

Visualizing Corrosion in Automobiles using Generative Adversarial Networks

Andre Von Zuben¹, Renato G. Nascimento², and Felipe A. C. Viana³

^{1,2,3} *Department of Mechanical and Aerospace Engineering, University of Central Florida, Orlando, FL, 32816, USA*
avzuben@knights.ucf.edu
renato.gn@knights.ucf.edu
viana@ucf.edu

ABSTRACT

In this contribution, we classify the state of corrosion of cars in images as none, mild, moderate, and severe. We use generative adversarial networks to help to transfigure non-corroded car images to the other classes. In other words, the model ingests an image with a car having no corrosion and generates an image of this car at any of either mild, moderate, or severe corrosion levels. We proposed an approach that is able to handle the particularities of this application. For example, we had to work with only several hundred images as opposed to the many thousands to million images commonly found in many computer vision problems. Our data is highly unbalanced with many more images of cars with no corrosion dominating the cars with any level of corrosion. Additionally, the data is poorly labeled, as classification is highly subjective. Despite the challenges, results indicate that our generative adversarial networks can be trained with relative accuracy given limitations on the data set. Obviously, these results show that the performance of the model depends on how well the training set represents the particular target corrosion level.

1. INTRODUCTION

Undoubtedly, corrosion is an expensive problem for the automotive industry. As a matter of fact, there are studies estimating that the global cost of corrosion can reach 2.5 trillion US dollars (around 3.4% of global product), while the national costs of corrosion generally represent approximately 1–5% of the gross national product (Koch et al., 2016; Liu, Guo, Wang, & Yergin, 2018; Hou et al., 2017). Factors that induce corrosion in vehicles include, but are not limited to, extreme temperatures, high levels of humidity (through exposure to rain, snow, and coastal areas), accumulation of dirt, mud, and debris, presence of wet condensates and appreciable concentrations of chloride ions and de-icing salts. The large variations

in corrosion-inducing factors across the markets in which the vehicles are sold coupled with variations in vehicle usage make predicting vehicle corrosion a daunting task. In this work, we aim to use the learning and generation capabilities of generative adversarial networks to aid in this task.

As discussed by Goodfellow et al. (2014), generative adversarial networks consist of a two-player adversarial game with two main components, a discriminator and a generator. The discriminator network learns to determine whether a sample is from the model distribution or the data distribution. The generative network creates an artificial sample and tries to fool the discriminator. Recent studies show that generative adversarial network architectures achieve impressive results especially in image-to-image translations applications (Zhu, Park, Isola, & Efros, 2017; Karras, Laine, & Aila, 2019; Shaham, Dekel, & Michaeli, 2019), and are the main motivation for this work.

In this paper, the discriminator part of the deep learning model learns how to identify different levels of corrosion (none, mild, moderate, and severe) present in image containing cars. On the other hand, the generator part of the model learns how to transfigure images of cars from one class into another. After training, the generative adversarial network is able to ingest the image of an automobile (used or new) and predict how the same would look like at different corrosion levels. Figure 1 illustrates the results we obtained with our proposed approach.

In our numerical experiment, we propose solutions for the challenges that are particular to the application. For example, we dealt with largely unbalanced datasets (the higher the corrosion level, the harder it is to acquire relevant and high-quality images). Additionally, we had to handle highly subjective labeling, which imposes all the problems associated with noisy dataset. Finally, the dataset is relatively small for the complexity of the problem. While in many computer vision problems the datasets contains many thousands to million images; in our application, we had to work with only several hundreds. The methodology we developed here can be used to aid with visualization of damage over long periods of time. We believe

Andre Von Zuben et al. This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 United States License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

