TRANSPORTATION POOLED FUND PROGRAM
QUARTERLY PROGRESS REPORT

Lead Agency (FHWA or State DOT): Kansas DOT

INSTRUCTIONS:
Lead Agency contacts should complete a quarterly progress report for each calendar quarter during which the projects are active. Please provide a project schedule status of the research activities tied to each task that is defined in the proposal; a percentage completion of each task; a concise discussion (2 or 3 sentences) of the current status, including accomplishments and problems encountered, if any. List all tasks, even if no work was done during this period.

Transportation Pooled Fund Program Project # *(i.e, SPR-2(XXX), SPR-3(XXX) or TPF-5(XXX)* **TPF-5(535)**	**Transportation Pooled Fund Program - Report Period:** ☐Quarter 1 (January 1 – March 31) ☐Quarter 2 (April 1 – June 30) ☒Quarter 3 (July 1 – September 30) ☐Quarter 4 (October 1 – December 31)

TPF Study Number and Title: **TPF-5(535): Human-centered Steel Bridge Inspection enabled by Augmented Reality and Artificial Intelligence**

Lead Agency Contact: **David Behzadpour**	**Lead Agency Phone Number:** **785-291-3847**	**Lead Agency E-Mail** David.Behzadpour@ks.gov
Lead Agency Project ID: Click or tap here to enter text.	**Other Project ID (i.e., contract #):** Click or tap here to enter text.	**Project Start Date:** 10/1/2024
Original Project Start Date: **10/1/2024**	**Original Project End Date:** **9/30/2027**	**If Extension has been requested, updated project End Date:** Click or tap to enter a date.

Project schedule status:

☒On schedule	☐On revised schedule	☐Ahead of schedule	☐ Behind schedule

Overall Project Statistics:

Total Project Budget	Total Funds Expended This Quarter	Percentage of Work Completed to Date
$600,000	$19,542	25%

Project Description:

The main objective of this proposed research is to provide state DOTs practical tools for supporting human-centered steel bridge inspection with real-time defect (e.g., fatigue cracks and corrosion) detection, documentation, tracking, and decision making. The proposed research will not only bridge the gaps identified in the IDEA project, but also expand the existing capability by developing AI algorithms for crack and corrosion detection. In addition to AR headsets, the project will also develop AR-based inspection capability using tablet devices. The tablet device can be used to perform AR-based inspection directly in a similar way to the AR headset. It can also leverage Unmanned Aerial Vehicles (UAV) for remote image and video acquisition during inspections, enabling bridge inspections from a distance in a human-centered manner.

Progress this Quarter
(includes meetings, work plan status, contract status, significant progress, etc.):

1. **Task 1: CV and AI algorithms for crack and corrosion inspection**
   This quarter, progress was made in improving crack and corrosion segmentation through the combined use of tiling and ensemble learning. Tiling enhanced data quality by magnifying fine-scale damage features and increasing dataset size, with 4× tiling applied to corrosion images and 9× tiling applied to crack images, resulting in clearer and more balanced training data. For corrosion, ensemble learning across multiple deep learning models, including YOLOv8s, YOLOv8m, and YOLOv8x, showed that bagging strategy achieved the highest performance with an average IoU of 0.67, outperforming boosting (0.61) and stacking (0.55). For cracks, the ensemble evaluation revealed that boosting, which weighted stronger YOLOv8 models and incorporated confidence calibration, achieved the best performance in crack segmentation with an IoU of 0.49, while bagging and stacking produced lower and inconsistent results, underscoring the need for advanced fusion techniques to enhance detection sensitivity and consistency.
2. **Task 2: AR-based software for human-centered bridge inspection**
   AR infrastructure inspection tool development has continued with ongoing refinement of features on the Magic Leap 2 platform. Documentation/recording issues with the headset have been resolved, and new features have been introduced to provide additional utility for field inspectors. Updates have been made to our inference pipeline to allow for newer YOLO models to be used, allowing for better optimization and accuracy. Consistent persistent anchorage remains a challenge; current development is focused on achieving reliable stability across sessions.
   Meanwhile, testing with RGB-D camera has continued, with ongoing exploration of 3D reconstruction for deployment on UAV and mobile devices for more efficient inspection workflows.

