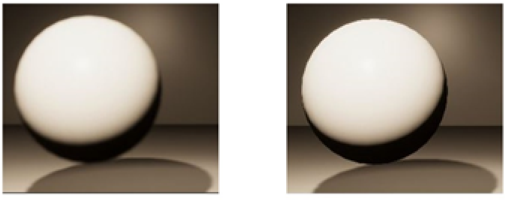
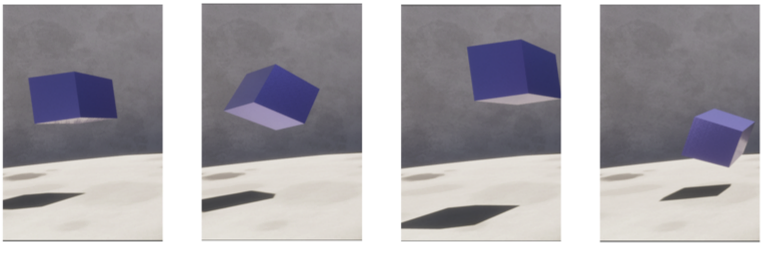


Figure 4. Depth of field effect applied to a sample image

and force it to focus on other features. To achieve this result, the noise type (Gaussian, Poisson, ...), size and colors can be randomized.

Figure 5. Zoom in on a sample image with (left) and without (right) a noise postprocessing effect

Part's position and rotation. The aim of this database generation is to get a diversified dataset representing different shifts of the part and their annotation, so this factor is expected to be the most influential on the model's performance.



a) b) c) d)

Figure 6. Images of the same part with randomized positions and orientations. To create this environment we placed a cube, a background, and a directional light source. We then added a noise and a distortion effect to the rendering camera. In a) there is the result rendering this scene with the original configuration of the cube. In b) we randomized the rotation of the cube. In c) only the position of the cube was randomized. Lastly, in d) both the rotation and the position of the cube were randomized.

3.3 The Workflow

The aim of this proposed approach is to automate most of the dataset generation approach using an internally developed software to help reduce the time-to-market of new projects by minimizing the time needed to generate training datasets for new references.

In order to be able to use the dataset generation software on a new reference, the following steps must be taken:

New reference's CAD model optimization and preparation. As specified in the Secion 3.1, the CAD models need to be converted and simplified before they can be used in our software. The second step is to configure the materials of the optimized meshes to make them look as realistic as possible and then to add the proper behaviors to the different CAD models to specify how they behave during the simulation. This step should be repeated for each new reference as every reference has specific properties.

Simulation environment preparation (as close as possible to the real-world environment). The aim of this step is to use the CAD models of the machine/production system (or to approximate them if it is not possible to obtain them) to recreate a simulation environment for the generated images of our parts to be as realistic as possible in terms of objects placement in the scene [4] or in terms of 3D rendering (in our case achieved through the use of Unity 3D's High Definition Rendering Pipeline HDRP [5]). To do so, first we need to gather ground truth images extracted from the cameras in the machine representing different variations of the environment. Then we must optimize the CAD models and prepare them by configuring their materials (we chose to use Physically Based Rendering PBR materials to add to the realism of the generated images) and adding their behaviors (scripts) to match as close as possible what is represented in the ground truth images. All that remains is to add the cameras and lighting sources and configure them accordingly (based on ground truth images) to obtain the best rendering.

This step (see Figure 7) needs to be done only once per machine. It is not necessary to reconfigure the environment if we are only adding a new reference to our production system.

Simulation configuration and dataset generation. Once the environment and the part's model have been configured, we specify how the simulation environment will be randomized to generate a new dataset of training images. A graphical user interface has been added to the software to make this configuration as intuitive as possible.

Dataset adaptation. The previous annotation process requires a person to manually indicate if the edge is present in each of the zone. When creating randomized images, the current version of the developed software also generates the corresponding annotations.