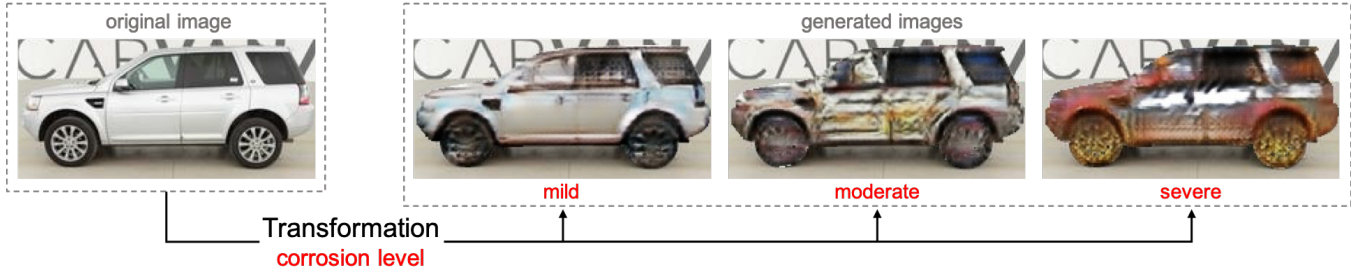


Figure 1. The proposed model learns how to transform a car image into 3 different levels of corrosion.

when coupled with modeling of physics of failure, it will serve as an aid tool to the prognosis community.

The remaining of the paper is organized as follows. Section 2 presents a brief review of the literature, contextualizing our contribution in terms of generative adversarial networks. Section 3 details our proposed formulation regarding generative adversarial network design, loss function definitions, and segmentation model. Sections 4 and 5 present and discuss the results of the numerical experiments. Finally, section 6 closes the paper recapitulating salient points and presenting conclusions and future work.

2. BACKGROUND AND RELATED WORK

Since its introduction, generative adversarial networks have been extensively studied, especially for computer vision tasks. Radford et al. (Radford, Metz, & Chintala, 2015) show that generative adversarial networks with convolutional neural networks (CNN) can effectively learn useful features from images. They have also laid the foundation and insights on how adequately train a generative adversarial network. In the application presented in this work, we need to condition the generative network to some specific characteristics (corrosion level). The concept of conditioning the learning of generative adversarial networks with prior information, introduced by Mirza et al. (Mirza & Osindero, 2014), is central to many state-of-the-art methods, especially for image-to-image translation.

Image-to-image translation, another critical aspect of our application, is the task of transforming an image from one domain to another (in our case, a car without corrosion to different levels of corrosion). The central premise of image-to-image networks is the capability to capture the shared and distinctive features of each domain, allowing the transfiguration of the different features while keeping the common aspects of different domains. The ‘‘pix2pix’’ framework (Isola, Zhu, Zhou, & Efros, 2017) successfully comprised conditional adversarial networks as a general-purpose solution to image-to-image translation problems. The success and impressive results achieved by ‘‘pix2pix’’ comes along with some significant limitations. The effectiveness of the method depends on very large sets of aligned image pairs. Nevertheless, many

applications (including the presented in this work) do not have such large supervised datasets available.

In order to learn to translate an image from a source domain to a target domain in the absence of paired examples. Zhu et al. (2017) proposed a method called cycle-consistent generative adversarial network (CycleGAN). The main idea is to use transitivity as a way to regularize structured data. They introduced the cycle consistency loss that captures the premise that if we translate from one domain to the other and back again, we should arrive at where we started.

We mostly base our work on the findings of the CycleGAN framework. The main differences are while in CycleGAN they focus on two pair images, translating for one domain to another, here we proposed a scenario with multiple domains. Moreover, we build a framework to use unsupervised data scraped from the internet, increasing the complexity and robustness of the model.

3. PROPOSED METHOD

The overall model architecture is illustrated in Figure 2. The main goal of the proposed model is to learn the mapping functions between four different domains A , B , C and, D (representing the respective corrosion levels classes: none, mild, moderate, and severe), as also illustrated in Figure 3. Given the purpose of this specific application, we only illustrate the transformation of car images without corrosion (A) to some corrosion level (B , C , D). Following the idea of the cycle consistency (Zhu et al., 2017), for each mapping from A to other domain (e.g., $A \rightarrow B$), we have a return mapping to A (e.g., $B \rightarrow A$). This gives a total of 6 mappings ($A \rightarrow B$, $A \rightarrow C$, $A \rightarrow D$, $B \rightarrow A$, $C \rightarrow A$, $D \rightarrow A$), each being modeled by a generative network (blue boxes in Figure 2). We utilize the architecture from CycleGAN (Zhu et al., 2017) for our generator models, which have accomplished notable results in unpaired image-to-image translation problems. Each of them is composed of 2 stride-2 convolutions, 9 residual blocks (H, Zhang, Ren, & Sun, 2016), and 2 stride- $\frac{1}{2}$ convolutions.