Anticipated work next quarter:

1. **Task 1: CV and AI algorithms for crack and corrosion inspection**
   Next quarter, the work will focus on expanding and refining the segmentation pipeline through two main directions: (1) dataset enhancement, by incorporating additional corrosion and crack images from varied structures and applying advanced techniques to improve generalization; (2) model optimization, including fine-tuning ensemble strategies, exploring hybrid architectures, and evaluating light/medium weight deployment-ready models for real-time inspection.
2. **Task 2: AR-based software for human-centered bridge inspection**
   During the current reporting period, we plan to refine our implementation of Magic Leap's spatial anchors to ensure reliable persistence across inspections. This will allow us to being creating and integrating a database to store inference results that work seamlessly with our headset application. In parallel we aim to develop a solution for 3D localization and spatialization on UAV and mobile devices to lay the groundwork for future inspection tools.

Significant Results:

1. Task 1: CV and AI algorithms for crack and corrosion inspection

The accurate detection of cracks and corrosion in infrastructure requires methods that can capture fine-scale damage while maintaining robust performance across diverse datasets. Two strategies: tiling and ensemble learning were used to improve segmentation in this section.

Tiling addresses the challenge of large, high-resolution inspection images where cracks and corrosion occupy only small, localized regions. When analyzed globally, these features may be overlooked, leading to poor detection. By splitting images into fixed-size patches, tiling magnifies local features, expands dataset size without requiring new images, and reduces class imbalance by ensuring balanced representation of damaged and undamaged regions. In this step, the corrosion dataset was processed with 4× tiling, and the crack dataset was prepared using 9× tiling. This approach improved the visibility of small cracks and corrosion spots, providing the models with clearer and more consistent training data. Figure 1 illustrates a 4x tiling of the corrosion image.

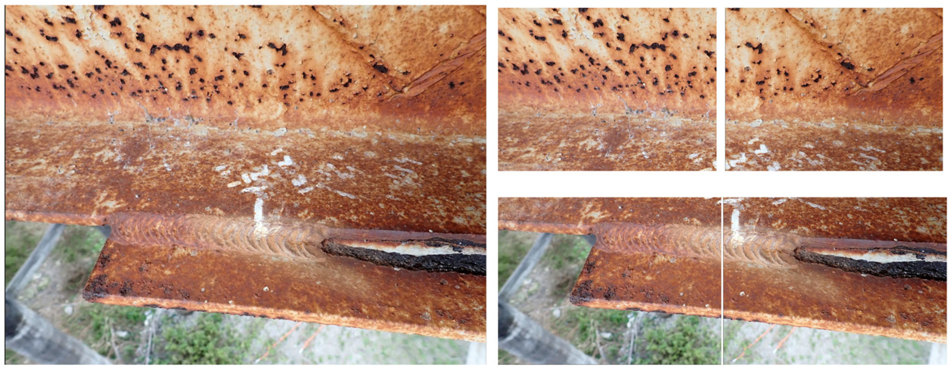


Figure 1. 4× tiling for a corrosion image

Ensemble learning is a machine learning strategy that combines the predictions of multiple models to produce a more accurate and robust outcome than a single model alone. Instead of relying on the strengths of just one architecture, ensembles aggregate diverse decision patterns, thereby reducing variance, mitigating overfitting, and lowering the risk of false negatives. In the context of crack and corrosion segmentation, ensemble learning was applied across three YOLOv8 variants (YOLOv8s, YOLOv8m, and YOLOv8x) each with different model capacities and feature extraction capabilities. The smaller YOLOv8s provides efficiency and speed, YOLOv8m balances accuracy with inference time, and YOLOv8x offers higher accuracy at the cost of computational overhead. Meanwhile, several ensemble learning strategies were explored: bagging, which trains models on different subsets of data and combines their predictions through majority voting or averaging; boosting, which sequentially re-weights misclassified samples to improve accuracy; and stacking, which uses a meta-model to learn the best way to combine outputs from base models. Among these, bagging proved most effective for corrosion, achieving higher IoU values by stabilizing predictions and enhancing generalization. Figure 2 depicts the ensemble model architecture used for damage segmentation.

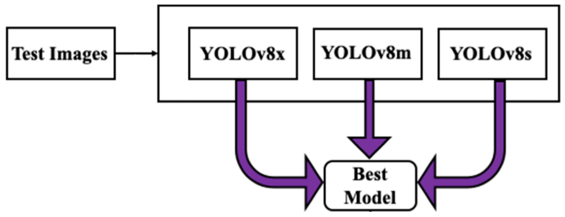


Figure 2. Ensemble model architecture for damage segmentation

Corrosion segmentation was performed on a dataset of 739 images, preprocessed with a 4× tiling strategy to improve the visibility of localized corroded patches that are sometimes difficult to separate from the background. Tiling increased the effective dataset size and ensured that more corroded regions were represented in training, allowing the models to learn more details. Three YOLOv8 variants (s, m, and x) were employed within different ensemble strategies to enhance performance. As summarized in Table 1, the results showed that bagging achieved the highest average IoU of 0.67, followed by boosting at 0.61 and stacking at 0.55. These outcomes indicate that bagging provided the most stable and accurate corrosion segmentation by reducing variance and improving generalization, and tiling played a crucial role in training corrosion patterns to the detection models. Figure 3 illustrates sample corrosion segmentation results.

