Computer-Vision Based UAV Inspection for Steel Bridge Connections

JI YOUNG LEE, CHUNGWOOK SIM, CARRICK DETWEILER and BRENDAN BARNES

ABSTRACT

Corrosion on steel bridge members is one of the most important bridge deficiencies that must be carefully monitored by inspectors. Human visual inspection is typically conducted first, and additional measures such as tapping bolts and measuring section losses can be used to assess the level of corrosion. This process becomes a challenge when some of the connections are placed in a location where inspectors have to climb up or down the steel members. To assist this inspection process, we developed a computer-vision based Unmanned Aerial Vehicle (UAV) system for monitoring the health of critical steel bridge connections (bolts, rivets, and pins). We used a UAV to collect images from a steel truss bridge. Then we fed the collected datasets into an instance level segmentation model using a region-based convolutional neural network to train characteristics of corrosion shown at steel connections with sets of labeled image data. The segmentation model identified locations of the connections in images and efficiently detected the members with corrosion on them. We evaluated the model based on how precisely it can detect rivets, bolts, pins, and corrosion damage on these members. The results showed robustness and practicality of our system which can also provide useful health information to bridge owners for future maintenance. These collected image data can be used to quantitatively track temporal changes and to monitor progression of damage in aging steel structures. Furthermore, the system can also assist inspectors in making decisions for further detailed inspections.

Ji Young Lee, Department of Computer Science and Engineering, University of Nebraska-Lincoln, Lincoln, NE 68588, U.S.A.. Email: jiyoungl@cse.unl.edu

Chungwook Sim, Department of Civil Engineering, University of Nebraska-Lincoln, Omaha, NE 68182, U.S.A.. Email: csim@unl.edu

Carrick Detweiler, Department of Computer Science and Engineering, University of Nebraska-Lincoln, Lincoln, NE 68588, U.S.A.. Email: carrick@cse.unl.edu

Brendan Barnes, Department of Civil Engineering, University of Nebraska-Lincoln, Lincoln, NE 68588, U.S.A.. Email: brendan.barnes@huskers.unl.edu

INTRODUCTION

Corrosion is one of the most typical bridge deficiencies shown in steel bridge members [1–3]. In most cases, visual inspection is typically conducted first. However, where there is severe corrosion, further assessment may be required, such as tapping the bolts or measuring section losses to evaluate the severity of corrosion damage. This process becomes a challenge when some of the bolts, rivets, or pins are placed in a location where inspectors have to climb up or down the steel members [4–6]. To assist this inspection process, we are using a UAV to collect image data and automate the process in identifying critical connecting members and the level of corrosion damage. We collected 80 to 100 images from an large-scale structures lab and a steel truss bridge under service in Elkhorn, Nebraska. These images were used to train our computer-vision based system to automatically identify the steel connections and deficiencies within a given image.

Unlike cracks observed in concrete bridge decks, corrosion damage on steel members and connections are more challenging to detect because the level of corrosion may vary by the color depending on the deterioration level (light vs severe corrosion). In addition, the location and area of corrosion spread out on steel members may vary by bridges. Simple image processing methods or machine learning based detectors trained through user-defined descriptors do not perform well in such complex conditions. Therefore, in this paper, we implemented Mask R-CNN [7, 8], which can extract rich features without any human-crafted engineering process.

Many of the current practices in assessing bridge health deficiencies often exclusively rely on qualitative and subjective data provided through human inspections. With limited budget and resources, it is challenging to monitor the conditions of the corrosion damage in many of these steel bridges and to track the temporal and spatial change in damage. Our computer-vision based UAV sensing system can assist the inspection process and collect data which can be used later to monitor the progression of damage in aging steel structures.