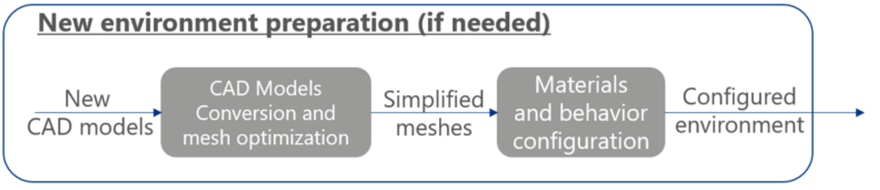


Figure 7. A visual representation of the preparation of a new environment, in which we sum up the tasks to be done in order to prepare the CAD files of a new environment

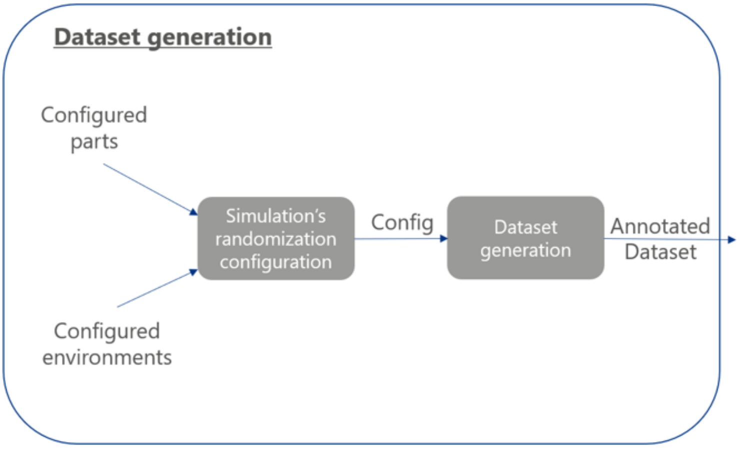


Figure 8. A visual representation of the dataset generation step, in which we sum up the tasks to be done in order to configure and generate a new dataset

In prior case studies, we focused on object detection dataset and thus our annotation software generates annotation in the form of bounding boxes lists. This automatic annotation tool needs to be updated to be compatible with the classification requirement of this application. So, a new script will be added to modify the generated images (by taking crops of the regions of interest in each image) and adapt annotations to be ready for training.

The following steps will allow us to automate the annotation of the produced dataset:

1. Identifying the concerned segments of edges for each Region of Interest on the CAD model.
2. Placing transparent features on the edge segment corresponding to each Region of Interest. Each feature will be named with a different label to help with its identification.

3. Converting the 3D coordinates of each feature to screen space to get the pixel coordinates of that feature (the current annotation algorithm gives a list of all features in the image as bounding boxes with 0 pixels in both width and height).
4. Estimating the slope and y-intercept of the edge based on the pixel coordinates of the features (for each RoI).
5. Knowing the slope and y-intercept of each edge on the master image which was used for calibration and the height of every annotation zone (A,B and C) we will be able decide if the estimated edge in this image is present in each zone and convert this decision to the annotation "in which zones is the edge present?".

With this automation we can represent the tasks needed to be done in this step, as shown in Figure 9.

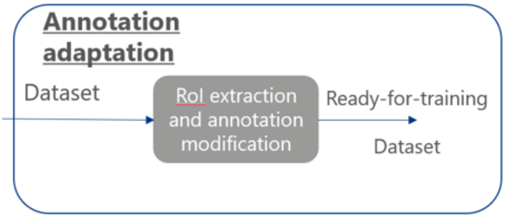


Figure 9. A visual representation of the dataset annotation's adaptation that sums up the tasks to be done in order to adapt the annotation of the dataset

Figure 10 illustrates the expected results of this process on a RoI crop. Steps 1 to 3 will provide us with the pixel coordinates corresponding to each feature. Step 4 will allow us to determine the position of the edge relatively to the crop. And step 5 will determine if the edge is present in each zone and convert the annotation to the corresponding format.

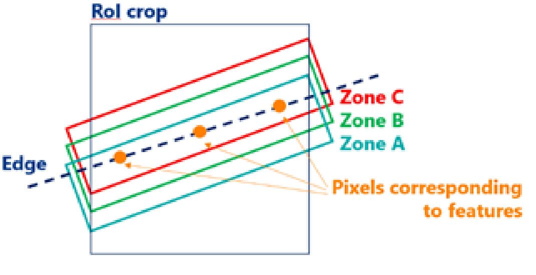


Figure 10. An illustration of the usage of features to determine the edge's location

Figure 11 represents the chronological order of the different steps that need to be taken as well as the inputs and outputs of every step. The first two steps (new parts preparation and new environments preparation) can be done separately, and their results (configured parts/environments) can be saved to add more flexibility to the process. The last two steps can be repeated as much as needed until the results satisfy the requirements.