Table 1. Performance results for Corrosion detection

Ensemble Mode	**Average IoU**
Bagging	0.67
Boosting	0.61
Stacking	0.55

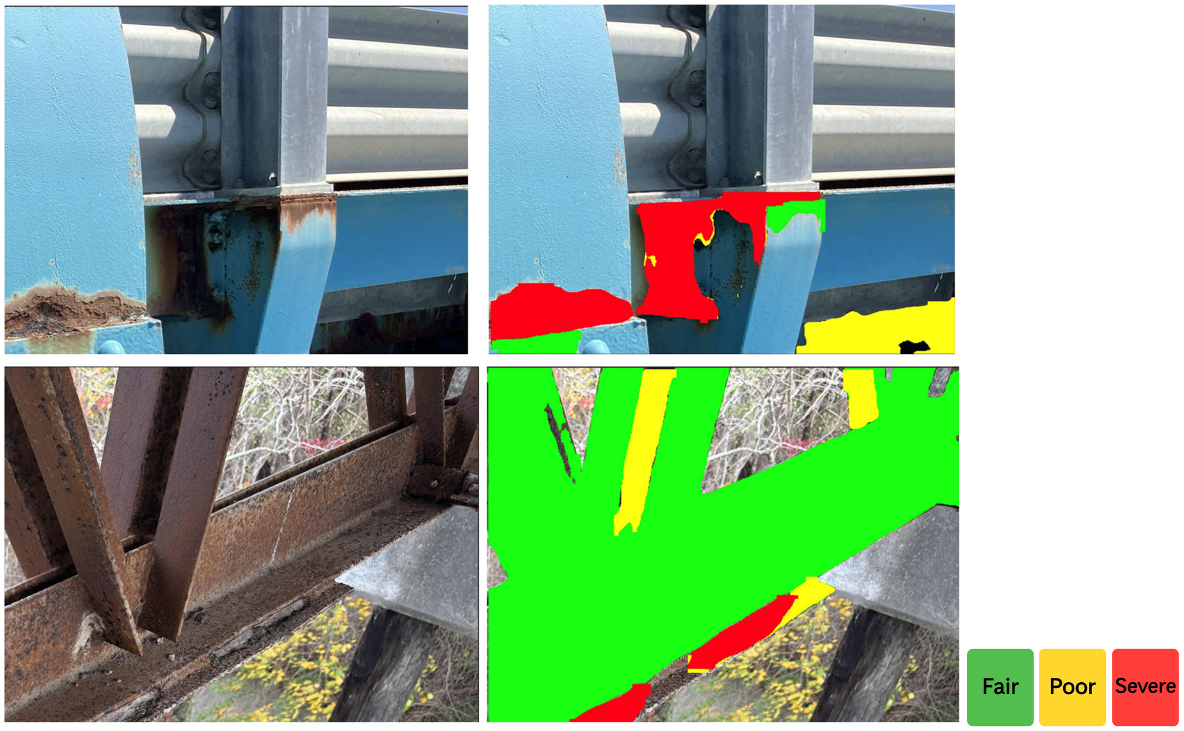


Figure 3. Example of corrosion segmentation results: original images (left) and model's output (right)

The ensemble evaluation demonstrated that different strategies yielded varying levels of performance in crack segmentation. Bagging, which equally averaged predictions across YOLOv8s/m/x models, provided modest improvements but was limited by noisy overlaps, resulting in relatively low IoU. Boosting, which assigned higher weight to stronger models (YOLOv8x) and incorporated confidence calibration, achieved the best overall performance with an IoU of 0.49, highlighting the value of weighted combinations in capturing fine-scale crack features. Stacking, initially implemented as a "best-confidence" selector, produced inconsistent results due to reliance on single predictions, with IoU around 0.37. Overall, the results summarized in Table 2 show that boosting offers the most reliable balance between detection sensitivity and robustness, while also emphasizing the need for more advanced fusion techniques, such as adaptive or confidence-weighted stacking, to further reduce false negatives and improve segmentation consistency across challenging test cases. Figure 4 shows sample crack segmentation results.