INSTANCE SEGMENTATION

Instance segmentation aims to identify meaningful objects in the image which provides a pixel-wise location and information of each individual instance [7, 9]. However, detecting every single object from an image is a challenging topic in the computer vision domain because it requires the correct detection of all objects in one image while segmenting each instance and the image itself may look very different depending on the light conditions, present shadows, or the angle the image was taken. The need for identifying the precise location of each instance has motivated many researchers to develop instance segmentation models. Mask R-CNN [7] is one of the state-of-the-art algorithms for conducting instance level segmentation for a given image. Mask R-CNN is an extended version of Faster R-CNN [10] which can extract features and Region of Interests (RoIs). In the Mask R-CNN, once the features and RoIs are identified from a given image, network head branches including mask prediction, bounding box regression, and label classification are performed simultaneously. Algorithm 1 shows the pseudo code of our Mask R-CNN model for the segmentation process with given image sets.

Algorithm 1 Mask R-CNN

SET CONFIGURATION()
if mode $= training$ **then**
    CREATE MODEL()
    LOAD WEIGHTS($w_{pretrain}$) ▷ w: weights
    $D_{train,val}$ LOAD DATASET($train, val$) ▷ D: dataset
    TRAIN($D_{train,val}, \eta, e$) ▷ η: learning rate, e: epoch
else if mode $= inference$ **then**
    LOAD MODEL()
    LOAD WEIGHTS($w_{checkpoint}$)
    $D_{test} \leftarrow$ LOAD DATASET($test$)
    RUN DETECTION(D_{test})
end if
function LOAD DATASET($data$)
    $I, L \leftarrow$ LOAD IMAGE ANNOTATION FILE($data$) ▷ I: raw image, L: labeled data
    while l in L **do** ▷ l: label of each image
        while i in l and not $NULL$ **do** ▷ i: instance of each label
            $Mask \leftarrow$ DRAW MASK($I, i_{x,y}$) ▷ $i_{x,y}$: x,y-coordinate of labels
        end while
        $D \leftarrow I, Mask$
    end while
    return D
end function
procedure TRAIN($D_{train,val}, \eta, e$)
    while $i < e$ **do**
        $F_{map} \leftarrow$ BACKBONE($D_{train,val}$)
        $G_{train,val}$ REGION PROPOSAL NETWORK($D_{train,val}, F_{map}$) ▷ G: generator
        CLASSIFIER($G_{train,val}, F_{map}, \eta$)
        SAVE CHECKPOINTS()
    end while
end procedure
function REGION PROPOSAL NETWORK(D)
    $H \leftarrow$ GENERATE ANCHORS(D) ▷ H: anchors
    while $i < length(D)$ **do**
        $I, GT_{class,box,mask}$ LOAD IMAGE GT(D) ▷ GT: ground-truth
        $H_{refined}, RPN_{box} \leftarrow$ BUILD RPN TARGET($I, H, GT_{class,box}$)
        $RPN_{RoIs} \leftarrow$ GENERATE RANDOM ROIS($I, GT_{class,box}$)
        $RoIs, MRCNN_{class,box,mask}$ BUILD DETECTION TARGET($RPN_{RoIs}, GT_{class,box,mask}$)
        $batch$ $I, RoIs, RPN_{match,box}, GT_{class,box,mask}, MRCNN_{class,box,mask}$
    end while
    $G \leftarrow batch, H_{refined}$
    return G
end function
procedure RUN DETECTION(D)
    $H \leftarrow$ GENERATE ANCHORS(D)
    $H_{refined}, RPN_{box} \leftarrow$ BUILD RPN TARGET($I, H, GT_{class,box}$) ▷ T: target
    $H_{detect} \leftarrow$ RUN GRAPH($I, H_{refined}, RPN_{box}$)
    $H_{top} \leftarrow$ SORT(H_{detect})
    CLASSIFIER(H_{top})
    DISPLAY RESULT()
end procedure

