Accurate Damage Dimension Estimation in AI Driven Vehicle Inspection System

Adrita Barari, N. V. S. Abhilash(✉), Payanshi Jain, Ankit Sati, Karthik Sai Datta, and Chirag Jain

Data Science and Insights, Genpact, New York, USA
abhilash.nvs2@genpact.digital

Abstract. With the rise of Artificial Intelligence, research in Computer Vision and Deep Learning has made substantial advancement on understanding visual content. This has led industries across multiple sectors to upgrade and automate processes which require some visual understanding of the physical world. Damage assessment in vehicles is an important step for insurance claims and auto finance industry. The damage assessment involves granular part detection, damage localization and classification into different damage types such as dent, scratch, crush, tear, etc. Further, physical dimension estimation of the damages is required for computing cost to the customer. Currently, this process involves manual interventions, where an agent assesses the damages physically. This process leads to long turnaround time for the insurance majors and auto finance organizations. In this paper, we present a near real-time end-to-end solution which yields accurate damage detection and propose approaches for providing dimensions of the damages for accurate repair cost estimates for the vehicle. The solution further incorporates an ensemble of computer vision algorithms to calculate dimensions, generating bounding boxes, and consolidating the damage predictions to provide an overall damage estimate from multiple images captured by vehicle owner. Insurance majors and auto finance organizations can substantially reduce their turn around time for claims and charging penalties for damage incurred, respectively by integrating the developed solution in their existing process.

Keywords: Deep learning · Computer vision · Auto finance · Auto claims

1 Introduction

The current Auto Finance industry performs external damage assessment of the leased vehicle before taking it back from their customers. The assessment involves a granular analysis of the damages incurred to the body of the vehicle. The customer is charged based on the number and length of the damages present in each panel of the body. Currently, the assessment process is manual and time consuming. Many ongoing researches focus on to improve the process using automated computer vision systems to detect damages in the vehicle while providing accurate dimensions of the damage. An ideal damage detection system should be able to localize the damages in the vehicle accurately. However, designing a robust system is challenging as the images captured can have various lighting conditions, variable zoom, high reflective and shiny surfaces on vehicle's body, etc. The system should also provide an accurate estimation of the damage dimensions. Detecting the dimensions of the damages can also have challenges like lack of reference object, variable perspective at the corners of the vehicle, etc. In this paper, we extend our damage detection solution with accurate and efficient damage dimension estimation in vehicles to create an end to end pipeline. The concepts presented in this paper are transferable to other visual damage diagnosis systems as well.

2 Related Work

There have been previous efforts at damage detection in buildings using high resolution satellite images [1, 2]. For damage detection in cars, Jayawardena et al. [3] proposed a 3D model projection of an undamaged vehicle on that of a damaged vehicle to identify damages. Only recently have researchers started using deep learning approaches for solving damage detection. Patil et al. [4] have proposed a deep learning based approach for classifying different types of damages in cars. However, our proposed solution is different from [4] since our final objective is to perform damage estimation. The developed framework is a deep learning and computer vision based pipeline. For filtering out irrelevant images of the vehicle, Convolutional Neural Network (CNN) architectures such as ResNet50 [5, 6], Inception_v3 [7], Xception [8] were experimented with. Object detection algorithms were explored to localize the damages and classify the damage types into dents, scratches, tears, misalignments etc. For object detection, architectures such as Faster-RCNN [9] and 'You Look Only Once' (YOLO) [10] were assessed. However, to identify the accurate damage area, we finally use a semantic segmentation approach to predict the accurate segmentation maps of the damages. Deeplab [11] with Xception-65 backbone gives us the best performance for predicting accurate segmentation maps. The proposed solution in explained in details in the following section.