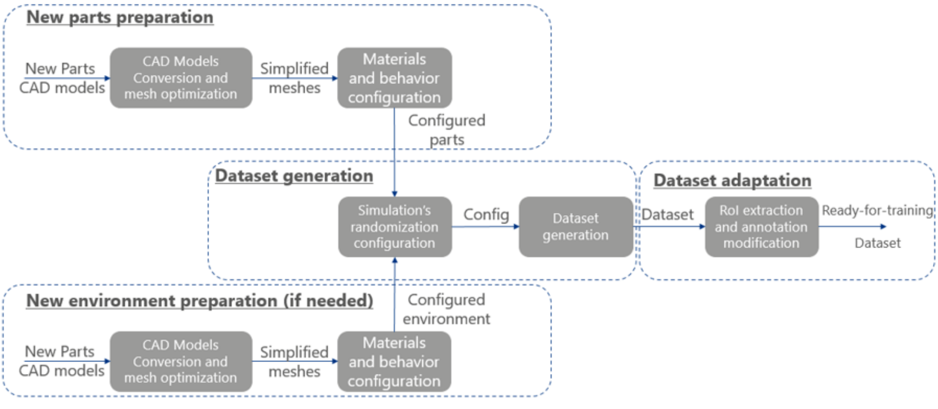


Figure 11. A visual representation of the proposed approach.

3.4 Expected Results of the Proposed Approach

If applied correctly, this approach can result in the generation of highly realistic images that are automatically annotated and are featuring our parts in a simulated environment that is as close as possible to the real-world environment.

In order to apply this approach, many tools and softwares need to be developed. Most of these tools have already been prepared by our team for prior projects. And we are in the process of adapting them to this use-case.

As this is a work in progress, we cannot yet show any simulated images for this project. In Figure 12 you can see crops of images of the same parts' reference in real-world and in a simulated environment that we prepared for another project/use-case.

The aim of these images is to show the realism that can be achieved with the current version of our software.

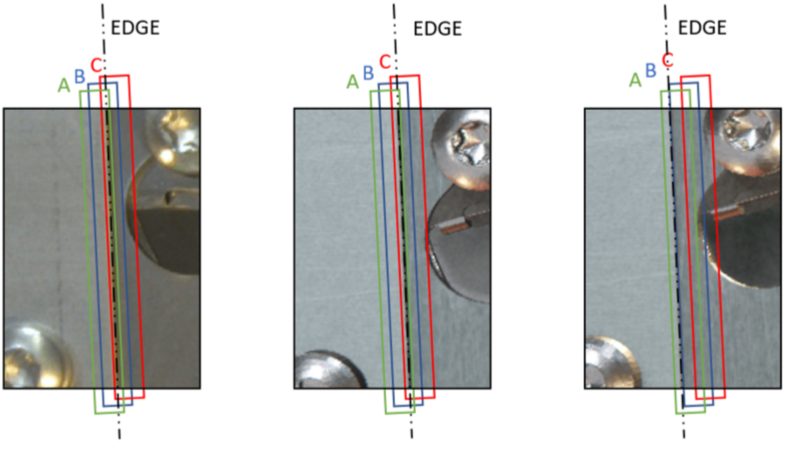
Please note, that in this case the original images have a resolution of $(4\,096 \times 3\,000)$ pixels. In addition, these images are crops of the same Region of Interest with a resolution (140×200) pixels in these 3 images. Each one of these crops shows the same edge in a random position. Therefore, even if these images were not meant to be used in this way, they can showcase how we will apply our approach on the images that will be rendered later.

As it can be seen in Figure 13, this edge's position varies from one crop to another relatively to the 3 zones which were defined based on the position of the

a) b) c)

Figure 12. Crops of the same Region of Interest (140 × 200) pixels in three images: a) is a crop from a real image that was taken as a reference to create the simulated environment used to generate the original images that were cropped to obtain b) and c). In these images lighting, noise, and parts' positions were randomized.

edge in the Master Image (crop a) in this case). As specified in Section 3.3, the slope and y-intercept of the edge will be estimated based on the position of transparent features in the 3D model. The script will then be able to automatically evaluate if the edge is present or not in each of the zones corresponding to the generated images. These calculations should lead us to the results specified in Table 1.



a) b) c)

Figure 13. The position of the edge relatively to the defined zones A, B, and C in each of the crops a), b) and c)

The different combinations of presence or absence of the edge in zones A, B, and C will lead to an estimation of the needed trajectory adjustment to minimize the

Crop	a)	b)	c)
Is the edge present in Zone A?	Yes	Yes	Yes
Is the edge present in Zone B?	Yes	Yes	No
Is the edge present in Zone C?	Yes	Yes	No
Estimated trajectory adjustment	0 mm	0 mm	Depends on calibration

Table 1. Presence of the edge in the different zones represented on each image crop and the estimated trajectory adjustment based on the combination of presence/absence of the edge in these zones

rework rate on these parts. This estimation is based on the results of a calibration process. This calibration process is repeated for each Region of Interest to generate the adjustment values for each combination. Since these images were generated for another project, no calibration was done for these parts. Which is why we mentioned that in Table 1. For crop c) the estimated trajectory adjustment depends on the calibration process.