Details of the Mask R-CNN used in this paper can be found in reference [8]. We customized the functions LOAD DATASET and RUN DETECTION tailored to our application. To predict the locations of the target objects, the Mask R-CNN extracts features from the given sets of images with convolutional backbone. The backbone architecture initially provides the most prominent features from a given set of data during the training phase. The extracted feature map is shared with the region proposal network (RPN) and the network head. The RPN produces the expected RoIs for each object by overlapping the generated anchors and ground truth regions. The RPN generates random regions by pooling the image object only to the foreground or background, regardless of its classes as a bounding box. However, as RPN produces multiple sizes of boxes, this process induces a problem to the convolutional neural networks based classifiers which only accepts images with fixed-size. Mask R-CNN mitigates this problem by employing an interpolation layer which modifies every target box to have an identical size. The interpolated features are shared with the network head branches which predicts the class labels, binary masks, and bounding boxes of each RoI. The Mask R-CNN uses losses for training from each branch as follows:

$$L = L_{cls} + L_{box} + L_{mask}$$

where L_{cls} indicates the log loss of classification prediction, L_{box} indicates the loss of bounding box refinement, and L_{mask} indicates the average cross-entropy loss of each RoI mask.

EXPERIMENTAL RESULTS

Experimental Setup

We collected images as shown in Figure 1 from the bridge site with a quadcopter UAV manually controlled by two remote pilots. The pairs of collected images and their labeled masks were carefully prepared with guidance and assistance from domain experts in Civil Engineering. We fed these data into our Mask R-CNN model. The entire training procedure was performed as end-to-end training, which does not require any pre or post-processing for selected labeled datasets. Based on the loss curves of the training and validation set, we fine-tuned our hyper-parameters such as the backbone architecture, pretrained weights, learning rate, learning momentum, weight decay, anchor scale, and the anchor size as shown in Table I.

TABLE I. FINE-TUNED HYPER-PARAMETERS

Mask R-CNN	
Learning rate	0.01
Number of epochs	300
Minimum confidence	0.6
Backbone architecture	ResNet 101 [11]
Pre-trained weights	COCO dataset [12]
Loss weights for each branch	equal
Number of classes	$2-3$

Figure 1. Example images from the outdoor dataset.

The implemented model was trained on an Ubuntu $16.04.06$ LTS with four Tesla $V100$ GPUs with a $16GB$ memory for each GPU. We used an Intel Xeon $E5-2698$ CPU with $v.4$ $2.2GHz$ with $256GB$ $LRDIMM$ $DDR4$ system memory, which took around $63sec$ per epoch for training each dataset.

Dataset

We collected two types of image datasets. For our initial testing, we collected images from an indoor lab facility. A total number of 80 photos of a steel frame in the Large-Scale Structures Laboratory of the University of Nebraska-Lincoln were collected to train our model. The images have steel bolts, nuts, and some light corrosion on the member. Each image is in a JPG format and contains a three-channel RGB color space. The dimension of the image is $3,024$ by $4,032$ pixels. Within the guidance of the domain experts, we tagged each image with polygon shaped labels of two classes: bolt and corrosion. The camera setup used in collecting the photos had a shutter speed of $1/4sec$, $4mm$ focal length, an ISO value of 125, and an f stop of $f/1.8$.

After the initial training, we collected our second dataset of outdoor images from an actual bridge site. We used the DJI Mavic Pro quadcopter UAV to collect the outdoor images from the bridge site in Elkhorn, Nebraska. On the day of data collection, the maximum wind speed was $16MPH$ and the visibility was $10SM$. Two certified remote pilots maneuvered the UAV by standing 3 meters away from the bridge site, and a visual observer was present during the flight to avoid crashing in data collection. The built-in camera gimbal on the UAV provides -90 to 30 pitch and 0 to 90 roll degree both in horizontal and vertical direction. The camera setup used in collecting the photos had an average shutter speed of $1/516sec$, $4.73mm$ focal length, an ISO value of 100, and an f stop of $f/2.2$.

A total number of 100 images directly taken from the bridge site were used to identify rivets, bolts, and pins members on the steel truss bridge. Each image is in a JPG format with three-channel RGB color space with an image size of approximately $3,000$ by $4,000$ pixels. For the outdoor experiment dataset, we focused mainly on identifying locations where severe corrosion and "bleeding" was observed between the connections and the main steel member where iron-oxide corrosion rust is dripping down.

Evaluation

The loss curves of training and validation set per epoch for the outdoor dataset is shown in Figure 2 and the visual segmentation results from the indoor and outdoor dataset is shown in Figure 3.