3 Proposed Solution

The proposed solution consists of three key modules, namely damage detection, damage dimension estimator and damage duplicity removal. The solution leverages multiple images of the damaged vehicle taken by the user from different viewing angles as input. In our previous work [12, 13], we described our damage detection pipeline which is a prerequisite for estimating dimension. The same has been captured below in Sect. 3.1. The damage detection module is treated as an object segmentation problem since we need to find the precise locations of the damages down to the pixel level. The localized damages are then fed into the dimension estimating module which predicts the dimensions of the detected damages in each image. Further, we pass the results obtained to a duplicity removal module which removes redundancy if the same damage is present in multiple images. Figure 1 illustrates the end-to-end pipeline on a high level.

3.1 Damage Detection

Our damage detection pipeline consists of three main sequential steps which are relevance filtering, vehicle part segmentation and vehicle damage segmentation. All the deep learning models were trained using a Nvidia Tesla M60 GPU with 8 GB RAM.

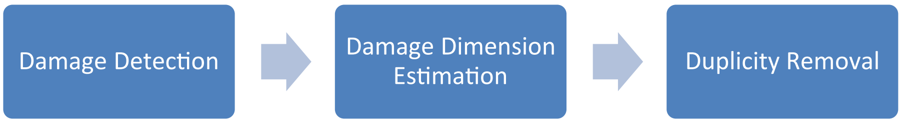

Fig. 1. Block diagram of the damage estimation pipeline

Relevance Filtering. This preprocessing step aims to increase the efficiency of the damage detection module by filtering out irrelevant images. Often the images captured by the customers include images irrelevant to visual damage detection such as images of vehicle documents, odometer readings, license plate etc. Customers may also capture images of car interiors which is out of scope of our current work. The relevance filtering model aims to filter out such images which are irrelevant to our damage analysis. A ResNet50 model trained on around 5000 hand-labelled images has been used for this image classification task. The model was trained for 50 epochs with a learning rate (lr) of 0.001. The accuracy achieved for this model was 91%.

Vehicle Part Segmentation. Parts segmentation plays an important role because Auto Finance companies usually calculate the cost by counting number of damages in each vehicle part. The vehicle part segmentation model was trained using DeepLabv3+ [11] architecture with TensorFlow backbone and segments 24 vehicle parts. The dataset used for this task is the open source PASCAL Parts dataset [17], along with some hand labelled dataset. The training ran for around 8 h for 30000 iterations with a lr of 0.0001. The mean Intersection over Union (mIOU) was 0.65 (Fig. 2).

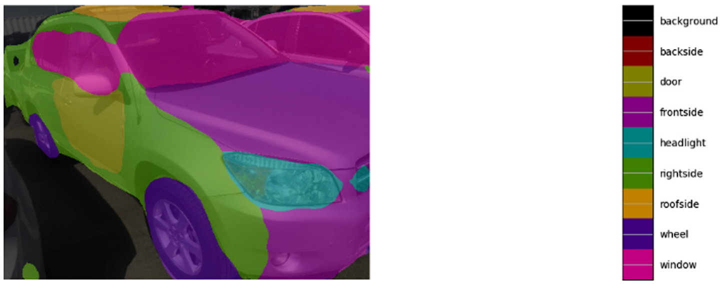

Fig. 2. Output of vehicle parts detection model

Vehicle Damage Segmentation. The vehicle damage segmentation model segments six different types of damages in the vehicle which are dents, scratches, misalignments, missing parts, crush/crumple and tear. This damage segmentation model helps to localize and classify the damages across the vehicle body. It should be noted that count of damage segments is intrinsic in this model, but scratches are a special case where detecting each line of the scratch is not practically possible by our deep learning model since the dataset was not labeled to incorporate such fine scratch lines. Thus, we use a Line Detection Algorithm, explained in the next section, which is used to overcome this problem specific to scratch damage types. Around 15000 images, annotated and verified by domain experts, were trained on a DeepLabv3+ [11] with Xception-65 architecture on Tensorflow backbone. The training used a batch size of 8 and a step lr with an initial lr of 0.001 and lr decay of 0.1 after every 10000 steps. The training ran for 30000 iterations over a span of 8 h. The mIOU was calculated per damage class as well as on the overall dataset. The overall mIOU achieved was 0.52. The final consolidation of outputs is done by performing intersection of the damage segmentation maps to the part segmentation maps. This module is helpful to provide damage count and dimension per damage per part of the vehicle.