3.5 Potential Applications of the Proposed Approach

This approach can be applied to multiple industrial projects and is not limited to this case study. Our team has been trying to implement it to different applications, using the internally developed tools. As mentioned in Section 3.3, during the new parts' reference preparation phase and during the new environments' preparation phase we must specify the behavior of the different components in this environment/part.

In the current version of our software, it is possible to simulate different behaviors from which we can mention the randomization of the position and the orientation of a component relatively to the part or environment. And the presence or absence of some components of the parts or environments. It is also possible to randomize some parameters in the shaders of the materials that are affected to the different CAD models in our parts and/or our environment to add to the randomization of the simulated images. The combination of these behaviors allows us to create rich and unique randomized datasets that cover a wider domain, and allow our trained models to be more robust.

The usage of the behaviors shown in Figures 14 and 15 allows us to create datasets which are fit to train models for simple visual inspection applications. That makes it possible to imagine implementing this methodology for other use-cases in the future.

3.6 Limitations of the Proposed Approach

As promising as this approach can be, it is very challenging to put it in place. The needed effort/time to develop such generic applications/pipelines should not be underestimated. Furthermore, the optimization and the preparation of the

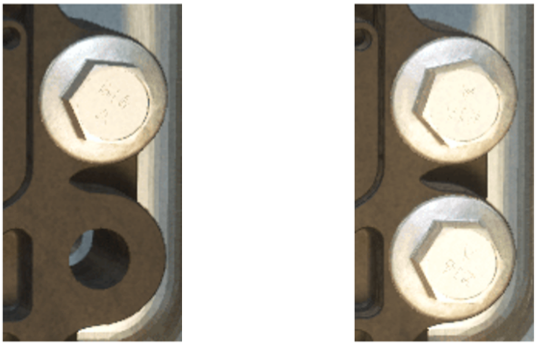


Figure 14. Zoom in on rendered images of components (screws) which have been configured to randomly rotate and/or disappear

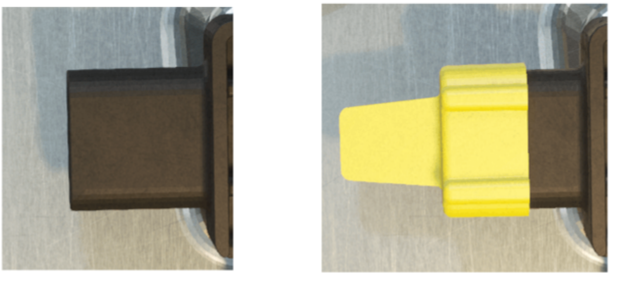


Figure 15. Example of a component (yellow plastic cap) being absent (left) or present (right) during the generation of simulated images

CAD files and the textures/materials needed to make their rendering realistic for parts and/or for environments can be time consuming and can require a specific set of skills that are not common in an industrial environment. We also have to stress the fact that this approach is not necessarily adaptable to all use-cases as it can be even more challenging to recreate realistic variations in some cases (e.g. rendering images of defects in visual inspection applications with randomized defects).

4 CONCLUSION AND FUTURE WORK

In this paper, we presented, as a case study, a computer vision application for robot trajectory adjustment using deep learning for welding processes in the automotive industry. We discussed the requirements of this application, the approach that was used to generate and annotate the needed dataset for the training, the results of the integration of this case study as a proof of concept in a factory, and we exposed its limitations.

We then detailed the approach that we will be taking to remediate to this situation and the expected results once this solution is applied. In future work we will be adding the environment representing the machine to our image generation application to create a simulated dataset, train the model on it and then compare the results on a real test dataset. We will also prepare hybrid datasets by mixing both real and synthetic images and analyze the impact of using different ratios on the inference on real-images-only test dataset.

