Deep Learning Machine Vision Solutions for Monitoring Safety Structures to Supplement Building Information Models

Tommi Aihkisalo
VTT Technical Research Center of Finland
Oulu, Finland
tommi.aihkisalo@vtt.fi

Anu Purhonen
VTT Technical Research Center of Finland
Oulu, Finland
anu.purhonen@vtt.fi

Ilkka Niskanen
VTT Technical Research Center of Finland
Oulu, Finland
ilkka.niskanen@vtt.fi

***Abstract*—Building Information Models (BIM), provide a static view on building structures on design and construction time. During construction time, the safety structures and their positioning among conditions are vital for the safety of the employees on site. The models usually lack the temporospatial information regarding non-static safety structures as maintaining of such information manually is laborious. Thus, automated and continuous safety structure monitoring is cost-efficient and vital keeping the model up-to-date while improving construction site safety and risk information sharing among the construction related personnel during the project. This paper researches of providing up-to-date and supplementary safety structure information to building models by means of various deep learning-based machine vision solutions. The machine vision tasks consist of deep learning safety related object detection in images, point-clouds, and further instance segmentation to enable safety structure fitness determination by more traditional machine vision means**

Keywords—deep learning, computer vision, safety structure, building information model

I. INTRODUCTION

The rate of work accidents in the construction domain is high [1]. Consequently, there is a need for continuous monitoring of construction sites to avoid safety hazards. Safety risk for an individual worker depends on many risk factors including occupation type (e.g. a roofer vs a concrete worker), activity type (e.g. variation in used tools), temporal and spatial factors (e.g. work duration or concurrent activities), and used safety management controls (e.g. safety planning, training, and use of protection systems) [2]. Health and safety inspections at construction sites are used for preventing accidents from occurring. Computer vision can be used for automating part of the health and safety inspections. Computer vision has been used most often to monitor people, personal protective equipment, and machinery [3]. In this work, the focus has been on mitigating the risk of falling. The U.S. Occupational Safety and Health Administration classifies [4] conventional fall protection systems into guardrail systems, safety nets, and personal fall arrest systems. In this work, we decided to support health and safety inspections by ensuring that the required safety elements, mainly safety nets and fences, are in place. Additionally, the goal was to monitor that the construction site remains tidy which influences both fall and fire risks. The technical target of these experiments was to apply deep learning assisted computer vision tasks to adequately determine the existence of the related safety structures or hazardous situations and to create an aggregated service for reporting.

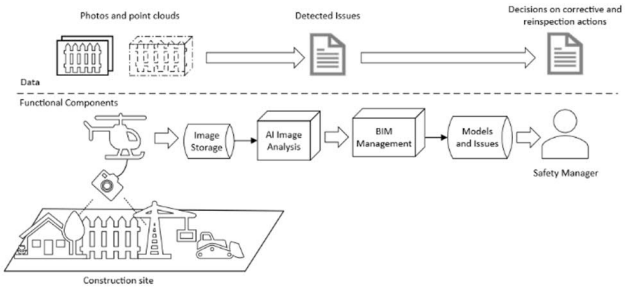

Fig. 1. Intended workflow with dataflow and main functional components in detecting safety issues on construction site. Data capture completed with flying drone precedes the AI image analysis that reports the detected issues to a BIM Management. The safety manager decides on the consecutive actions to mitigate and further repeated monitor and analysis cycles.

New buildings are more and more constructed utilizing Building Information Models (BIM) [5]. Updating BIM model to conform to what is actually built, when changes are being made during construction is thus needed. Technologies for this are already available. The progress at the construction site can be recorded and the potential deviations can be used for updating the as-planned BIM to as-is BIM [27].

Unlike the construction itself, safety structures are not static. In order to achieve an as-planned BIM including safety structures, it would require creating a model, where safety structures would be added based on the time when they are needed. This would require extra work but what is more, in order to be able to automatically compare the as-planned and as-built model, the model of the safety structure should conform to the actual structure used. Furthermore, the installation of the structures should be the same.

The approach presented in this work proposes to overcome these challenges by combining computer vision-based analysis with attaching location of safety structures to as-planned BIM model. The approach allows for variation in type and installment of safety structures as it is based on finding out possible missing of required safety structures. This knowledge is then used for alerting Health and Safety (H&S) manager for requirement of potential actions.

The automated detection of safety factors here is based on computer visual detection of the conditions and situations which may contribute to the formation of a risk. The digital visual material, photographs or video, collected from the work site by automated means may be analyzed to detect the potential risks and related factors with a suitable deep learning-based computer vision solutions. Automated detection workflow, as illustrated in Fig. 1, includes steps of

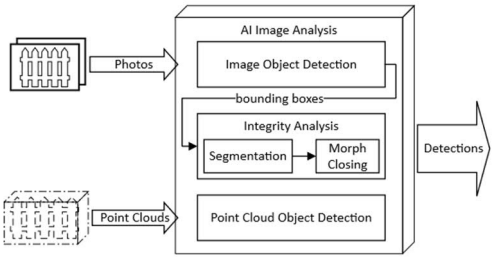

Fig. 2. Functionalities of the AI Image Analysis service component include point cloud and image object detection producing bounding box information for reporting detections and for further safety net integrity analysis purposes. The integrity analysis uses morphological closing after the segmented safety net bounding box.

digital imagery capture, analysis either on board real-time or as post-processing after completed capture mission and finally gathering results to update associated BIM models through reporting. The reports and updated model indicate the need for the required manual corrective actions and validation of their effectiveness further.