Moreover, we have 4 adversarial discriminators (D_A , D_B , D_C , D_D) where each one aims to distinguish between images

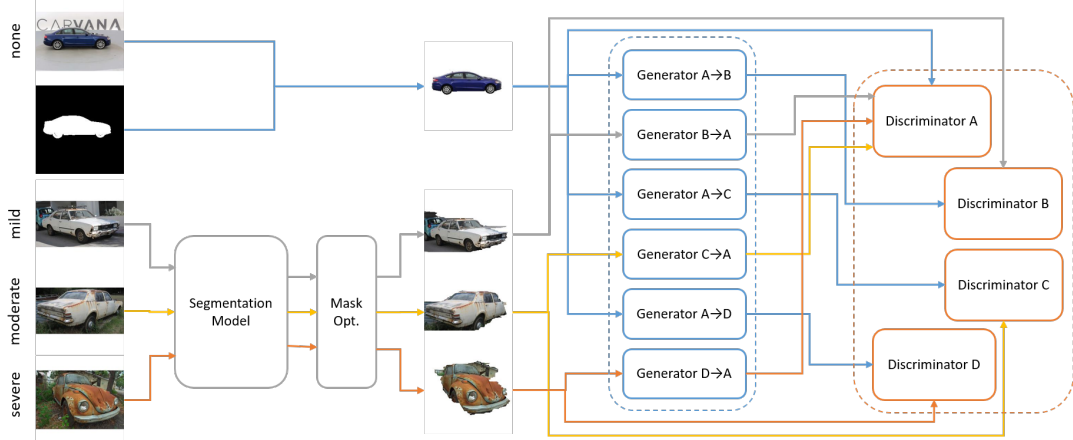


Figure 2. Full network architecture. At first, the mask is applied to the input without corrosion. For all the other rusty classes, the segmentation model predicts the masks, passing each of them to the mask optimizer, and then the optimized mask is applied to the rusty images. Following, each sample is passed to the corresponding generator (blue boxes), which produces its corresponding representation on the other domains. Finally, we have 4 adversarial discriminators (orange boxes) where each one aims to distinguish between images from the original domain and the translated images.

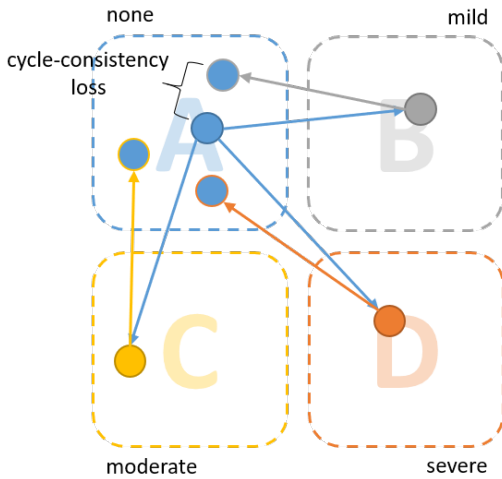


Figure 3. Cycle-consistency loss illustration.

from the original domain and the translated images (orange boxes in Figure 2). For instance, D_B distinguishes images between the domain B and the generated images from $A \rightarrow B$. Each of our discriminators is constituted of one stride-2 convolutional layer, followed by 3 blocks of one stride-2 convolution and instance normalization and a final stride-2 convolution. Additionally, we built a segmentation network to eliminate the background of the input images, so our generative networks can focus the learning only on the car as detailed in Subsection 3.2.