Table 2. Performance for the trained crack detection models

Ensemble Mode	Average IoU
Bagging	0.34
Boosting	0.49
Stacking	0.37

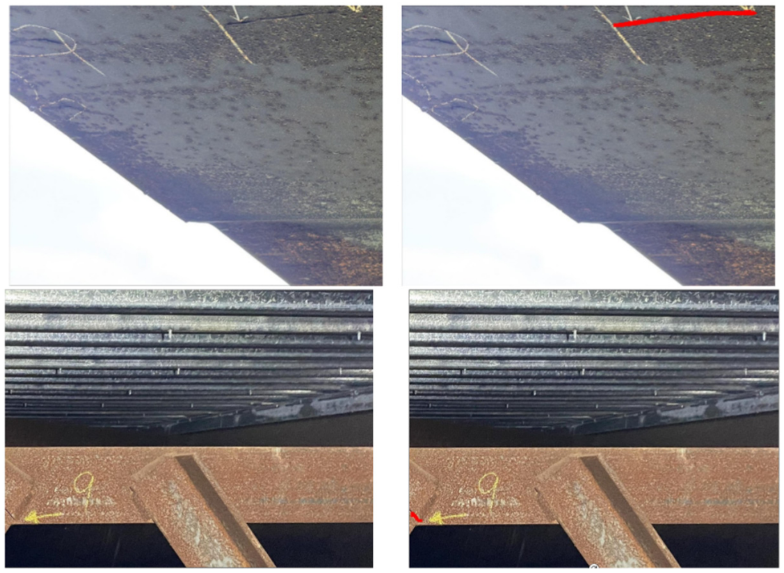


Figure 4. Example of crack segmentation results: original images (left) and model's output (right).

Together, tiling and ensemble methods can create a more effective pipeline: tiling enhances the data by emphasizing fine-scale features and increasing dataset samples, and ensemble learning maximizes accuracy and robustness by leveraging the strengths of different YOLOv8 models. These strategies can significantly advance the performance of automated crack and corrosion segmentation, setting the foundation for integration into other learning frameworks, dataset augmentation, and future deployment in real-world inspection systems.

2. Task 2: AR-based software for human-centered bridge inspection

Subtask 2.3: AR software environment for AR headset

Previously, we experienced issues with recording in-app view using the Magic Leap's built-in camera. Software conflicts arose between the application's camera usage and the process of capturing the user's perspective, this prevented reliable video recording. After much troubleshooting and communication with other Magic Leap application developers, the issue has been resolved, and the application can now be recorded for demonstration purposes without complication.

After previously identifying Magic Leap Spectator (Fig. 5) as a potential asset for our application, we have explored its functionality and confirmed its usefulness for cooperative inspections. The necessary libraries have been updated, and the plugin has been successfully imported into our application and is now enabled. This feature allows multiple inspectors to simultaneously view the digital inferences produced by a single Magic Leap headset, improving collaboration during inspection tasks.

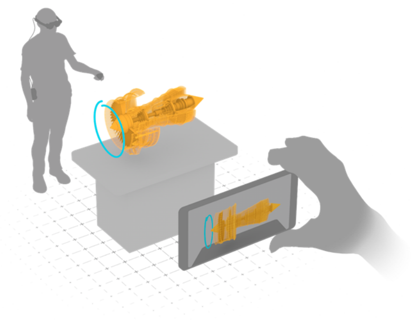
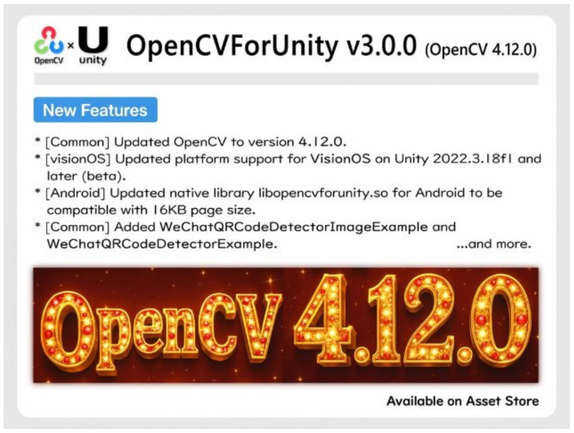


Figure 5. Magic Leap Spectator illustration

We have also updated our inference pipeline to support newer versions of YOLO models. The previous pipeline was limited to older versions of YOLO such as YOLOv8, restricting our opportunities for improved models with better performance and accuracy. With the release of the most recent update for OpenCVForUnity, shown in Fig. 6, we were able to integrate the new inference engine into the application. This update enabled support for the latest YOLO models such as YOLOv12 and has significantly improved the program's flexibility in model usage. Additionally, the new inference pipeline functions fully asynchronously and ensures that the application does not hitch when an inference is being produced.