The AI Image Analysis service component as in Fig. 1, combines a set of deep learning-based computer vision functionalities as illustrated in Fig. 2. These are object detection for detecting the existence of safety barriers, safety nets and untidy places. Furthermore, similar detection may be concluded for the point clouds for determination of more exact location in three-dimensional space. The safety net integrity check succeeds object detection task analyzing the detected bounding boxes containing safety nets first by segmenting the safety net for the integrity analysis by more traditional computer vision means. The results of the individual detections are then reported respectively to supplement the building models at the BIM management. Determination of the missing safety structures can be done separately on the BIM management level.

Before reaching such capability, a selected deep learning neural networks model needs to be trained as in any other data based machine learning solutions using a suitable training data set. For the deep learning purposes, the training data needs to be gathered, stored, quality controlled and annotated with suitable labels. In general, this requires substantial amounts of training data in the form of digital imagery containing such features as required to be detected. Also, the quality of the captured imagery is important in two senses. Firstly, semantically they should contain what is desired to be detected and secondly technical quality should be reasonable regarding resolutions etc. The resolution usually may have an impact on the final detection precision but for training and detection they are always practically scaled down to a fixed size as dictated by the detection algorithm.

However, the final detection functionalities should be robust and able to detect even in less than perfect conditions some, even manmade, imperfections are beneficial in the training data. Some of the more complete and mature algorithm implementations may utilize artificial augmentation of images with various methods [6]. The raw digital material as such is not enough to reach the adequate levels of detection precision in the final model within reasonable computing time. Thus, the training data needs to be refined manually with an annotation process that should mark those risk objects on the images as identified for the application. Especially in this case annotation classes will include safety barriers and safety nets as risk countermeasures; and piles of garbage as an indicator for untidy location.

Furthermore, to improve the spatial detection and position determination of the required safety structures, object detection or segmentation could be executed on the laser scanned point clouds. The experimentation consisted of pre-processing and labelling of the point cloud data to create a 3D training data set and training one 3D point cloud object detection model type for the inference testing purposes.

Mostly similar boundaries, conditions and processing phases apply also when applying the object detection in 3D point clouds as those of the 2D images. However, the differences and complexities are mostly introduced along the additional spatial dimension on the source data. This is evident in the higher complexity of the deep learning models and data collection and preparation for the learning purposes.

Here, as a training data source we will rely on the 3D point clouds and images captured by the flying drone and rovering robot. The photogrammetrically fusioned LIDAR scans and photographs from the project's sample building to form a full 3D illustration. One sample capture of the complete building being constructed contained alone over 23 million data points. These huge data sets were split to smaller and digestible chunks due to the limit of available effort for labeling and computing capacities just enough for trialing feasibility.

The net defect detection approach in this study is a hybrid combination of deep learning based segmentation to detach the network from the image background complemented with a segmented network integrity analysis with traditional computer vision operation. For the object segmentation deep learning solution, we utilized the same dataset as in object detection but with pixel level annotation. Creation of the pixel level annotation was an even more labor-intensive task and was completed to some extent only to briefly study feasibility of complete net condition detection workflow. In this study we combine this safety net object segmentation with the morphological closing based defect analysis, which was presented also in [7]. This method applies morphological closure operation on the segmented net in order to find kernel size and if it exceeds the threshold which leaves no background regions unless there is a tear in the actual net.

As the current tendency is towards the even deeper and more complex machine learning model architectures, we cannot ignore the performance costs of such solutions. It may be desirable if the created model would maintain high level of precision in detection but also allow detection processing on the handheld devices with a live video streaming material also as many handheld devices possess substantial amounts of graphic displaying but also AI targeted GPU processing power.

To increase the visibility and accessibility of the results of the automated detection of safety factors and risks, it is important to integrate the solution with an existing software tool commonly utilized in the construction domain. To address this goal, an integration between the detection system and the Bimsync platform [8] was established. Bimsync is a collaborative BIM tool that supports the most advanced standards such as IFC, bSDD and BCF. Bimsync enables sharing, visualization, and collaboration on BIM models, issues, documents, and drawings directly in the browser without the need to install additional plugins. The Bimsync integration enables submitting the detection results of safety

factors and risks directly to Bimsync where they can be collaboratively examined by different stakeholders of a construction project.

The main contribution of this paper is to aid automatization of H&S inspections at construction sites with

- detection of safety elements from 2D and 3D imagery
- safety net integrity analysis,
- and integration of automatic analysis with BIM management.

The rest of this paper is organized so, that in the following section we cover related work, in section III we highlight the used image, point cloud and segmentation datasets and the deep learning models for each of the AI Image Analysis service's functionalities. Section IV discusses and analyses the achieved training results for each deep learning functionality. Furthermore, the section V studies the next step integration to the rest of the workflow services and VI concludes this paper.

II. Related Work

In the object detection application domain various models based on the convolutional networks have proved to be suitable and precise enough. Solutions like Faster R-CNN [9], ResNet-101 [10] and various versions of Yolo [11] have evidentially been highly performing even with video feeds while maintaining adequate detection precision.