Line Detection Algorithm. As discussed, above, the Vehicle Damage Segmentation Model detects the segmentation map of the detected damages. The output of this model can be used directly for damages such as dents, crush, crumple, tear, misalignments etc., where the customer charge out is calculated based on the damage area. However, for scratches, the focus is on scratch length rather than the total area. Hence, refinement is needed to further localize the scratch lines since the actual scratches are usually a small fraction of the larger detected scratch blob. Figure 3 shows the output of scratch lines refined from a scratch blob. A line detection algorithm is run on the retrieved scratch blobs. The line segment detector looks through the scratch blob and localizes the exact regions where the scratches are present so that the customer can be charged accurately based on scratch lengths. The OpenCV implementation of the probabilistic Hough Line algorithm [14, 15] with a minimum line length of 15 and maximum line gap of 5 is found to give the best performance on this dataset. Line detection algorithms detect straight lines only. Thus, for curved scratches, we take the summation of smaller line segments along the curved scratch lines.

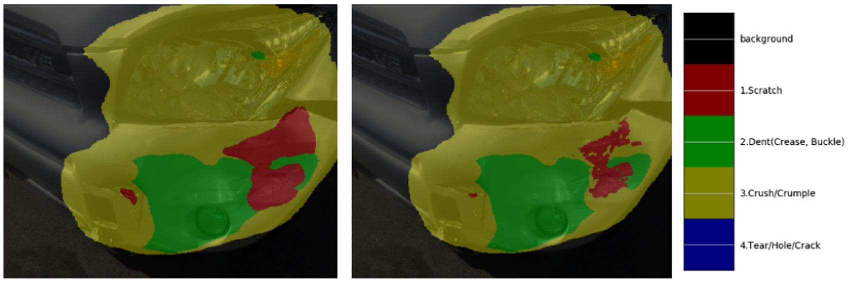

Fig. 3. (a) Output of damage detection (left) (b) Scratch lines from detected scratch blob (right)

3.2 Damage Dimension Estimation

Our dimension detection module contains multiple use case specific approaches to get the dimensions of the damages based on the view of the car visible. The main challenge to estimate the physical dimensions of any object from an image is to find a reference object with known dimensions in the image. Reference objects can then be used to find the scale which is defined as the pixel to inch ratio of the object. The scale can be then used to find the physical dimensions of the damage. The above steps, however, are under the assumption that both the object and the damage are in the same plane and the plane is orthogonal to the camera used to capture the image. This assumption can fail when the damages are present in the corner of the car, which is a point of intersection of two planes, and we address these issues below. We propose two algorithms to estimate the dimensions of the damages based on the reference object we chose, namely wheel and part as a reference object.

Wheel as a Reference Object. Wheels have a very distinct pattern irrespective of vehicle type. This allows the part detection model to detect it with high accuracy. Hence it is chosen to be a reference object in most of our scenarios. The physical dimensions of the wheel can be retrieved based on the car model information. However, challenges such as multiple wheel visibility at different locations, partial wheel visibility, tilted wheel, still exist. Below we enlist the various possibilities of wheel visibility and how we handle it to get the scale.