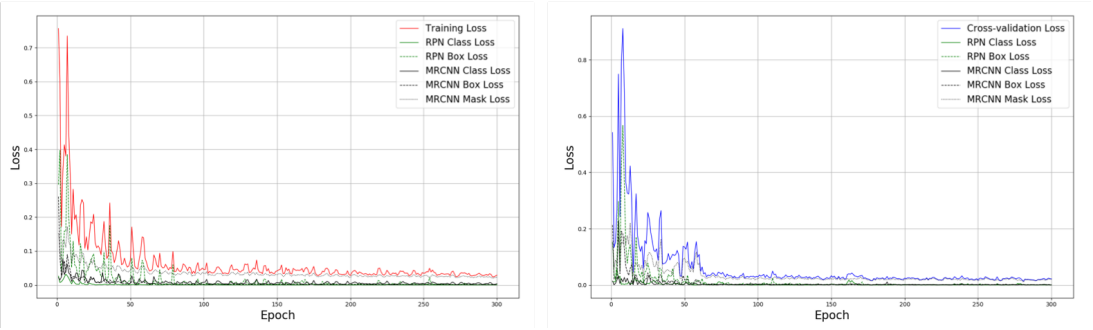

Figure 2. Training and validation loss curves per epoch from the outdoor dataset.

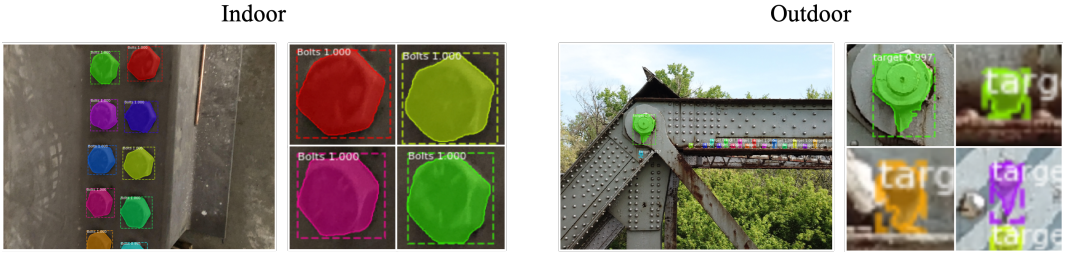

Figure 3. Visual segmentation results from the datasets.

Although the training and validation graphs fluctuate in some epochs in Figure 2, both graphs reached a plateau with reasonable losses without showing overfitting after more than 100 epochs. The left image in Figure 3 shows the prediction results of the indoor dataset with the bolt, and the right image shows the prediction results of the outdoor dataset with rivets and pins that show rust bleeding. In both images, different colors indicate that the objects are identified as different instances.

We used an average precision (AP) score for all classes that we defined for both datasets under a particular Intersection over Union (IoU) threshold to evaluate our model [13]. The IoU metric quantifies the percent overlap between the target mask and the prediction outputs. Table II shows performance evaluation results based on the AP scores with varying IoU thresholds for each dataset.

We observed that the fine-tuned model for the indoor dataset shows better performance than the outdoor dataset because the AP score was not decreased with a stricter IoU threshold. However, the model reveals that it cannot currently detect all instances with a stricter threshold for the outdoor dataset. This is probably because the indoor dataset was created in a controlled lighting condition with identifiable hexagon shaped of the object compared to the bleeding rust products from the outdoor dataset. In addition, the outdoor dataset suffers from the imbalance between classes since the number of rivets within the image was dominating compared to the number of bolt and pin members.

TABLE II. PERFORMANCE EVALUATION BASED ON AP SCORE.

Dataset	$AP_{.50}$	$AP_{.75}$
Indoor	0.90	0.90
Outdoor	0.89	0.36

These results indicate the need for a sufficient and more balanced dataset for outdoor experiments.