Some ready-labelled 3D point cloud data sets exist e.g. several for autonomous driving but also for the construction industry. Some benchmarking data sets for detection of the constructed building structures itself exist like IFC datasets [12]. Such typically lack the object classes required here, namely the safety nets and barriers. To create a more robust and reliable detection model, a fully or at least partially automated process would be desirable in data set labelling. Such could be achievable with the aid of the 2D image object detection solution sketched in the earlier section. Another option would be the utilization of detection model architectures like MVX-NET [13] capable of utilizing multi-modal input data consisting of the point clouds and location images. However, their benefits are questioned e.g. in [1], which also sees augmentation procedures as better options.

Two main types, namely image-based and various forms of point-cloud-based detection, methods exist [14]. Some of the 3D point cloud object detection solutions use 2D projected bitmap images from the point cloud views as detection basis. Other solutions utilize the points and their features or other volumetric presentations, e.g. voxels, instead extracting the detectable features from the point clouds. The 2D bitmap projection usually leads to a lower accuracy but consumes less computing resources, which are the opposite case for the feature extracting methods.

These approaches include initial Single Shot Detection (SSD) style anchor box creation and other input data preprocessing techniques intended for a fast inference for the real time processing needs. Typical distinction between the various point cloud object detection approaches is the utilized voxelization method, i.e. a discretization process of producing a reduced size representation of the original point cloud data. This intermediate reduced representation should still capture enough features to allow precise enough object detection on the succeeding layers of the neural network model.

One applicable object detection model architectures is PointPillars [15]. It is a deep learning architecture for object detection in the LIDAR produced point cloud streams. It employs a somewhat similar approach as YOLO for the 2D image based operations. The intermediate representation of the PointPillars architecture is based on representing a group of points as pillar shaped voxels. These voxels or pillars are discretizing the arbitrary points to a fixed set of pillars on the original X-Y plane. The pillars capture and combine an enclosed number points and their related features as vectors which are further reduced in dimensions by the initial pillar feature neural network. The pillar feature network in a sense produces a reduced dimension pseudo image of the pillars or voxels representing the whole point cloud. This intermediate form is presented to the succeeding 2D convolutional network based object detection network.

The fully comprehensive safety net integrity analysis is a challenging task for the data oriented machine learning due to lack of suitable image training datasets. Thus, fully deep learning alternatives were not considered here. However, a solution combining deep learning-based segmentation with a more traditional computer vision analysis provides an adequate solution. A selection of segmentation implementations, Segmentation models [16], provided an adequate and diverse platform for trialing safety net segmentation options for further processing. Supporting a variety of model architectures like Unet, Linknet, FPN and PSPNet with a similarly wide selection of 25 different backbones including VGG19, ResNet and such.

For the actual tear or other damage detection several computer vision techniques have been introduced. Some are for the fishing industry domain like [17] that is based on traditional computer vision algorithms and [18] based on neural networks. On other application fields like metal industry [19] applies deep learning based object segmentation task for surface defect detection. The supervised deep learning methods require vast amounts of data illustrating the defects, which is not available for this study. However, [7] applied simple morphological closing operation and thresholding for number of the vanishing background patches in order to detect tears in the nets after several filtration and segmentation procedures from the video frames.

Comparing planned BIM with as-built BIM can be used for detecting potential problems and ensuring that documentation of the construction remains accurate. Unfortunately, safety measures are not yet usually planned in BIM models [20]. As safety objects are temporary objects as well as vary in shape and size, modeling them accurately may not be possible. Consequently, comparing as-built BIM model with planned model may not be directly usable for detecting missing safety measures. However, the locations of required safety objects are needed in automatic safety inspections to point out what needs to be monitored.

III. Data and deep learning models

The AI Image Analysis component, in Fig. 2, combines three separate computer vision tasks. Each of the deep learning tasks require their own model architectures and respective training datasets which are briefly introduced here.

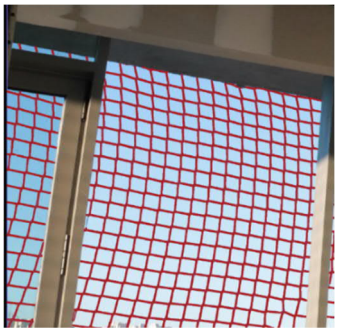

Fig. 3. Part of the sample image with labelled pixels highlighted in red that are part of the safety net. Pixel classification dataset is for training instance segmentation in later phase.

The associated research project partners captured and provided sets of digital images, video clips and point clouds for research purposes here. These captured sceneries of the building construction sites in various locations and stages of completition. The collected and labelled image dataset finally consisted of over 3,500 individual images split to training and validation sets of 2,700 and 800 images, respectively. The available point cloud set consisted of over 28 million points that was split to smaller training and validation blocks.

The current trend in the visual object detection and image classification is towards even deeper neural networks. These outlay a challenge on the available data quantities and quality but also on the available computing resources. To maintain a reasonable computing resource consumption levels and time frame in the training process finetuning pre-existing models and transfer learning procedures was favored. The finetuning and transfer learning utilizes an existing model to diversify or redirect it to detect new classes of objects in the new domain.