3.1. Loss Function Definitions

The overall loss is compound by two main terms. There are adversarial losses for matching the distribution of generated

images to the data distribution in the target domain. Additionally, there are cycle consistency losses to prevent the learned mappings from contradicting each other.

The adversarial losses are applied for each mapping as:

$$\begin{aligned} \min_G \max_D L_{GAN}(G, D_{T^*}, O^*, T^*) = & \\ & \mathbb{E}_{t \sim P^{data(t)}} [\log D_{T^*}(t)] \\ & + \mathbb{E}_{o \sim P^{data(o)}} [\log(1 - D_{T^*}(G(o))], \end{aligned} \quad (1)$$

where G aims at minimizing the objective against an adversary D that tries maximizing it; O^* and T^* as origin and target domain ($A, B, C, \text{ or } D$); G is the mapping $O^* \rightarrow T^*$; F is the come back mapping $T^* \rightarrow O^*$; and finally, $o \sim P^{data(o)}$ and $t \sim P^{data(t)}$ are the data distribution for origin $o \in O^*$ and target $t \in T^*$ considering the training samples $o_{i=1}^N$ for each domain.

To ensure that the learned mappings are cycle-consistent, we use the cycle consistency loss as:

$$\begin{aligned} L_{cyc}(G, F) = & \mathbb{E}_{o \sim P^{data(o)}} [F(G(o)) - o] \\ & + \mathbb{E}_{t \sim P^{data(t)}} [G(F(t)) - t], \end{aligned} \quad (2)$$

where the forward cycle consistency is enforced given that, for each image o_i from domain O^* , the image translation cycle should be able to bring o back to the original image (i.e., $o_i \rightarrow G(o_i) \rightarrow F(G(o_i)) \approx o_i$). Similarly, should also satisfy the backward cycle consistency (i.e., $t_i \rightarrow F(t_i) \rightarrow G(F(t_i)) \approx t_i$).

The final loss is a balanced combination:

$$\begin{aligned} \min_{G, F} \max_{D_{O^*}, D_{T^*}} L(G, F, D_{O^*}, D_{T^*}) = \\ L_{GAN}(G, D_{T^*}, O^*, T^*) \\ + L_{GAN}(F, D_{O^*}, T^*, O^*) \\ + \lambda * L_{cyc}(G, F), \end{aligned} \quad (3)$$

where λ controls the relative importance of the two objectives.

3.2. Segmentation Model

As we demonstrate later on Section 4, the preliminary results of the proposed model show that the backgrounds of the Rusty Cars dataset had an evident effect on the model outputs. In order to overcome this undesirable effect, we proposed the addition of a segmentation model to try to eliminate the background and focus the learning only on the car. We implement the segmentation model in a U-Net architecture (Ronneberger, Fischer, & Brox, 2015) with the MobileNetV2 (Sandler, Howard, Zhu, Zhmoginov, & Chen, 2018) as the backbone. While the MobileNetV2 works as the encoder part of our U-Net, we employed 5 transposed convolutional 2D layers as our decoder, using the MobileNetV2 blocks #1, #3, #6, #13, and #16 outputs as skipped connections to reconstruct the image mask.

As illustrated in Figure 4, the generated mask is still very noisy. We further improved the segmentation results by adding a mask optimization algorithm with an adaptive threshold technique (Yasira Beevi & Natarajan, 2009). At first, we look for all background regions with an area of less than 10% of the total area and transform them into the foreground. This process guarantees that there are no holes left on our foreground prediction. After that, we employ the same algorithm scanning for foreground regions with an area of less than 10% of the total, switching them to the background. Applying this method, we end up removing the noisy foreground predictions. We have chosen the threshold of 10% considering the nature of our dataset, which is composed of car pictures.

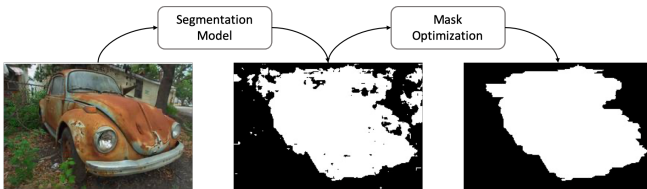


Figure 4. Segmentation Model and optimization to generate masks for the Rusty Cars dataset.