The work in progress concerns the automated dataset generation and annotation. New models will be trained on the randomly generated datasets and a comparison of the results of the training on the original dataset and the results of our proposed approach will be conducted. With the implementation of the proposed approach, we expect to reduce the time-to-market of newly added references by reducing the dataset creation time from 2 weeks to approximately one day (including new reference's CAD model optimization and preparation, if the environment has already been prepared for other references). This estimation is subjective but based on the time-to-market reduction of prior industrial projects. We also expect the training metrics to improve as the dataset will be more diversified and aimed to give a better representation of all the possible variations to the environment making the inference more robust regarding lighting condition variation and blurriness.

REFERENCES

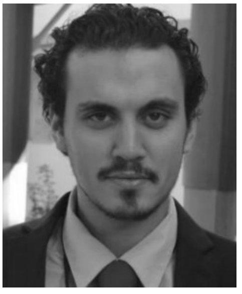
[1] Howard, A. G. et al.: MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. 2017, arXiv: 1704.04861.

[2] Posada, J.—Toro, C.—Wundrak, S.—Stork, A.: Ontology Supported Semantic Simplification of Large Data Sets of Industrial Plant CAD Models for Design Review Visualization. In: Khosla, R., Howlett, R. J., Jain, L. C. (Eds.): Knowledge-Based Intelligent Information and Engineering Systems (KES 2005). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 3683, 2005, pp. 184–190, doi: 10.1007/11553939_27.

[3] Tobin, J.—Fong, R.—Ray, A.—Schneider, J.—Zaremba, W.—Abbeel, P.: Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017, pp. 23–30, doi: 10.1109/IROS.2017.8202133.

[4] Schwarz, M.—Behnke, S.: Stillleben: Realistic Scene Synthesis for Deep Learning in Robotics. 2020 IEEE International Conference on Robotics and Automation (ICRA), 2020, pp. 10502–10508, doi: 10.1109/ICRA40945.2020.9197309.

[5] Unity's HDRP Official Documentation. `https://docs.unity3d.com/Packages/com.unity.render-pipelines.high-definition@12.0/manual/index.html`, last accessed: 20/05/2021.



Mohamed SLIM WERDA is a mechatronics engineer with an M.Sc. degree in artificial perception and robotics. He is a Ph.D. student at the University of Clermont Auvergne, France, working on proposing a novel methodology to facilitate the integration of DL-based modules in cyber-physical production systems. His Ph.D. is financed by and done in partnership with Faurecia Automotive Seating. He currently holds a position of Assistant Professor at Sigma-Clermont Engineering School, France, teaching various subjects related to project management, systems engineering, cyber-physical production systems, and digital transformation of enterprises. His research interests include edge and cloud computing for DL, computer vision and robotics, IoT, reconfigurable CPPS, automating data engineering processes, and project management.

Theodor AL SAIFY is Ph.D. student from University of Technology Belfort-MontbÈliard in the domain of robotics, artificial intelligence and computer vision. His Ph.D. was done at Faurecia Clean Mobility in Bavans. During those years he took part in multiple courses related to AI, IoT and NVIDIA topics. Also his thesis contributions were deployed in multiple Faurecia plants in the world (USA, Spain, Netherlands) and also in the pattern recognition field.

Khalid KOUISS is Professor of industrial computer sciences and industrial engineering. He is also an expert in industrial automation and system engineering. He obtained his Engineer diploma in 1988 and the Ph.D. in 1992 from Ecole Centrale of Nantes in France. His research activities concern system modeling, system engineering, intelligent manufacturing systems (IMS), manufacturing execution systems (MES), automation for complex manufacturing systems, industrial robotics and more generally the deployment of Industry 4.0 concepts. Today, his main activities are shared between professorship at the Engineering School SIGMA in Clermont-Ferrand and his work as a member of Pascal Institute Research Laboratory at the University of Clermont Auvergne (UCA) and as an automation expert consultant.

Jaafar Gaber received his Ph.D. degree in 1998 from University of Lille I. He is currently Researcher at the FEMTO-ST Institute (UMR CNRS 6174) and UTBM. Prior to joining UTBM, he was a research scientist at CSI Institute in George Mason University in Fairfax (Virginia, USA). His research interests include modern distributed systems, including IoT, ubiquitous and pervasive computing, reconfigurable, embedded computing and distributed/parallel algorithms. His research has been frequently supported by European grants within EU FP6 and EU FP7 and industry. He is also involved in European EACEA agency projects. Prior to join FEMTO-ST in 2017, he led the research axis on mobility within the OPERA Research Lab. He is a member of the IEEE and ACM.