A. Image Training Data

The manual labelling was partially aided iteratively with detection results produced early by the detection algorithms trained with a small, already annotated dataset. This incremental and smaller set of images provided enough accuracy to detect at least partially objects of interest in the rest of the images in the complete dataset. The final labelling results were then manually corrected and improved. In 3,500 dataset images, we were able to label over 5,500 individual labels. The labelling tool used for the YOLO formatted annotation and labelling purposes to classify and mark bounding boxes was the freely available labelImg tool [21].

B. Point Cloud Training Data

Typically raw LIDAR point clouds are sparse and without stitching preprocessing to a denser representation. The data set available for our purposes, however, was a result of

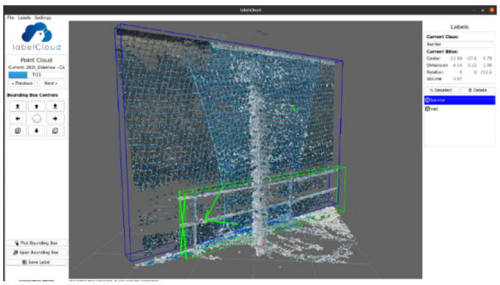

Fig. 4. LabelCloud tool for annotating point clouds with one cropped block in where a safety net and safety barrier are labeled.

photogrammetry combining stitched point clouds and photographs of the locations. The sample starting point is the 23 million point cloud of the building used for the project's case study. The preprocessing of this included cropping to individual floors and then cropping of those to fixed size blocks. Thus, the point cloud was cropped vertically to five individual floors of the building. These cropped floors have visible internal safety nets and arbitrary safety barriers among the structures removing the floors and ceilings from the views. Furthermore, each of the floors were then cropped to a fixed size block of 5 x 5 x 5 units as measured in the CloudCompare tool [22]. See Fig. 5 of one of the floors with one highlighted sample crop and resulting point cloud block for labeling before adding to the training dataset. This slicing produced blocks of c.a. 250,000 points maximum for manual labeling tasks. The point number limitation was estimated with regards to the computer resources and respective time limitations available for this experimentation. Out of a single available large point cloud we were able to source 125 suitable blocks. All empty and those with meaningless number points were left out.

The annotation process to label the objects of interest in such point clouds is a similar process as with the images in 2D case described earlier. The additional dimension, however, will make the amount of data higher and working with the labelling tools more challenging. In this case, the photogrammetric nature and high density of the point clouds aided the recognizing the labelable objects for the human eyes. This is not necessarily the case with the regular sparse raw LIDAR scans. On the other hand, the density of the point cloud data and thus vast data size burdens the processing of them for object detection purposes. The solution used here was to slice and crop data to more processable pieces. The sliced point cloud data set was annotated with the point cloud

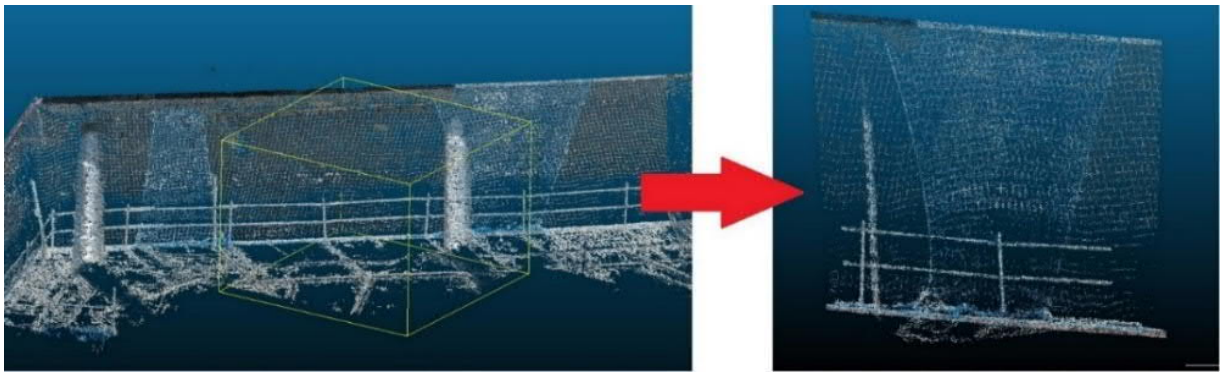

Fig. 5. Floor sliced point cloud of the one of the sample buildings floors of and one 5 x 5 x 5 crops for labeling purposes.

capable labelling tool, labelCloud [23] as visible in Fig. 4 with a sample point cloud crop to be added to training dataset.

C. Instance Segmentation Training Data

A selected images from the object detection set consisted of eight high resolution images. On these a task of pixel level labeling was concluded with QuPath [24] segmentation tool mainly aimed for medical and biological science uses. After a brief cycle of trial and elimination, it was selected as it had some supporting automated tools for the manual labeling task of which color-based thresholding and simple neural network classification were most important. The segmentation label masks among the underlying images were further split into 320 x 320 pixel images to form a training data set. The final training data set included 590 small images and masks split into 392 training, 131 validation and 67 testing samples respectively. See Fig. 3 for a sample split with marked net structure pixels.