Detecting Wheel Pose and Visibility. To detect wheels, we use the output of the part segmentation model described in Sect. 3.1. The segment maps containing the wheel part is filtered for further use. We use shape descriptors to detect whether the wheel is partially or fully visible. We select HuMoments [16] as shape descriptors to identify the wheel pose and visibility. HuMoments are a set of seven numbers calculated using central moments of an image that are invariant to image transformations. The first six moments are invariant to translation, scale, and rotation, and reflection, while the seventh moment's sign changes for image reflection. We use OpenCV built-in HuMoments function to get the values of HuMoments for all the training samples in which any of the wheels of the car is visible. All seven moments together describe the shape of the portion of the wheel visible. An SVM model is trained on the HuMoment values of the wheel segmentation maps to classify them in the following categories: frontal pose-full visibility, frontal pose-partial visibility, tilted pose-full visibility, tilted pose-partial visibility.

Estimating Scale. Depending on the pose and the visibility of the wheel, we decide our strategy for estimating scale. In the first case, i.e. frontal pose-full visibility, the wheel segmentation map can be directly used to calculate the wheel dimensions in pixels. In the second case, i.e. frontal pose-partial visibility, we reconstruct the wheel using Hough Circles [13] and get the corresponding radius of the wheel in pixels. Since the actual dimension of the wheel is known to us, the pixel to inch ratio is calculated as the scale factor. The third case i.e. tilted pose-full visibility, occurs when the vehicle image is taken from non-frontal angles. In such a case, we apply perspective correction on the wheel segmentation map to transform it to a frontal pose and then get its accurate pixel dimensions. The last case i.e. tilted pose-partial visibility, is the most challenging of these cases. Here, we first use perspective correction on the wheel segmentation map and then reconstruct it using Hough Circles to get the scale factor.

Although, our algorithm performs well in the first three cases, the last case is subject to distortion due to perspective correction on the image. If the initial image is not captured in high resolution, it may be difficult to get enough features to correctly reconstruct the wheel from the perspective image owing to noise and distortion. Moreover, the Hough

Circle [13] reconstruction gives erroneous results for wheels which are less than 50% visible in the image. Work is in progress to address all wheel poses and visibilities and make the algorithm robust to distortion due to perspective correction.

Multiple Wheels Visibility. For a 4-wheeler passenger vehicle, we can safely assume that from any view angle of the vehicle at most two wheels are visible. However, both the wheels can have different scales if the plane of the wheels is not orthogonal to the point of view. So, if the damage is present in the plane as that of the wheels, the scale of the damage would be in between the scale of the two wheels. To overcome this, we incorporate perspective correction explained above, along with a simple approach based on weighted average. The scale factor of the damage is a weighted average of the scale of the two wheels detected, where the weight is inversely proportional to the distance of the damage to that wheel.

Part as a Reference Object. A fallback algorithm is used for images and planes where wheels are not visible or are not a reliable source for scale. Generally, such images involve zoomed-in images or images of front and rear end of the vehicle. In such cases, we cannot rely on wheel to retrieve a good scale estimate. For cases where front and rear end of the car is visible, we leverage the fact that we have the car width and the vehicle height information. Here, the scale is calculated from the physical dimensions of the car and its corresponding pixel width detected from part detection. The results of the different stages of the damage detection and estimation modules have been presented in Fig. 4.