We also used a recall metric based on the confusion matrix to measure the missing percentage of the target component. In this evaluation, the ratio of each classified instance to the total number of instances within the image is calculated. The performance evaluation results using the confusion matrix and the recall are shown in Table III. Each element of the matrix was derived from the average value from all corresponding classes of each image. The equation used for the recall is as follows:

$$Recall = 1 - \bar{FN}/(\bar{FN} + \bar{TP})$$

where $\bar{FN}$ is the average false negative and $\bar{TP}$ is the average true negative.

The performance evaluation results demonstrate that our model can predict most of the target classes for both indoor and outdoor datasets with accuracy above 90%.

CONCLUSION

We implemented the Mask R-CNN model to train our machine to automatically detect steel connection members (bolts, rivets, and pins) and corrosion deficiencies from a given image. We initially trained our model through an indoor dataset with bolts and light corrosion on steel members. Then, we collected a second dataset from a bridge site using a UAV. The experimental results show that our system is capable of identifying corrosion deficiencies on steel members. However, we need to increase the number and types of the dataset from field to train our model with better and improved accuracy. We expect that such computer-vision based UAV inspection may assist the current inspection process. In addition, the collected image data can be used to track temporal and spatial changes quantitatively to monitor the progression of any damage in aging steel structures.

ACKNOWLEDGEMENT

This study was financially supported by the NSF BD Spokes Program (Award #1762034). Their support is gratefully acknowledged.

TABLE III. EVALUATION BASED ON CONFUSION MATRIX AND RECALL.

			Actual		*Recall*
			Positive	Negative	
Indoor	Predicted	Positive	0.99	0.13	0.96
		Negative	0.05	0.93	
Outdoor		Positive	0.91	0.05	0.91
		Negative	0.09	0.95	

REFERENCES

1. Shanthakumar, P., K. Yu, M. Singh, J. Orevillo, E. Bianchi, M. Hebdon, and P. Tokekar. 2018. "View planning and navigation algorithms for autonomous bridge inspection with uavs," in *International Symposium on Experimental Robotics (ISER)*.
2. Khan, F., A. Ellenberg, M. Mazzotti, A. Kontsos, F. Moon, A. Pradhan, and I. Bartoli. 2015. "Investigation on bridge assessment using unmanned aerial systems," in *Structures Congress 2015American Society of Civil Engineers*.
3. Correia, M., H. Perneta, M. Salta, L. Gaillet, H. Patricio, and F. Schoefs. *Duratinet : Maintenance and Repair of Transport Infrastructure - TECHNICAL GUIDE: Steel Structures - Part III, Deterioration*, LNEC Publisher, LISBOA, pp. 86.
4. Narazaki, Y., V. Hoskere, T. A. Hoang, and B. F. Spencer Jr. 2018. "Automated vision-based bridge component extraction using multiscale convolutional neural networks," *arXiv preprint arXiv:1805.06042*.
5. Lattanzi, D. and G. Miller. 2017. "Review of robotic infrastructure inspection systems," *Journal of Infrastructure Systems*, 23(3):04017004.
6. Khaloo, A., D. Lattanzi, K. Cunningham, R. DellAndrea, and M. Riley. 2018. "Unmanned aerial vehicle inspection of the Placer River Trail Bridge through image-based 3D modelling," *Structure and Infrastructure Engineering*, 14(1):124–136.
7. He, K., G. Gkioxari, P. Dollár, and R. Girshick. 2017. "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969.
8. Abdulla, W. 2017, "Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow," https://github.com/matterport/Mask$_RCNN$.
9. Girshick, R., J. Donahue, T. Darrell, and J. Malik. 2014. "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587.
10. Ren, S., K. He, R. Girshick, and J. Sun. 2015. "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, pp. 91–99.
11. He, K., X. Zhang, S. Ren, and J. Sun. 2016. "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.
12. Lin, T.-Y., M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. 2014. "Microsoft coco: Common objects in context," in *European conference on computer vision*, Springer, pp. 740–755.
13. Csurka, G., D. Larlus, F. Perronnin, and F. Meylan. 2013. "What is a good evaluation measure for semantic segmentation?." in *BMVC*, Citeseer, vol. 27, p. 2013.