OpenCVForUnity v3.0.0 (OpenCV 4.12.0)

New Features

* [Common] Updated OpenCV to version 4.12.0.
* [visionOS] Updated platform support for VisionOS on Unity 2022.3.18f1 and later (beta).
* [Android] Updated native library libopencvforunity.so for Android to be compatible with 16KB page size.
* [Common] Added WeChatQRCodeDetectorImageExample and WeChatQRCodeDetectorExample. ...and more.

OpenCV 4.12.0

Available on Asset Store

Figure 6. OpenCVForUnity Update

Up to this point, our application has utilized a segmentation model designed for fault and crack detection in concrete structures/surfaces. With the availability of new models targeting steel bridge fatigue and erosion, we have applied our updated inference pipeline to one of these new models and successfully demonstrated its use and effectiveness on the headset.

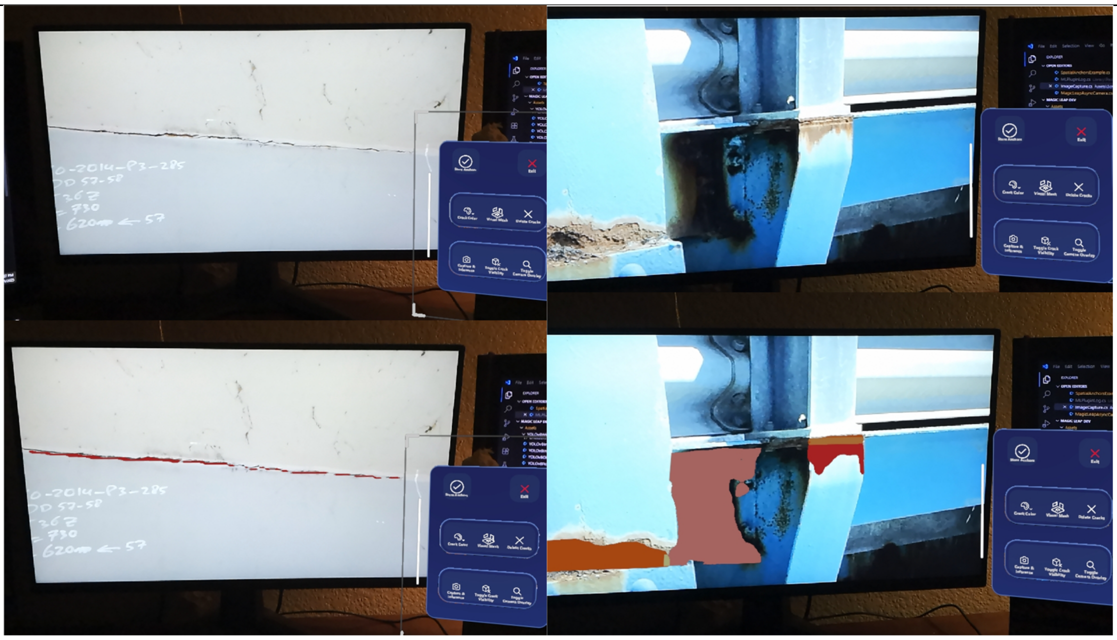


Figure 7. AR-based Inference of Steel Crack (left) and Corrosion (right)

Spatial anchors and persistent anchorage continue to be a key focus of development. We are actively working with Magic Leap's spatial anchor system, though challenges remain with achieving stable anchor storage and integrating the new functionality into the inference and projection pipeline. We consulted with the Magic Leap support team on their developer forum, where we were directed to review an example Unity file that utilizes the latest unity version (the same version that our app relies on). With their example as a template, we were able to adapt their implementation to our needs and successfully incorporate the functionality into our project.

Subtask 2.4: AR software environment for tablet device and UAV

In this quarter, we continued investigating the use of a depth camera (Intel RealSense D455) to construct a 3D model of the surrounding environment from a short video stream. This 3D model will enable high-quality anchoring of holograms to the real-time inspection video streamed from the UAV to the tablet device.