4. NUMERICAL EXPERIMENTS

4.1. Dataset

We started by creating a dataset to solve the problem presented in the paper. This new dataset was created from scratch using pictures retrieved by Google Images searches and com-

plemented with samples presented on the Kaggle Carvana Image Masking Challenge (Kaggle, n.d.) dataset. Details of the datasets, training methods, ablation studies, and results can be found throughout this section.

We did not find a readily available and pre-labeled dataset to solve the proposed problem. Therefore, we created a dataset based of images freely available. Initially, we have created a web scraper script that queries Google Images with a given term and saves a determined number of results. Our base dataset was composed of the matches of the "car rust" and "rusty car" queries. As expected, the search retrieved a couple of noisy data, which we manually remove from the dataset. Finally, we classified the images in three corrosion levels categories: mild, moderate, and severe, as shown in Figure 5.



Figure 5. Examples of images in our Rusty Cars dataset. The first row shows examples of cars with mild corrosion levels, we can observe samples of moderate corroded vehicles on the second row, and the last row shows instances of cars with a severe corrosion level.

Our dataset was also lacking cars with no corrosion in it. We then used the Kaggle Carvana Image Masking Challenge (Kaggle, n.d.) dataset, which contains a large number of car images. Each car has precisely 16 images, each one taken at different viewpoints as shown in Figure 6. This dataset also includes a cutout mask for each of the provided pictures. In addition to this, we also performed data augmentation, cropping the images resulting in a partial view of the left, center, and right part of the car, as shown in Figure 7.

Although we have a massive amount of images picturing cars with no corrosion provided by the Carvana dataset, the number of rusty car images with a reasonable quality is deficient. Our final dataset is composed of 65 mild, 97 moderate, and 73 severe samples. As the goal of this project is not to remove corrosion from cars, we have opted to repeat the rusty images during the training process. Therefore, we added 300 random Carvana pictures to our dataset. The size of all images in our datasets is 256×256 .



Figure 6. Examples of images and masks in the Kaggle Carvana Image Masking Challenge dataset.



Figure 7. Examples of images in the extended Carvana dataset, with multiple car models and diverse point of views and different crops.

4.2. Training of Neural Networks

We first trained our segmentation model (even before we train our generative adversarial network). We trained the model using the original Carvana dataset across 50 epochs, using Adam optimizer (Kingma & Ba, 2014) with a 0.001 starting learning rate reducing on the plateau by a factor of 5% and sparse categorical cross-entropy loss. Figure 8 shows the loss and the accuracy progression during the epochs. It is noticeable that the segmentation model is overfitting, implying that some changes need to be done either on the architecture or the training strategy to overcome this issue.

After that, we pre-process our data to save time during the training of the generative adversarial network. First of all, we generate the masked images for our samples without corrosion. Following, we predict the masks for all the other rusty classes, passing each of them to our mask optimizer. Finally, the rusty masked images are created.

The proposed network was trained for 200 epochs with a batch size of only one sample. Therefore, for each step of each

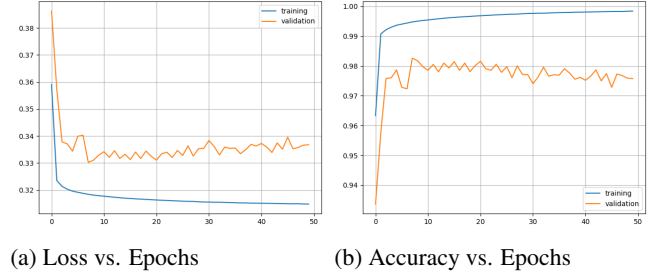


Figure 8. Segmentation model training plots.

epoch, a sample of each of the domains is taken as input, and their corresponding representation on the other domains are generated ($A \rightarrow B$, $A \rightarrow C$, $A \rightarrow D$, $B \rightarrow A$, $C \rightarrow A$, $D \rightarrow A$). Next, the generator and discriminator losses are calculated, as discussed in Section 3.1. Their values through the steps are shown in Figures 9, 10, and 11. The λ parameter described on Equation 3 was fixed to 10.