D. Image Object Detection Models - YOLO

The object detection task employed one open-source implementation of YOLO version 5 [25] convolutional detection architecture and network. Training of the applicable large YOLOv5 model was made in transfer learning style freezing convolutional layer weights up to layer 10 to reduce the complete training time while reaching adequate levels of accuracy. The ready model is already capable of detecting and locating generic objects in the images present in the COCO dataset, which unfortunately lacks the safety related objects needed here. The transfer learning process will shift the COCO detection capabilities to detect objects relevant in our own construction safety structure and risk dataset. The transfer learning process was completed in 400 epochs with a large pretrained model with 46,642,120 parameters in total with batch size of 10 and AdamW optimizer's initial learning rate of 0.0032. The hyperparameters were based on the YOLO developers' optimization runs with VOC data set. This YOLO implementation supports various data augmentations like color, translation, scaling, shearing, flip and mosaic which were utilized as defined by default parameter settings.

E. Point Cloud Object Detection Models - PointPillars

The selected point cloud object detection model was PointPillars available in Open3D-ML library [26] that supports the use of KITTI formatted point clouds as training data. KITTI format required a separate data feeder implementation in order to accommodate our dataset for training.

During the training process a notion was made how the applied model architecture detected the objects from the birds-eye-view (BEV) vantage point. To accelerate training and improve the detection results in this case, in our data set point cloud chunks were flipped 90 degrees about the horizontal X-axis. This increased substantially upper projection surface area of the detectable objects that tend to be thin but wide. The training used learning rate of 0.00008 with weight decay of 0.01 among other standard implementation parameters as provided with batch size of six for 80 epochs in total.

Some initial trials in this setup showed that AdamW optimizer with weight decay mechanism leads to better results. The availability of stable implementation of AdamW, now, is limited to PyTorch platform only. Open3D-ML developers report about stability problems of the AdamW on the Tensorflow and therefore our tests were run on PyTorch only.

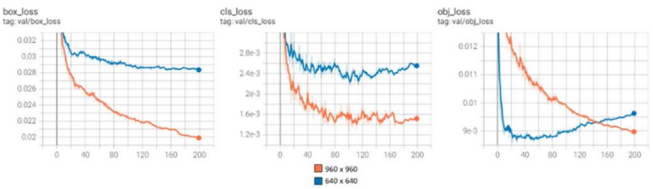

Fig. 6. Development of box, class and object losses during two training runs of 200 epochs on 640 x 640 and 960 x 960 pixel images.

F. Instance Segmentation - Segmentation Models

Segmentation models [16] as safety net segmentation solution with the segmentation training data demonstrated a feasible solution to find a meaningful combination of the segmentation model architecture and its backbone for this trial. However, it was noticed that our small trial-sized segmentation training set was barely enough, and no final conclusions regarding the achievable precision should be drawn. The provided backbones weights were pretrained on 2012 ILSVRC ImageNet dataset and used as such. The training was completed with learning rate of 0.0001, batch size of 10 for 200 epochs with combinations of UNet, PSP, Link and FPN model architectures and VGG, ResNet, SEResNet, ResNeXt, SENet, DenseNet, Inception-ResNet-v2, MobileNetV2, and EfficientNetB7 backbone combinations for comparison.

IV. Results

For comparative reasons the YOLO object detection network training was concluded with the image sizes of 960 x 960 and the default 640 x 640. The training process was continuously monitored and evaluated with the performance metrics as explained in the preceding section. The training continued for 200 epochs, or timesteps during which all the data is consumed once, around which the validation losses started converging. The validation losses for the 960 x 960 and 640 x 640 image sizes are presented in Fig. 6. The losses for the smaller image size starts to diverge after convergence around 100 epochs. Continuing training for the larger image size may improve resulting accuracy slightly but only after a much longer training period.

Evolution of the two threshold mean average precision (mAP) metrics during training illustrated on Fig. 7. The mAP@05 and mAP@05-095 use additional threshold criteria for all detected object classes as Intersection over Union (IoU) ratio of <0.5 and 0.5-0.95, respectively. Apparently, the model reaches higher accuracy of ~0.75 and ~0.525 mAP @0.5 and @0.5-0.95 respectively with higher resolution of 960 x 960 images. Of the specific detection classes, with the safety net the model reaches 0.954 and 0.696 mAP depending on the image resolution, which is deemed adequate at this point of time. The mAP @0.5 results on the higher resolution for other detection classes, barriers and untidy places 0.810 and 0.520 respectively, were somewhat lower due to the high variance in appearance in training data.

The Open3D-ML reports the levels of 52.8 to 61.6 mean average precision, mAP@70, on BEV and 3D views respectively for the commonly available KITTI data sets [27] in detecting cars, pedestrians and cyclists. With KITTI dataset some minor variation is reported between the same PointPillars implementation on Tensorflow and PyTorch

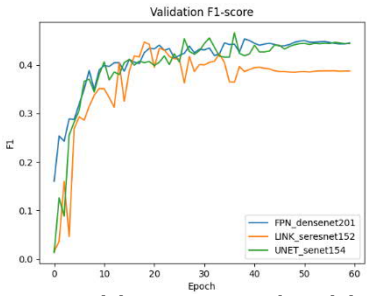

Fig. 8. Segmentation validation F1 score achieved during training with FPN, LINK and UNET model architectures and their best performing backbone solutions. Other weaker combinations omitted for the clarity.

platforms, PyTorch producing slightly lower results. In our limited data set the model was able reach mAP levels 80-90 for safety nets and 15-20 for barriers on BEV with PointPillars on PyTorch. The results were indicated at unrestricted IoU level. From the practical experience of labelling the point clouds and now shown in the result, it is obvious that the barriers were less common items on the point cloud data set than the safety nets and thus detected with lower precisision.