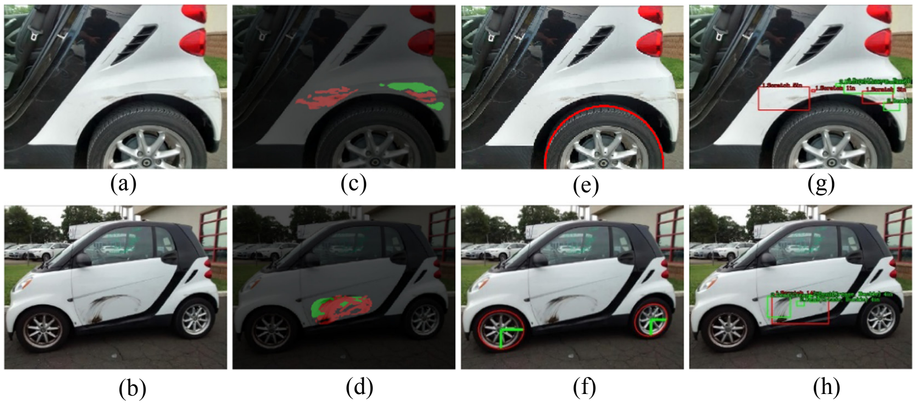

Fig. 4. Different stages of damage estimation pipeline. (a), (b) Two images of the same vehicle which focus on scratch and dents in different parts of the car body. (c), (d) Damage segmentation outputs where red indicates scratch and green indicates dent. (e), (f) Obtaining scale factor by reconstruction of partially visible wheel and multi wheel visibility. (g), (h) Damage lengths computed for the detected damages using the scale factor. (Color figure online)

3.3 Damage Duplicity Removal

As mentioned in earlier sections, we assume that customer captures multiple pictures of the vehicle and the damages. Since the image of a car is captured from multiple angles, there is a high probability that the same damage may be present in multiple images. Since our deep learning models are not trained to identify whether it is the same damage or a different one, we must incorporate post processing for damage duplicity removal. To address this issue, we use the approach of image stitching, going pairwise and finding damages which are the same across multiple images, as shown in Fig. 5. Once this is done, we stitch the relevant images together and map the damage information on the stitched image so that each unique damage is counted just once. This ensures that customers are not charged multiple times for the same damage.

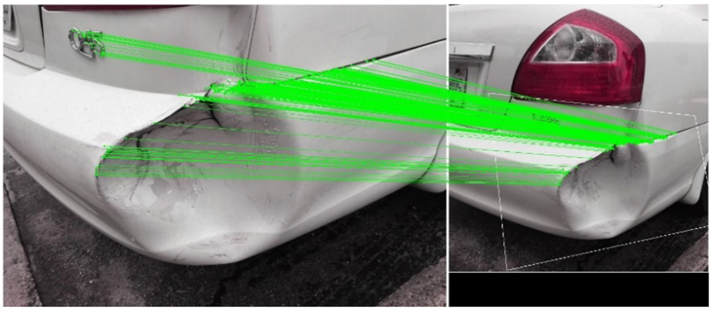

Fig. 5. Damage duplicity removal using pairwise image registration

4 Experimentation and Results

4.1 Evaluation of Damage Detection Module

For evaluation of vehicle parts segmentation and the vehicle damage segmentation model, we use approximately 15000 images. Each vehicle damage detection case contains around 16 images of the same vehicle. The images are generally taken from 8 different angles with approximately 45 degrees shift and with variable zoom. This rich dataset was used to validate our pipeline components and qualitatively assess the damage segmentation and parts detection model. Brief information about the various training datasets and evaluation metrics of the components of Damage Detection pipeline are provided below in Table 1.

4.2 Evaluation of Damage Estimation Module

Since there are no publicly available datasets which record damage lengths in vehicle damages, the evaluation of the damage estimation module was done through in-house data collection. Once collected, the damages were identified and the measured manually.

Table 1. Image dataset used for components in damage detection pipeline model training

Model name	Dataset size	Class names	Performance
Relevance filter	~5K Hand labeled dataset	Relevant, Irrelevant	Validation Accuracy - 91%
Parts segmentation	PASCAL Parts (Only Cars) + ~1K hand labeled image segmentation data [17]	24 part names from PASCAL Parts for Cars	Validation mIOU - 0.65
Damage segmentation	~15K Hand labeled image segmentation dataset	Scratch, Missing Dent, Crush, Tear, Misalignment	Overall Validation mIOU - 0.52

Scratch lengths were documented for around 50 samples. The same 50 samples were then given as an input to our damage estimation pipeline. The pipeline failed to perform on 4% of the cases due to challenges such as complex angles and zoom variations in the captured images. The reflection and dirt present on vehicle parts also lead to 21% of false positives.

5 Conclusion and Future Work

In this paper, we have shown that the current manual process of damage analysis in the Auto Finance industry, can be automated with our pipeline. We have also shown the importance of dimension measurement of the damages in calculating the charges to the customer. To address this, we have briefly presented our damage detection pipeline, and more importantly described our scaling strategy, i.e. using segmented wheels to get physical dimensions of the damages. We also demonstrated solutions to overcome challenges like partial wheel visibility with perspective tilts and multiple wheels visibility issues. Overall, the proposed approach provides insight to build a fast and accurate scaling system for damage analysis which can be of use with any of the damage analysis pipeline. Our solution is scalable across multiple domains and easily configurable for various business cases.

Possible ways of improvement of mIOU of our segmentation models include training our models with diverse and tightly annotated datasets. Experimentation with different semantic segmentation architectures with further hyperparameter tuning will also improve our semantic segmentation output. Ongoing work also targets to make the damage duplicity reduction module more accurate. Future work includes application of the developed techniques on live video streams so that damage estimates can be generated in real time.

Acknowledgment. The authors would like to sincerely thank Vikram Mahidhar, Sreekanth Menon, Amit Arora, Siva Tian, Krishna Dev Oruganty, Arun Pavuri and Rajat Katiyar for their help and support during the project development. We would also like to extend our thanks to the Genpact Product Engineering team and domain experts for helping in productizing the solution.

References

1. Samadzadegan, F., Rastiveisi, H.: Automatic detection and classification of damaged buildings, using high resolution satellite imagery and vector data. Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci. **37**, 415–420 (2008)
2. Kouchi, K., Yamazaki, F.: Damage detection based on object-based segmentation and classification from high resolution satellite images for the 2003 Boumerdes, Algeria earthquake. In: Proceedings of the 26th Asian conference on Remote Sensing, Hanoi, Vietnam (2005)
3. Jayawardena, S., et al.: Image based automatic vehicle damage detection. Ph.D. thesis, Australian National University (2013)
4. Patil, K., Kulkarni, M., Sriraman, A., Karande, S.: Deep Learning Based Car Damage Classification, pp. 50–54 (2017). https://doi.org/10.1109/icmla.2017.0-179
5. Krizhevsky, A., Sutskever, I., Hinton, G.: ImageNet classification with deep convolutional neural networks. In: NIPS (2012)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, pp. 770–778 (2016)
7. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, pp. 2818–2826 (2016)
8. Chollet, F.: Xception: deep learning with depthwise separable convolutions. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, pp. 1800–1807 (2017)
9. Girshick, R.: Fast R-CNN. In: 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, pp. 1440–1448 (2015)
10. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, pp. 779–788 (2016)
11. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs (2016)
12. Arun, P., Abhilash, N.V.S.: Deep learning and computer vision based damage analysis for auto insurance claims. In: ICBAI (2018)
13. Abhilash, N.V.S., Adrita, B., Ankit, S., Arun, P.: AI driven automatic vehicle damage assessment for auto insurance and auto finance industry. In: ICBAI (2019)
14. Cha, J., Cofer, R.H., Kozaitis, S.P.: Extended hough transform for linear feature detection. Pattern Recogn. **39**(6), 1034–1043 (2006)
15. Duan, D., Xie, M., Mo, Q., Han, Z., Wan, Y.: An improved hough transform for line detection. In: 2010 International Conference on Computer Application and System Modeling (ICCASM 2010) (2010)
16. Huang, Z., Leng, J.: Analysis of Hu's moment invariants on image scaling and rotation. In: Proceedings of 2nd International Conference on Computer Engineering and Technology (ICCET), vol. 7, pp. V7-476 (2010)
17. Chen, X., Mottaghi, R., Liu, X., Fidler, S., Urtasun, R., Yuille, A.: Detect What You Can: Detecting and Representing Objects using Holistic Models and Body Parts (2014)