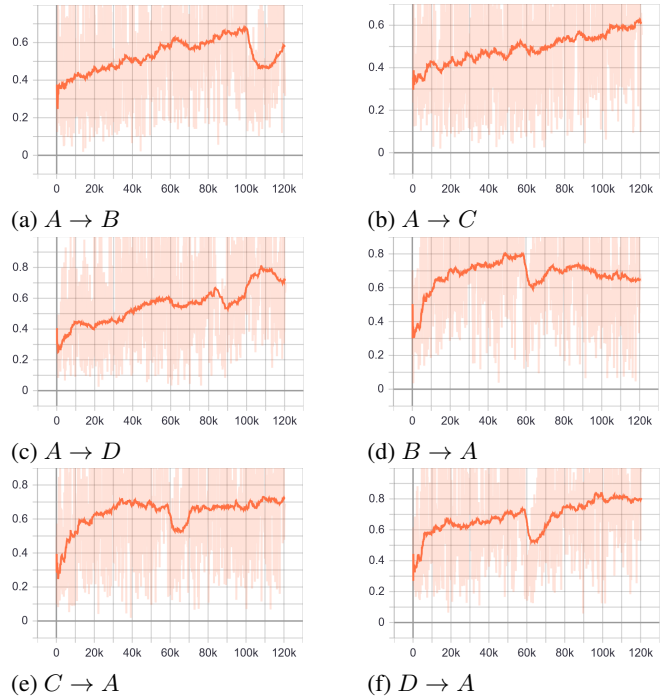


Figure 9. Generator loss vs. steps

4.3. Testing Neural Networks

We tested our solution without the segmentation model to analyze its importance to the results. As shown in Figure 15a, when we removed the segmentation model from our architecture, our network focused on learning the background changes instead of the car features.

Our next experiment was regarding the influence of the mask optimizer on the outcomes. After removing it from our schema, the results were generated with some holes due to the poor

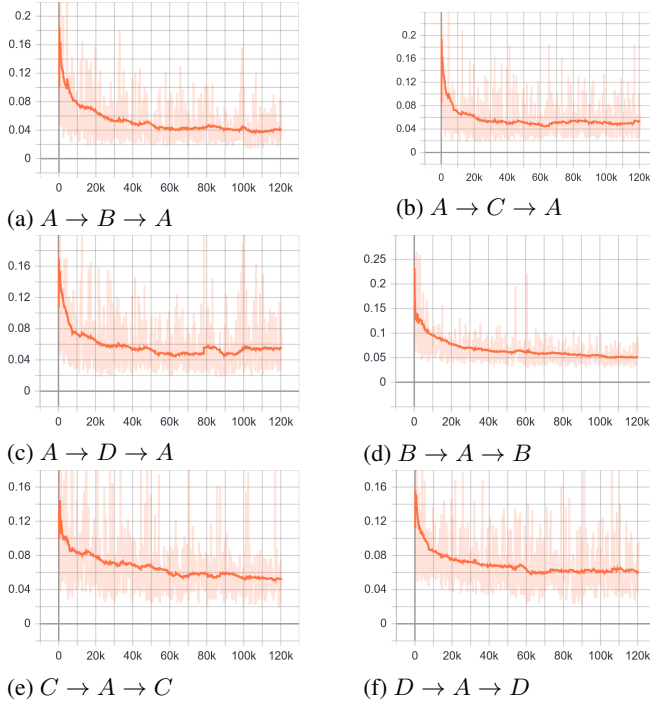


Figure 10. Generator cycle consistency loss vs. steps

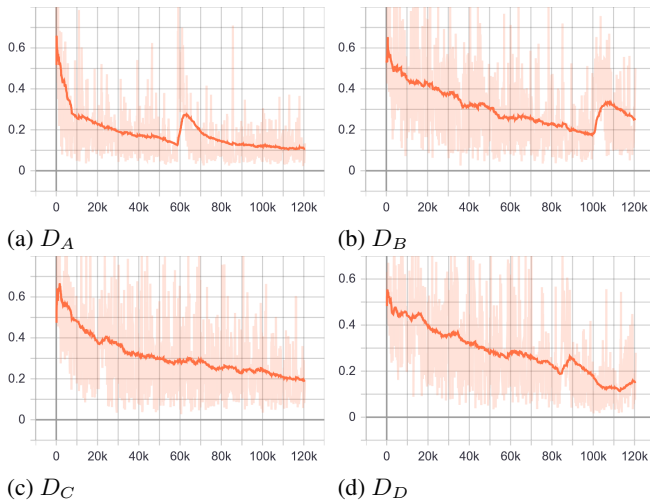


Figure 11. Discriminator loss vs. steps

quality of our predicted masks. Figure 15b shows that our model learned to reproduce the effects of a poorly generated mask applied to the original image.

We evaluated our model using our extended Carvana dataset (Figure 7) described here. For each of the samples without corrosion that we randomly selected, we generated its corresponding representation on each of the other domains (mild, moderate, and severe).

Overall, our proposed network could learn the characteristics presented in different levels of corrosion, as illustrated in

Figures 12, 13, and 14. It is possible to observe that the rust starts on the car fenders and the lower part of the doors for mild levels, then it continues to spread over the doors and the hood in moderate levels, and finally to the whole car in severe conditions. Our model could handle images showing the whole car and also partial view scenarios, where only a small part of the car is visible.

5. DISCUSSION

After analyzing the results presented in the previous section, we can establish that the implementation of the image-to-image generative adversarial network with cycle consistency with multiple classes, and therefore, multiple domains, is possible. While the results here exhibited thus far not perfect recreations, they validate the concept. We believe that results would get dramatically better with potentially less problematic set of images of rusty cars.

The model was able to transfigure the input images (with no corrosion) to the different corrosion levels classes, even considering the large variability in point of view, scale, and incomplete images. Unfortunately, many examples manifested inconsistency in the colors of the transformed cars, especially in the severe class. We believe that it might be a direct artifact of the low number of samples in the rusty car dataset. This subset has just a few color variations; and in the most severe case, the brown ‘rusty’ color predominant. For instance, the bright blue color of the second column example in Figure 12 is transformed into more common colors in all classes.

Sometimes, during the training of the model, the quality of the transformation outputs started to degrade after several epochs. This might be directly related to the intrinsic multi-objective loss function. As presented in the Equation 3, the factor λ can be used to control the relative importance of the objectives, and it was kept fixed during the whole training. We believe that a scheduling adjustment of this factor could help the model training stability.

Additionally, when training without removing the background from the images, as demonstrated in Figure 15, the performance of the model noticeable decreases. Consequently, it becomes clear for us that the segmentation model played a crucial role in achieving the presented results. We understand that the quality of the results could substantially be improved by having better cutout masks for the rusty cars dataset. This could be achieved by improving the labeled dataset (including pictures purposefully taken to highlight rusty cars, manually annotating segmentation masks, or improving the segmentation model and mask optimization used). After examining the training history and outputs of the segmentation model shown here, we believe that, potentially, the model could be improved by adjusting the architecture (using more layers of MobileNetV2 or replacing it with a more complex backbone).



Figure 12. Example of results when input images exhibit the entire car. First row shows the input images. Second, third and fourth rows show the generated images for each respective class of corrosion level (mild, moderate and severe).



Figure 13. Example of results when input images exhibit closer big section of the car. First row shows the input images. Second, third and fourth rows show the generated images for each respective class of corrosion level (mild, moderate and severe).



Figure 14. Example of results when input images just exhibit a small part of the car. First row shows the input images. Second, third and fourth rows show the generated images for each respective class of corrosion level (mild, moderate and severe).

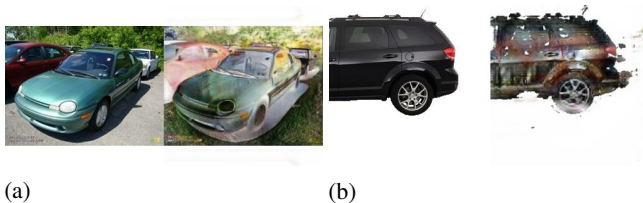


Figure 15. Ablation results: (a) Network without segmentation model focuses on the background. (b) The model learned to reproduce the holes due to a badly generated mask.

6. CONCLUSION

In this work, we proposed an approach for visualizing corrosion of automobiles based on generative adversarial networks. We extended the CycleGAN capabilities to multiple classes, enabling it to transform images between multiple different domains. In our application, we used our models to transform images of cars presenting no signs of corrosion to cars having different levels of corrosion (from none to either mild, or moderate, or severe).

The main challenges found in this research had to do with the dataset. Images of cars without signs of corrosion were easily obtained (Carvana dataset) and had examples of different car models, points of view, colors, etc. The only drawback was the low variability in light conditions (as these images were all in relatively high white light exposure). However, high-quality images of cars with different corrosion levels were extremely difficult to obtain. The dataset collected here was biased towards old car models (which are not a reflection of the Carvana dataset), with busy backgrounds, and poor paint

color variability. These problems are on top of the subjective classification of the level of corrosion the cars presented in these images. Under these circumstances, we believe the model was still able to handle the poor dataset and it was capable of transforming the car images in different levels of corrosion. As we already discussed, the results can be improved; but nevertheless, they show the model capabilities and constitute a valid proof of concept. In the future, we could couple this approach with modeling of physics of failure to aid with visualization of damage progression. Alternatively, once our proposed method is perfected, we could use it in conjunction with virtual and augmented reality for training of personnel in visual inspection.

REFERENCES

- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative adversarial nets. In *Advances in Neural Information Processing Systems* (pp. 2672--2680).
- H, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (p. 770-778).
- Hou, B., Li, X., Ma, X., Du, C., Zhang, D., Zheng, M., ... Ma, F. (2017). The cost of corrosion in China. *Npj Materials Degradation*, 1(1), 1--10. doi: <https://doi.org/10.1038/s41529-017-0005-2>
- Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Com-*

- puter Vision and Pattern Recognition* (pp. 1125--1134).
- Kaggle. (n.d.). Carvana image masking challenge [Computer software manual]. (<https://www.kaggle.com/c/carvana-image-masking-challenge/data>)
- Karras, T., Laine, S., & Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4401--4410).
- Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv 2014. *arXiv preprint arXiv:1412.6980*.
- Koch, G., Varney, J., Thompson, N., Moghissi, O., Gould, M., & Payer, J. (2016). International measures of prevention, application, and economics of corrosion technologies study. *NACE International*, 216.
- Liu, M., Guo, Y., Wang, J., & Yergin, M. (2018). Corrosion avoidance in lightweight materials for automotive applications. *NPJ Materials Degradation*, 2(1), 1--4. doi: <https://doi.org/10.1038/s41529-018-0045-2>
- Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.
- Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.
- Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: convolutional networks for biomedical image segmentation. In N. Navab, J. Hornegger, W. M. Wells, & A. F. Frangi (Eds.), *Medical image computing and computer-assisted intervention -- miccai 2015* (pp. 234--241). Springer International Publishing. Retrieved from http://dx.doi.org/10.1007/978-3-319-24574-4_28 doi: 10.1007/978-3-319-24574-4_28
- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018, June). Mobilenetv2: Inverted residuals and linear bottlenecks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, USA.
- Shaham, T. R., Dekel, T., & Michaeli, T. (2019). Singan: Learning a generative model from a single natural image. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 4570--4580).
- Yasira Beevi, C., & Natarajan, S. (2009). An efficient video segmentation algorithm with real time adaptive threshold technique.
- Zhu, J.-Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 2223--2232).