The modest amount of instance segmentation training data still allowed us to reach adequate results for trialing it as part of the integrity analysis concept. The training according to the losses and especially the achieved F1-score in validation converged after 60 training epochs as illustrated in Fig. 8. To keep figure readable only the best results for each model architecture with the corresponding backbone is included. The used batch size was 10 with 320 x 320 pixel image tiles. The best results were achieved in our limited dataset by FPN and UNET model architectures with DenseNet and SENet backbones respectively. The middle tile in Fig. 9 represents instance segmentation result for the first image section on the left in the same figure.

After the segmentation of the safety nets, the net mesh integrity analysis was completed with the simplified methodology as presented in [7] applying only morphological closing until some set conditions were met. Fig. 9 illustrates the sequential results of such with a captured image section and the results of the process starting with the net segmentation and finally the net integrity issue detection highlighting the suspected area after edge detection.

The conditions and thresholds adopted and used here were like the original work. The closing operation was repeated with the increasing kernel size until the number of persisting background areas plateaued below some non-zero level. This suggests the existence of tears or irregularities on the segmented image of the net. That non-zero level was set to 3

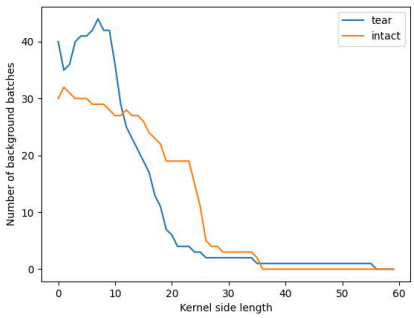

Fig. 11. The number of persisting background batches in two safety net segmented subtiles, one with obvious integrity fault (tear) and one intact (see Fig. 9), after morphological closing with increasing kernel side length.

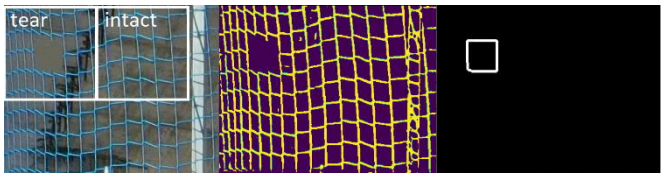

Fig. 9.

and after which according to the original methodology the kernel size is increased by 50%. After that increment and closing operation the persisting background areas, if any, are determined as irregularities i.e., tears on the net mesh structure. The 50% kernel size increment determines the minimum area of irregularity to 150% of the current kernel size. In our work, the rectangular kernel side length was gradually increased from 1 to maximum of 60 if no indication of irregularities earlier. Fig. 10 represents the development of number of persisting background area batches after morphological closing with a certain kernel size for two separate 320 x 320 pixel image tiles. One of such had tear on the net and one was intact as indicated in Fig. 10. The kernel size here refers to a side length of square kernel in pixels. The curve for the image tile with a tear starts to converge below the predefined level of 3 with a kernel side length of 20 and larger and 27 and larger respectively for the intact sample. However, background batches persists even after 50% increment in kernel size up to 30 and even beyond with a torn image sample suggesting integrity issue on the sample. The intact sample preserves zero background batches after closing operation with kernel size 36 and larger.

A. Analysis of Training Results

A common problem shared with each type of data set utilized in this study was the lack of suitable ready full data sets. The unique requirement in our study was inclusion of the safety structures and risks limiting our work to merely a feasibility study than a fully featured production ready system.

It is apparent that the overall precision increases with the increased image size. The large high-definition images with proper lighting make the detectable details naturally clearer and thus easier to detect. However, even when using transfer learning approach, the consumption of the computing resources is high and resulting overall run time is long. The 640 x 640 transfer learning session of 200 epochs lasted for over 24 hours with moderately high-performance GPU acceleration capability offered by Nvidia Quadro RTX 5000 and 16 GB of dedicated RAM. A similar training session with 960 x 960 images lasted for over 86 hours. These times applied only after finding a working and reasonable training setting. Larger image sizes could be used to fully exploit the HD or 4K resolutions for even higher accuracy. But that would come at the cost of the severely extended training time at least with the available computing resources and not considered here.

Interpreting the precision and loss curves suggests that continued training may improve the overall accuracy a little bit further. But it may also prove to be unjustified use of computing resources regarding the levels of achievable improvements. So far, the best available model was the large YOLO model type consuming images of 960 x 960. It was

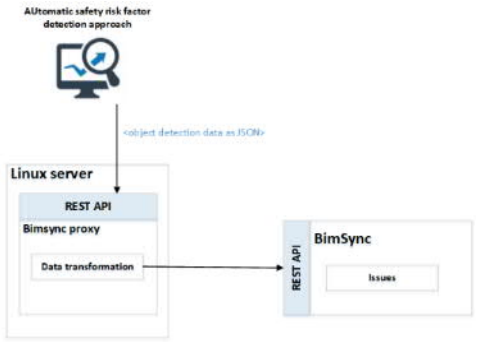

Fig. 12. Integration of the object detection engine and the Bimsync tool.

able to reach reasonable accuracy levels without regressing to overfitting.

Still, the results may vary significantly in practice as most of the available training data was quite homogeneous in image quality, lightning conditions, and certain detectable objects' symmetry in shape and scenery composition. The training dataset's homogeneity has a negative impact on the model's general accuracy in the real-world usage with more varying image qualities and contents, i.e. covariate shift can be expected.

The point clouds and the applicable object detection solutions are sensitive to the vantage point in relation to the observable objects. Here in our data set it was noted and corrected by flipping the training data set point clouds 90 degrees to provide a full facial view from the bird-eye-view. Object detection in point clouds would offer a full spatial detection of the objects of interest. The exact locations and positions would be useful in the result reporting phase and information overlay building. The first and major challenge is the consumption of the resources in capturing and building similarly dense point cloud presentations. The normal sparse LIDAR point clouds may not capture e.g., the thin safety nets on desired recognizable level.

The safety net integrity analysis functionality consisting of instance segmentation and morphological closing operation was performed for the bounding boxes containing a detected safety net. Clearly, the accuracy achieved shows that the segmentation task would greatly benefit from larger amounts and diversity of the training data. As a trial, the current low amounts proved to be enough for show casing the functionality. The preceding morphological closing process was chosen here for the net integrity analysis due to lack of real image data with genuine net faults. Utilizing deep learning segmentation allows omitting several filtering and other complex computer vision tasks while providing good input data for the closing operation to detect integrity problems. A study like [28] approaches this problem with generative networks to generate a defect free sample to find any residual defects. It may not work with the safety nets e.g. due to high variance of the possible safety net droops, folds and see-through layering.

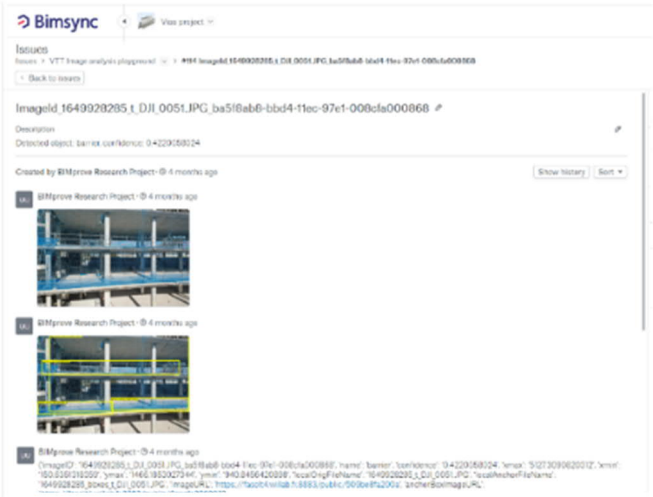

Fig. 13. Results of the object detection module in Bimsync graphical user interface.

V. BIMSYNC INTEGRATION

The established integration between the object detection system and the Bimsync tool aims at improving the visibility, accessibility and dissemination of the detection results. This means that whenever safety risk factors are detected from images, the results are subsequently submitted to the Bimsync platform and reported as new BCF issues. The integration is implemented utilizing the Bimsync REST API v2, which supports, e.g., the exchange of BCF issues between software applications via a RESTful web interface. In addition, a dedicated Python application was developed to transform the output of the object detection module into the format supported by the Bimsync REST API v2.

As mentioned above, the results of the object detection are inserted to Bimsync as new BCF issues. The developed integration approach is represented in Fig. 11. As can be seen, the results of the object detection module are first serialized to JSON strings and then submitted to the BimsyncProxy module. The BimsyncProxy then transforms the data into a format understood and supported by the Bimsync. As a result, a new BCF issue is created in Bimsync that contains relevant data elements extracted from the object detection results.

Fig. 12 shows, how the results of the object detection report at AI Image Analysis service appear in the Bimsync graphical user interface. As can be seen, the original file name of the analyzed image is used as a baseline when generating unique IDs for new BCF issues. In addition, the description section of the issue contains the results of the object detection in a human-readable format. The correlation to the actual physical location presented in the BIM model and the reported detections bases on the geographical location stored in the digital material's metadata, e.g. EXIF, storing also the camera orientation. Determination of detections' location is not covered in this paper though.

VI. CONCLUSION

In this paper, we have demonstrated how to support H&S inspections at construction sites with computer vision based analysis of risk factors including object detection from 2D/3D images and integrating the process with BIM-based construction site management. Currently, it is relatively easy to construct machine vision based deep learning systems and services for various uses technically. Many ready solutions for different data types are available but the challenge is in the data and its annotation as the publicly available data sets may not address the problem pursued. Iterative annotation effort may prove to be suitable by using early stage models to help in detection, or finding labeling tools supporting some basic level detections. The resource scarcity in computing power may be mitigated many times by applying fine-tuning or transfer-learning approaches.

The future work includes application of the photo-based object detection optionally to detect and thus help labelling at least initially the plane projections of the objects in the point clouds. Similarly, the safety net condition analysis with morphological closing may be applied to suitable projections of the 3D safety net bounding boxes.

ACKNOWLEDGMENT

This research was part of BimProve project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 958450. Authors would also like to thank especially Zurich University of Applied Sciences, HRS Real Estate SA and VIAS S.A for supplying necessary data, images and point clouds, for trials.

REFERENCES

[1] M. Zhang, R. Shi and Z. Yang, "A critical review of vision-based occupational health and safety monitoring of construction site workers," Safety Science, vol. 126, p. 104658, 2020.

[2] S. Choe and F. Leite, "Transforming inherent safety risk in the construction Industry: A safety risk generation and control model," Safety Science, vol. 124, p. 104594, 2020.

[3] R. Duan, H. Deng, M. Tian and D. Yichuan, "SODA: A large-scale open site object detection dataset for deep learning in construction," Automation in Construction, vol. 142, p. 104499, 2022.

[4] Occupational Safety and Health Administration, "Fall Protection in Construction," U.S. Department of Labor, 2015.

[5] S. K. Hire, S. Sandbhor and K. Ruikar, "Bibliometric Survey for Adoption of Building Information Modeling (BIM) in Construction Industry– A Safety Perspective," Archives of Computational Methods in Engineering, vol. 29, 05 2022.

[6] Z. Kolar, H. Chen and X. Luo, "Transfer learning and deep convolutional neural networks for safety guardrail detection in 2D images," Automation in Construction, vol. 89, pp. 58-70, 2018.

[7] A. Madshaven, C. Schellewald and J. Zhou, "Hole detection in aquaculture net cages from video footage," in Fourteenth International Conference on Machine Vision (ICMV 2021), 2022.

[8] Catenda, "BimSync," Catenda AS, 2022. [Online]. Available: www.bimsyn.com. [Accessed 1 May 2022].

[9] R. Shaoqing , H. Kaiming, R. B. Girshick and S. Jian , "Faster r-cnn: Towards real-time object detection with region proposal networks," in Advances in Neural Information Processing Systems, 2015.

[10] H. Kaiming, Z. Xiangyu , R. Shaoqing and S. Jian , "Deep Residual Learning for Image Recognition," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[11] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," 2018.

[12] C. Emunds, N. Pauen, V. Richter, J. Frisch and C. van Treeck, "IFCNet: A benchmark dataset for IFC entity classification," in Proceedings of the EG-ICE 2021 Workshop on Intelligent Computing in Engineering, 2021.

[13] V. A. Sindagi, Y. Zhou and O. Tuzel, "Mvx-net: Multimodal voxelnet for 3d object detection," in 2019 International Conference on Robotics and Automation (ICRA), 2019.

[14] D. Fernandes, A. Silva, R. Névoa, C. Simões, D. Gonzalez, M. Guevara, P. Novais, J. Monteiro and P. Melo-Pinto, "Point-cloud based 3D object detection and classification methods for self-driving applications: A survey and taxonomy," Information Fusion, vol. 68, pp. 161-191, 2021.

[15] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019.

[16] P. Iakubovskii, "Segmentation Models," GitHub, 2019. [Online]. Available: https://github.com/qubvel/segmentation_models. [Accessed 1 October 2022].

[17] R. A. H. Jakobsen, "Automatic Inspection of Cage Integrity with Underwater Vehicle," Norwegian University of Science and Technology, Trondheim, 2011.

[18] Q. Tao, K. Huang, C. Qin, B. Guo, R. Lam and F. Zhang, "Omnidirectional Surface Vehicle for Fish Cage Inspection," OCEANS 2018 MTS/IEEE Charleston, pp. 1-6, 2018.

[19] D. Tabernik, S. Sela, J. Skvarc and D. Skocaj, "Segmentation-based deep-learning approach for surface-defect detection," Journal of Intelligent Manufacturing, vol. 31, no. 3, pp. 759-776, 2020.

[20] J. P. Cortés-Pérez, A. Cortés-Pérez and P. Prieto-Muriel, "BIM-integrated management of occupational hazards in building construction and maintenance," Automation in Construction, vol. 113, p. 103115, 2020.

[21] Tzutalin, LabelImg, 2015.

[22] "CloudCompare 2.12.4," GPL software, 14 7 2022. [Online]. Available: http://www.cloudcompare.org/. [Accessed 13 3 2022].

[23] C. Sager, P. Zschech and N. Kuh, "labelCloud: A Lightweight Labeling Tool for Domain-Agnostic 3D Object Detection in Point Clouds," Computer-Aided Design and Applications, vol. 19, no. 6, pp. 1191-1206, March 2022.

[24] P. L. M. F. J. e. a. Bankhead, "QuPath: Open source software for digital pathology image analysis.," 2017.

[25] G. Jocher, "YOLOv5 by Ultralytics," Ultralytics, 5 2020. [Online]. Available: https://github.com/ultralytics/yolov5. [Accessed 1 August 2022].

[26] Q.-Y. Zhou, J. Park and V. Koltun, "Open3D: A modern library for 3D data processing," arXiv preprint arXiv:1801.09847, 2018.

[27] A. Geiger, P. Lenz, C. Stiller and R. Urtasun, "Vision meets Robotics: The KITTI Dataset," International Journal of Robotics Research (IJRR), vol. 32, pp. 1231-1237, 2013.

[28] G. Hu, H. Junfeng, W. Qinghui, L. Jingrong, X. Zhijia and H. Xingbiao, "Unsupervised fabric defect detection based on a deep convolutional generative adversarial network," Textile Research Journal, vol. 90, pp. 247-270, 2020.

[29] S. A. Prieto, B. Garcia de Soto and A. Adan, "A Methodology to Monitor Construction Progress Using Autonomous Robots," in Proceedings of the 37th International Symposium on Automation and Robotics in Construction (ISARC), Kitakyushu, Japan, 2020.