To evaluate the performance and feasibility of using the depth camera, we conducted a simple experiment with a backpack placed in the corner of a room. A 5-second video was first recorded, from which images and depth maps were extracted to generate a 3D model of the corner and the backpack. When a new point cloud was later acquired from the UAV with labeled damage features represented by the modem, the labeled points were mapped back onto the video to trace the region of interest.

A depth camera captures both RGB images and depth maps, which can be combined to generate colored 3D point clouds. From these point clouds, a 3D mesh model of a local region can be rapidly reconstructed. Figure 8 illustrates a 3D mesh model created from 12 frames captured by the depth camera within 5 seconds. The process enables the UAV to create a 3D model of the inspected region through a quick scan.

Once the 3D model is available, an inspection image, along with its depth map, can be captured. The image can be used to detect potential damages such as corrosion or cracking. Figure 9a and 9b show an example of an RGB image and its corresponding depth map, which can be used to generate a point cloud, as demonstrated in Figure 9c. In Figure 9c, several points are selected to label the modem which simulates damage features, and these points are then mapped back onto the image/video to annotate the modem within the 2D domain. The final result is presented in Figure 9d. In the next quarter, the annotation will be displayed to reflect the 3D nature of the feature when the camera view changes.

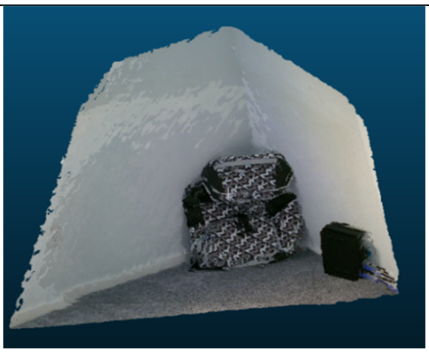
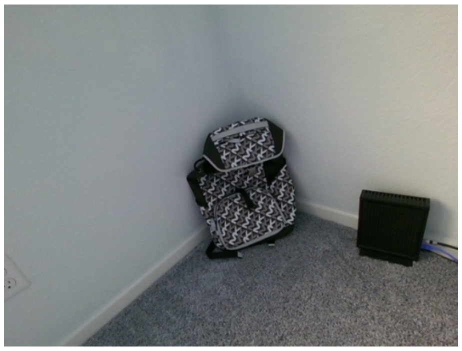
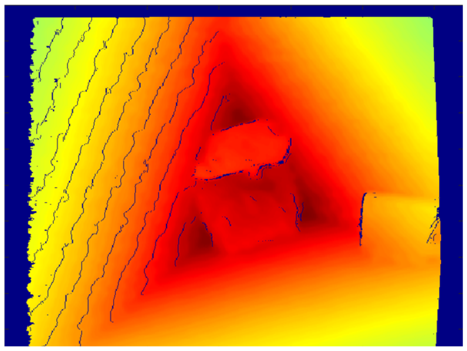
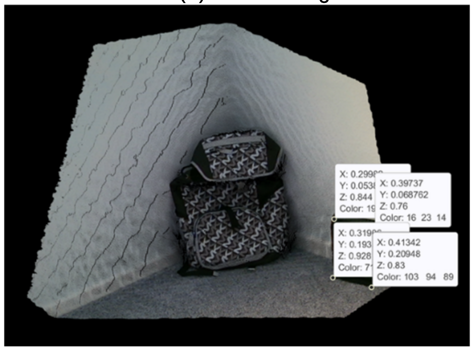


Figure 8. A sample 3D mesh model created by a short video from the RGB-D camera


(a) RBG image


(b) Depth map



(c) Constructed point cloud and selected target points

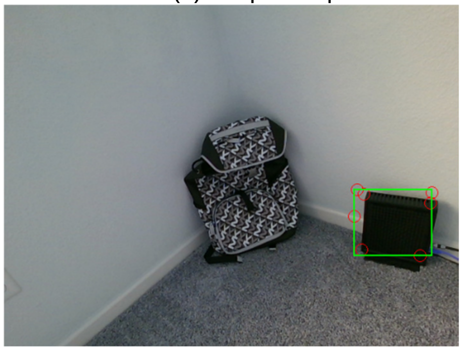

(d) A 2D image with anchored points and bounding box

Figure 9: Illustration of anchoring features from a new inspection image onto the 2D video

Circumstance affecting project or budget. (Please describe any challenges encountered or anticipated that might affect the completion of the project within the time, scope and fiscal constraints set forth in the agreement, along with recommended solutions to those problems).

N/A

Potential Implementation: