Enhancing Safety on Construction Sites by Detecting Personal Protective Equipment and Localizing Workers Using Computer Vision Techniques

Mohammad Akbarzadeh

A Thesis in

The Department of

Building, Civil and Environmental Engineering

Presented in Partial Fulfillment of the Requirements

For the Degree of

Master of Applied Science (in Civil Engineering) at

Concordia University

Montreal, Quebec, Canada

December 2020

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: Mohammad Akbarzadeh

Entitled: Enhancing Safety on Construction Sites by Detecting Personal Protective Equipment and Near-Miss Events Using Computer Vision Techniques

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science in Civil Engineering

Complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final Examining Committee:

Dr. Mazdak Nik-Bakht	Chair
Dr. Mazdak Nik-Bakht	Examiner
Dr. Chun Wang	Examiner (external)
Dr. Zhenhua Zhu	Co-Supervisor
Dr. Amin Hammad	Co-Supervisor

Approved by Dr. Ashutosh Bagchi

Chair of Department

Dr. Mourad Debbabi

Dean,

Date December 15, 2020

ABSTRACT

Enhancing Safety on Construction Sites by Detecting Personal Protective Equipment and Localizing Workers Using Computer Vision Techniques

Mohammad Akbarzadeh

The construction industry is among the world's most dangerous industries, with a high number of accidents and fatalities. Following safety guidelines and wearing the required Personal Protective Equipment (PPE) is an essential step in mitigating accidents. Safety managers and inspectors are responsible for making sure safety regulations are correctly followed. However, safety inspection is time-consuming, costly, and is done based on a random basis and for a short period.

In order to facilitate safety inspection, various research studies are done using different techniques and technologies. Detecting PPE using Computer Vision (CV) has gained a lot of interest in enhancing construction sites' safety. Nevertheless, detecting PPE on large construction sites and generating safety reports is still a big challenge. Additionally, real-world 2D localization of workers is critical to monitor workers' safety based on their location. This research proposes an automated framework consists of three modules to enhance the safety of construction sites.

The first module of the framework is the PPE Detection (PPED) module, which detects and tracks the workers and their PPE on large construction sites based on the frame segmentation technique. The second module is the PPE Safety Report Generation (PPESRG), which uses PPED results to match workers in two overlapping views and generate technical and practical high-level safety reports while protecting workers' privacy. Finally, the third module of the framework is a Single-camera Localization (SL) module that uses worker detection results from the PPED module and camera calibration parameters to locate workers on 2D real-world coordinate and monitor workers' safety based on their location on the construction site.

The proposed framework is validated using real-world construction videos, and the experimental results of each module demonstrate the practicality and robustness of applying on real-world construction sites. Based on different test videos, the PPED module has achieved 99.04% precision, 91.61% recall, and 90.77% accuracy. Furthermore, the generated safety reports are validated by the safety managers of the project as being practical for safety monitoring on the construction sites. Finally, the proposed CL module is validated with an average error of the average 1.58 *m* for locating workers on the construction sites.

The main contributions of this research are: (1) proposing a nested network based on frame segmentation technique that improved the worker and PPE detection rate on large construction sites, (2) proposing a safety report generation method, which benefits from PPED results of two cameras to generate practical safety reports while protecting workers' privacy and (3) single-camera based technique which is fast and easy to implement on large construction sites in order to locate workers. Future works will focus on accelerating the detection process, improving CV-based localization accuracy, and benefitting from other data sources to enhance generated safety reports (e.g., schedule, etc.).

ACKNOWLEDGEMENT

First and above all, I praise God for all the important things I have in my life: health, family, and good friends. My deepest gratitude goes to my supervisors Dr. Amin Hammad and Dr. Zhenhua Zhu: thanks for accepting me as your student, believing, and trusting in me; your advice and criticism encouraged me to fulfill my masters at my best. I also feel very fortunate to have fantastic research colleagues; Chen: thank you for all helpful advice, and I learned so much from you; Yusheng: thanks for always being such a good friend and all your help during this journey; and all my research office members: Tersoo, Chalrs, Ali, Roya, Roshanak, Majid, Yisha, Ghazale, Neshat, Shide, Negar and Fardin; You guys are fantastic!

I would like to thank Mr. Donald Konan for the kind help, coordination with the construction site, and support through this project.

I would also like to thank the committee members of my thesis defense.

Most importantly, my friends Sana and Ali. I feel very fortunate to be friends with you.

I want to thank Dr. Behzad Kamaledin and his wife Hale, Dr. Vahid Ghourbanian, and his wife Kimiya for your support and help during being away from home. My best friend, Mehran, and his wife, Farnoosh, you are the best; I have learned so much from you; thanks for always being there.

Words can not express my feelings about my family. My parents, Omran and Mahin: You gave everything to me in life, you have trusted me with the journey I wanted to start, and I could never ask for anything more. My one and only sister, Mahdiye: thanks for being there for our parents instead of me. This was impossible without you; may God bless you.

Lastly, I would like to thank Mitacs for financial support through this research study.

Dedicated to my lovely parents, my sister, and those who lost their beloveds in construction accidents.

TABLE OF CONTENTS

List OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Description
2D	Two Dimensional
3D	Three Dimensional
AWCBC	Association of Workers' Compensation Boards of Canada
BBS	Behaviour-based safety
CCOHS	Canadian Centre for Occupational Health and Safety
CNN	Convolutional Neural Networks
COP	Center of Projection
CPU	Central Processing Unit
CT	Centroid Tracking
CV	Computer Vision
DNN	Deep Neural Networks
FoV	Field-of-View
FN	False Negative
FP	False Positive
GPU	Graphical Processing Unit
H	Hardhat
HD	High Definition
HOG	Histogram of Oriented Gradients
HSV	Hue, Saturation, Value
HVSA	High Visibility Safety Apparel

IoU	Intersection over Union
KNN	K-Nearest Neighbors
MSDs	Musculoskeletal Disorders
NH	No-Hardhat
NIOSH	National Instate for Occupational Safety and Health
NSC	National Safety Council
NV	No-safety-Vest
PASCAL	Pattern Analysis, Statistical modeling, and ComputAtional Learning
PBBS	Proactive Behaviour-Based Safety
PCMS	Proactive Construction Management System
PPE	Personal Protective Equipment
PPED	PPE Detection
PPESRG	PPE Safety Report Generation
PTZ	Pan-Tilt-Zoom
RoI	Region of Interest
RANSAC	RANdom Sample Consensus
R-CNN	Regions with CNN features
RFID	Radiofrequency identification
RPA	Reverse Progressive Attention
RPN	Region Proposal Network
RTLS	Real-Time Location System
SL	Single-camera Localization

SSD	Single Shot multibox Detector
TP	True Positive
UWB	Ultra-wideband
V	safety-Vest
VCS	Virtual Construction Simulation System
YOLO	You-Only-Look-Once

Chapter 1 INTRODUCTION

1.1 General Information

Safety regulations are not always followed on construction sites, which is the main reason for accidents. Based on the statistics from the Association of Workers' Compensation Boards of Canada (AWCBC) [1] in 2017, 951 workspace fatalities were recorded in Canada, with an increase of 46 from the previous year [2]. According to SPI Health and Safety [3], more than 450 workers were killed, and over 63,000 workers were injured on construction sites in Canada in 2017. These accidents cost nearly $19.8B each year. Getting hit by falling objects and struck-by accidents are among the most common accidents on the construction sites [4], and the most important way of mitigating accidents is to wear Personal Protective Equipment (PPE). In addition to fatal injuries and casualties, there are other consequences of accidents [5]: time loss of project execution, damaging the reputation of the firm, mental illness of workers, cost of medical care, cost of recruiting and training new workers, compensation cost, cost of repairs and additional supervision, productivity loss, and cost of accident investigation.

The most severe type of struck-by accidents occurs when a worker is struck-by a moving vehicle or piece of equipment [6]. Traffic protection devices and plans are used on construction sites to prevent struck-by accidents by familiarizing workers who may be exposed to traffic hazards with the traffic protection plan. In addition to struck-by accidents, near-miss events are essential to consider on construction sites. According to the National Safety Council (NSC) [7], “near-miss is an unplanned event that does not result in injury or death, but could have’’.

Safety inspectors are also responsible for ensuring that safety regulations are followed by contractors to mitigate accidents [8]. Hardhats and safety-vests are the most basic PPE on the construction sites. "Employees working in areas where there is a possible danger of head injuries from impact, or from falling or flying objects, or from electrical shock and burns, shall be protected by helmets" [9]. Also, Canadian Centre for Occupational Health and Safety (CCOHS) emphasizes the importance of wearing High Visibility Safety Apparel (HVSA) for different lighting conditions and working close to moving vehicles [10].

Hardhats must be worn by construction workers all the time while working. Based on the Bureau of Labor Statistics (BLS) [11], 84% of construction workers that experienced head injuries were not wearing a hardhat. Additionally, BLS reported [11] that 10% of the total 4,340 fatal work injuries were caused by being struck-by equipment. Knowing that safety inspection is done randomly on construction sites and considering the high number of accidents caused by not wearing the required PPE, researchers investigated different tools and techniques for facilitating safety inspection on construction sites.

Existing research studies for detecting PPE on construction sites could be classified into sensor-based and Computer Vision (CV) based techniques. Sensor-based techniques are based on attaching tags to PPE to make sure that safety regulations are correctly followed [12]–[17]. However, detecting PPE on construction sites using sensor-based methods have some limitations. First, attaching tags is costly for large construction projects. Second, electromagnetic noise may affect the accuracy of the locating PPE. Third, the deployment process makes it challenging to apply on large construction sites [16].

On the other hand, CV methods do not have the mentioned limitations of sensor-based methods. However, due to the nature of the construction industry, detecting workers and their PPE by CV techniques from surveillance videos is a challenging task for the following reasons: (1) adverse weather conditions, (2) low lighting conditions, (3) low camera resolution, (4) varying camera height, (5) narrow Field-of-View (FoV) of the camera, and (6) occlusion [18]. Among these challenges, occlusion is the most significant barrier to object detection. Various research studies applied CV techniques for detecting workers and PPE on construction sites to facilitate safety monitoring. Nevertheless, some challenges have remained, which are: (1) detecting workers and their PPE in far-fields, (2) generating safety reports, and (3) real-world localization of workers.

The existing CV based research studies for PPE detection are mostly focused on near-field, single-camera detection, and detection results are not post-processed to generate safety reports. Additionally, workers and equipment localization are based on their location on the image frame, which is no applicable in capturing near-miss events or grouping workers or equipment working together in the real-world. In order to address these gaps, this research presents a novel approach for far-field PPE detection and safety report generation from two camera views. The approach has three modules: (1) the PPE Detection (PPED) module uses a nested DNN framework based on frame segmentation. PPED module detects and tracks workers and their PPE in near, mid, and far-fields. (2) The PPE Safety Report Generation (PPESRG) module, in which PPED results from two cameras, are post-processed to find potential matching workers and generate accurate and practical safety reports. (3) The single-camera Localization (SL) module uses the worker detection results and camera calibration parameters to locate workers on the construction sites.

1.2 Research Objectives and Scope

This research's main objective is to propose an automated framework using CV techniques to enhance safety on construction sites by detecting workers and their PPE to generate safety reports and locate workers in specific zones. Three sub-objectives are defined:

1. Developing a method for detecting workers and their PPE on large construction sites based on frame segmentation.
2. Generating detailed and summary safety reports based on worker matching and PPE detection results of two camera views.
3. Locating workers in specific zones on the construction sites to monitor workers' safety based on their location.

Monitoring workers and generating safety reports about construction workers' safety compliance is the final goal of this research. The study considers detecting workers wearing or not wearing PPE, generating safety reports, and locating workers in different zones on construction sites. In this research, workers and their PPE are detected and tracked under two camera views. Worker detection results are then used to match detected workers under two views, and finally, safety reports are generated. Additionally, specific zones are also defined to monitor workers' safety based on their location on the construction site.

1.3 Thesis Organization

General background and research objectives have been introduced in this chapter, and the remaining chapters are as follows:

Chapter 2: Current practices for safety management and technologies applied to facilitate and enhance construction sites' safety are reviewed. This chapter ends by specifying the research gaps that are addressed in this study.

Chapter 3: This research proposes a framework for enhancing safety on construction sites by generating safety reports of PPE compliance and localizing workers.

Chapter 4: This chapter describes the proposed method's implementation process, which is validated on collected data from real construction sites. The results for each part of the proposed framework are shown in this chapter, and it ends up highlighting the main contributions.

Chapter 5: The research results are discussed, and future research directions are recommended.

Chapter 2 LITERATURE REVIEW

2.1 Introduction

This chapter introduces the current safety management techniques on construction sites, followed by the two main techniques used for enhancing safety on construction sites. The limitations and gaps of the existing methods are summarized at the end of this chapter.

2.2 Workspace Injuries and Safety Management

Safety training is an essential part of safety management. Training and helping workers become familiar with the task and environment where they are supposed to work are considered preventive measures for construction workers [19]. The safety inspection ensures there are no potential or existing safety hazards on construction sites [20]. Different factors are used to specify how frequent inspection must be done, such as the number and size of different work operations, type of equipment and work processes, etc. [20]. Due to these factors, visual inspection is time-consuming, costly, and not very accurate. The five most accidents are shown in Figure 2-1, which are: (1) slips, trips and falls causing two-thirds of the 42,000 falls suffered by workers each year, (2) Falls from heights causing 18% of fatalities, (3) Struck by moving vehicles which in the past ten years caused 13% of the workspace accidents, (4) hit by flying objects or falling objects which in 2016 over 50,000 workers were injured 81 died of this matter and (5) electrocution which happens less often but is the most fatal [4].

Figure 2-1 Five Most Common Workspace Accidents [4]

Wearing appropriate PPE can mitigate most of the mentioned common accidents. As an example, hardhats prevent fatal accidents that are caused by being hit by falling objects. “Employees working in areas where there is a possible danger of head injury from impact, or from falling or flying objects, or from electrical shock and burns, shall be protected by protective helmets.” [21]. Workers might ignore wearing the hardhat for different reasons, such as discomfort and weather temperature, which increase brain injuries. Additionally, struck-by accidents are among the most common accidents on construction sites. One of the reasons for this type of accidents is when the equipment’s operator cannot see the worker. Various tools and techniques are investigated to enhance safety on the construction sites discussed in the following sections.

2.3 Sensor-based Methods for Health and Safety Control on Construction Sites

Behaviour-Based Safety (BBS) is a practical approach for managing safety issues on the construction site, which was extended to Proactive Behaviour-Based Safety (PBBS) by [14]. The proposed PBBS is composed of traditional BBS management and the Proactive Construction Management System (PCMS), which proposes location-based virtual construction through integrating Virtual Construction Simulation System (VCS) with a real-time location system (RTLS). The pilot study results show 36.07% reduced accidents and 44.7% safety index and applied on construction sites. Dong et al. [12] considered a virtual environment of the workspace to track workers' locations and generate warnings. The pressure sensor is placed in the hardhat to recognize that the worker is using it or not; data coming from the pressure sensors and RTLS are used for monitoring the PPEs and generate warnings if they are not worn.

Ultra-wideband (UWB) was another method used in [15] to monitor the non-compliance of safety regulations by placing UWB sensors on the PPE and the equipment. Experimental studies were conducted for identifying the unsafe conditions defined based on proximity, location, and movements of the tag. The overall procedure is to first smooth the raw data from UWB sensors and feed that to a developed motion detector algorithm, which clarifies the status of the tags (stationary or in motion), and finally safety violations based on the relative position and condition of the tag is detected. Siddiqui et al. [16] studied the UWB application to improve construction sites' safety and productivity. UWB application requires a set of cables for the communications between the sensors, which may be challenging to install on construction sites.

Kelm et al. [22] proposed a Radiofrequency identification (RFID) portal installed at the construction site entrance to control the personnel's required PPEs while entering the site. However, the proposed method can not identify if the PPE is worn or not, and also, it is limited to the entrance, and there is no control after passing the portal. RFID systems were to detect and locate the hardhats used by the workers. Recent work by [17] has used the Internet of Things (IoT) to detect the Non-hard-hat use (NHU) on the construction site. The proposed system has a waterproof and rechargeable RFID trigger, a waterproof and rechargeable NHU detector, an active RFID receiver placed in a hardhat, a smartphone app, a web app, and the could server. NHU detector has an infrared beam detector and a thermal infrared sensor. If the sensor is inactive, it means the worker is not wearing the hardhat.

RTLS systems on construction sites are an ongoing topic, and it has been studied extensively. However, there are still some drawbacks and limitations that make it challenging to adapt to construction projects. The RTLS techniques have some limitations for detecting PPE on construction sites. First, attaching the tags is costly for large projects. Second, electromagnetic noise may affect the accuracy of the location-based systems. Third, the deployment process makes it challenging to apply on large construction sites [16].

2.4 Computer-Vision Methods for Health and Safety Control on Construction Sites

Surveillance cameras are installed on construction sites for security reasons and for monitoring the progress of the projects. Surveillance videos recorded from the construction sites attracted the studies about ongoing activities [23], [24], and safety compliance. Several studies investigated CV methods for enhancing safety on construction sites by detecting PPE and capturing near-miss events.

2.4.1. Computer Vision Based Localization

Struck-by accidents were studied by Kim et al. [25] using a CV processing module and safety assessment module. First, entities (i.e., equipment and workers) are detected and tracked using background subtraction and morphological operation. Spatial information of entities is extracted and fed into the safety assessment module. The fuzzy system identifies the safety level based on proximity and crowdedness information, and a safety alarm is generated. Zhang et al. [26] proposed a method for evaluating the collision safety for workers and equipment. The proposed method uses the Faster R-CNN model to detect and track workers and equipment. The centers of detection bounding boxes are used to calculate the relative pixel distance based on the assumption that the construction scene is two-dimensional (2D). The calculated distance and status are analyzed through a fuzzy interface that evaluates the safety level of workers. Yan et al. [27] proposed reconstructing 3D bounding boxes from a 2D vision to recognize the 3D relationship between workers and equipment. In order to generate 3D bounding boxes, geometric of heavy equipment is obtained using the 2D bounding boxes from different sides of the equipment. Finally, depth is estimated using a pinhole camera model, and the crowdedness value is calculated based on the number of objects within a proximity of 6 *m* of an object.

2.4.2. Ergonomic Pose Detection

Dangerous behaviours and unsafe movements of the workers are also among the reasons for accidents on construction sites. The researchers studied different behaviours and activities of workers, which are defined as dangerous behaviour. The framework proposed by Han and Lee [28] aims to have an automated observation of workers. The framework consists of: (1) identification of unsafe behaviour, (2) collecting relevant motion templates, (3) extraction of the 3D skeleton, and (4) detecting the unsafe actions using motion templates and skeleton models.

Behaviour observation is an essential factor in modifying the worker's behaviour more safely. SangUk et al. proposed a collection of motion data using an economical depth sensor to detect unsafe behaviour. Motion data are transferred into a 3D space as a preprocess, classification is performed to identify a typical prior, and the selected prior is used to detect the same action in the test data. Ladder climbing is selected as a case study for proof of concept, and the test results show 90.91% accuracy for detecting the unsafe action.

National Instate for Occupational Safety And Health (NIOSH) defines Musculoskeletal Disorders (MSDs) as the injuries caused by sudden or sustained exposure to repetitive motion, force, vibration, and awkward position [29]. Injuries and illness of the workers' resulting in days away from work, are expensive issues for construction organizations. Computer vision techniques were investigated by Chunxia and SangHyun to identify non-ergonomic postures and movements. The proposed method is to get a 2D skeleton from the image sequence as well as obtaining the 3D coordinates and then reconstruct the 3D skeletons for each frame. The obtained 3D skeleton with the joints' coordinates can be used to recognize non-ergonomic postures. The occlusion effect on the 2D skeleton impacts generating the 3D skeleton, which affects the final classification of the non-ergonomic activity.

2.4.3. PPE Detection

The integration of detection and tracking of construction workers was proposed by [30]. Since the tracking results are not robust enough for practical applications, detection is used to initialize the tracking. The result of the proposed hybrid method is trajectory data that can be used for productivity and safety monitoring. In order to detect a worker wearing a safety vest, a 64 × 128 template from Histogram of Oriented Gradients (HOG), which locates a person in a standing pose, and the template-based tracking method were selected to be the most appropriate for construction applications. The overall framework starts by initializing the tracker in the first frame. In the next frame, the detected worker and a tracked worker are compared in order to be matched.

The proactive warning system proposed by Zhu et al. [31] detects workers and mobile equipment's current positions and predicts their future positions using Kalman filters. The proposed method uses two or more cameras installed on the construction site with different angles to record the activities. The equipment and workers' position are estimated using the triangulation principle and fed to Kalman filters to predict future positions.

The detection of the entities (e.g., workers, equipment, material, etc.) on a large construction site was studied [32]. The proposed method is an automated way of tracking entities where the detection of entities initiates tracking. The foreground is first recognized by subtracting the background from the frames, then people are detected based on HOG features, which use a predefined standing human template. In order to classify the worker from ordinary people, a worker is defined as a person wearing a safety vest. In the next step, HSV (Hue, Saturation, Value) colour histogram and K-Nearest Neighbors (KNN) are used to classify the workers and non-workers.

The proposed method by Park et al.[33] detects workers and hardhats by two different detectors using HOG features. Then, matching was used to make sure that each detected worker is wearing a hardhat. The detection technique is also used by Shrestha et al. [34] to detect hardhats and workers. The proposed method has two main parts of face detection and hardhat detection. The face detection program detects the workers' presence on images; then, the edge detection algorithm is used to find the hardhat features at the upper part of the face. Mneymneh et al. [35] proposed a motion detection method instead of background subtraction to get the worker's region. The two main components of hardhat detection are a cascade object detector applied to the upper region of the human and a colour-based classifier to discard wrong detections.

Fang et al. [36] proposed a deep-learning method using Faster R-CNN for detecting the workers not wearing hardhats. The proposed method considered various factors such as weather conditions, light conditions, and visual range and postures for detecting workers not wearing a hardhat. Fang et al. [37] proposed that the proposed computer vision-based method uses two Convolutional Neural Networks (CNN) first to detect a worker's presence and then detect the harness. Detected workers in the first stage are cropped and re-entered to the second neural network to detect the harness. Xie et al. [38] proposed a CNN for real-time detection of workers and hardhats. After detecting the worker and the potential hardhat, to ensure that the hardhat is appropriately worn, Intersection over Union (IoU) is calculated based on the worker and hardhat bounding boxes. Wu et al. [39] applied a deep-learning approach for detecting the hardhats worn by the workers. They proposed a one-stage CNN based on the Single Shot multibox Detector (SSD). In order to better detect the small-scale hardhats, Reverse Progressive Attention (RPA) was integrated into the SSD framework.

Nath et al. [40] proposed a real-time approach to detect PPE in order to improve the safety on the construction site. Three Deep Neural Networks (DNN) models based on You-Only-Look-Once (YOLO) were proposed to identify if workers adequately wear hardhats and safety vests. The second proposed approach has achieved the best accuracy and real-time speed. In this approach, the YOLO model detects workers and directly classifies compliance or non-compliance with safety regulations. The method proposed by [41] focused on localizing a person's head and classifying it to wearing or not wearing a hardhat class. The top-down approach is utilized to enhance the extracted features from relatively small objects (i.e., hardhat), and finally, a residual-block-based prediction is applied on a multi-scale feature map to detect wearing hardhats. Table 2-4-3-1 summarizes the CV based method for enhancing safety on the construction sites by detecting PPE.

Table 2-1. Overview of the previous CV techniques for PPE detection on construction sites

Reference	Year	Method	Real-time	Safety report	Multiple cameras	Far-filed	Detecting classes						Precision (%)[a]	Recall (%)[a]	Limitation
							Person	Hardhat	No-hardhat	Vest	No -Vest	Safety harness			
Park and Brilaksi [32]	2012	• HOG features • HSV colour space					✓			✓			-	-	Limited feature template
Park et al. [33]	2015	• HOG features • Background subtraction					✓	✓					94.3	89.4	Worker's feature might not match the used HOG template
Shrestha et al. [34]	2015	• Face detection • Edge detection					✓	✓					-	-	Face detection is not applicable in the far-field
Mneymneh et al. [35]	2017	• Motion detection • HOG features • Colour classifier				✓	✓	✓					84.97[b]	84.36[b]	Colour based classification under illumination effect
Fang et al. [36]	2018	•Faster R-CNN				✓			✓				93.7[b]	92.3[b]	Single PPE class detection
Fang et al. [37]	2018	• Faster R-CNN • Deep CNN										✓	79.2[c]	93.1[c]	Limited activities and the effect of harness's colour
Xie et al. [38]	2018	• CNN based hardhat detection	✓				✓	✓					54.6		Carried but not worn hardhats generate wrong results
Wu et al. [39]	2019	• SSD with RPA	✓					✓	✓				83.89		Limited to near-field workers
Nath et al. [40]	2020	• YOLOv3 • CNN classifier	✓				✓	✓		✓			72.3		Limited to near-field workers
Wang et al. [41]	2020	• Mobile net • top-down module	✓					✓	✓				88.4		Limited to near-field workers
Present study	2020	• Frame segmentation • Faster R-CNN • Safety report generation		✓	✓	✓	✓	✓	✓	✓	✓		**99.04[c]**	**91.61[c]**	

a) All methods are validated with different datasets

b) Far-field results

c) Combined final results of the nested network

2.5 Gaps in Body of Knowledge

The current literature could be divided into sensor-based and CV-based techniques to monitor safety on construction sites. However, sensor-based techniques [12], [14]–[16] have some limitations: first, attaching tags is costly for large construction projects. Second, electromagnetic noise may affect the accuracy of the locating PPE. Third, the deployment process makes it challenging to apply on large construction sites [16]. On the other hand, CV-based techniques do not have these limitations. The existing CV-based methods can be divided into two main categories: (1) feature-based [30]–[35] and (2) deep learning-based methods [36]–[39], [40, p.], [41].

The feature-based and motion-based CV methods are limited to detecting moving objects, predefined feature templates, and colour histograms. Therefore, detection is limited to moving objects, specific features and dependent on illumination conditions. On the other hand, deep learning-based CV methods do not have these limitations. However, there are three main challenges remained which are: (1) detecting workers and their PPE in far-field views, (2) benefiting from multiple cameras to cover large construction sites to generate safety reports, and (3) 2D real-world localization of construction workers. This research will address the challenges of existing deep learning-based CV methods for PPE detection and 2D real-world localization. The objectives of this study are: First, developing a nested network based on frame segmentation for detection and tracking workers and their PPE in near, mid, and far-field views. Second, generating detailed and summary safety reports based on matching workers in overlapping camera views. Third, monitoring workers' safety based on their location on the construction site. The details of the proposed framework are described in the following sections.

Chapter 3 RESEARCH FRAMEWORK

3.1 Introduction

This chapter introduces the main framework for enhancing construction sites' safety by detecting workers and their PPE, generating technical and high-level safety reports for PPE detection, and monitoring workers' safety based on their location. The framework consists of three main parts: (1) The PPE Detection (PPED) module which detects and tracks workers with PPE in near, mid, and far-filed views, (2) PPE Safety Report Generation (PPESRG) which matches PPE detection results under two camera views and generates PPE safety reports, and (3) Single-camera Localization (SL) module which uses the worker detection results to locate workers on the construction sites. The overall of the proposed framework is shown in Figure 3-1. The details of the proposed framework are explained in the following sections.

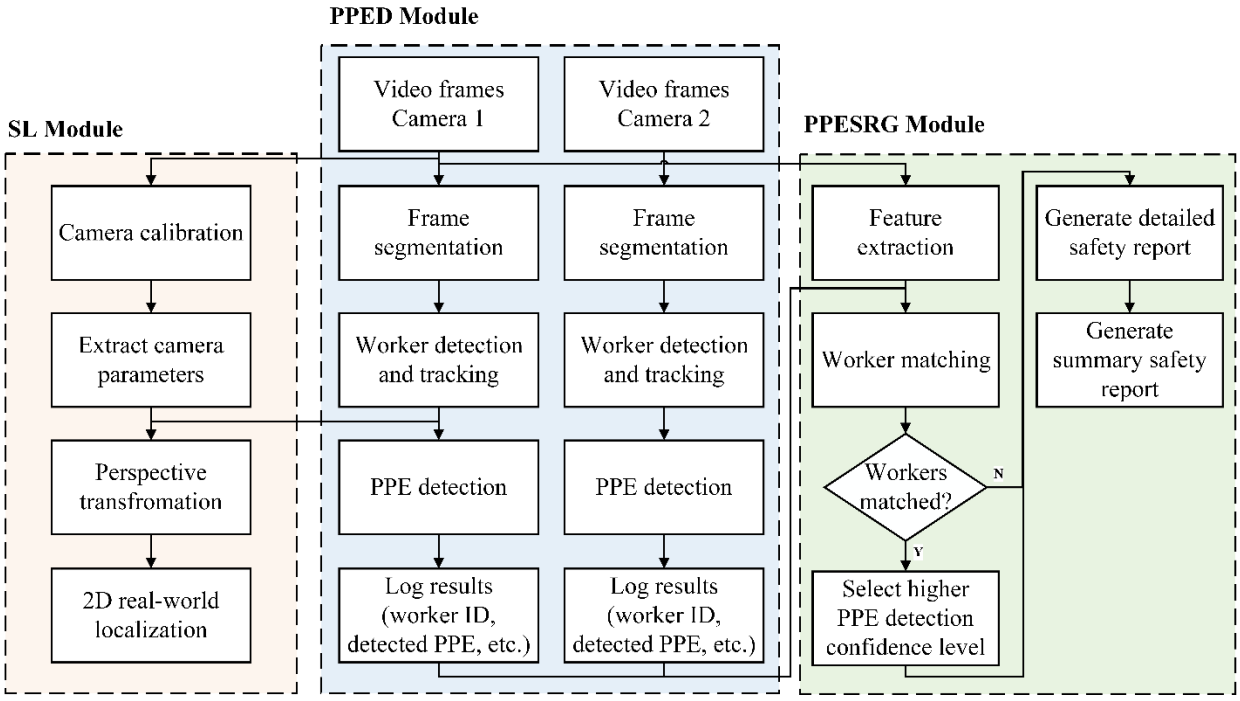


Figure 3-1 The overall proposed framework

3.2 PPE Detection (PPED) Module

Surveillance cameras are usually installed on nearby buildings or poles at height to have a broad view of the constriction site and avoid potential occlusions. Installing multiple surveillance cameras helps in solving the far-field detection challenge to some extent when far-field workers to one camera could be captured in other camera's near-field, as shown in Figure 3-2(a). Camera installation at height results in having a perspective view [42] of the construction site, making worker and PPE detection more challenging. Far-filed workers are captured in the upper part of the image frame, as shown in Figure 3-2(b), and as will be explained in the implementation section, the far-field workers' size is almost 1/3 of the near-field workers. Despite having multiple cameras in some cases, workers might only be captured in both cameras' far-field view, which remains a challenge for worker and PPE detection models.

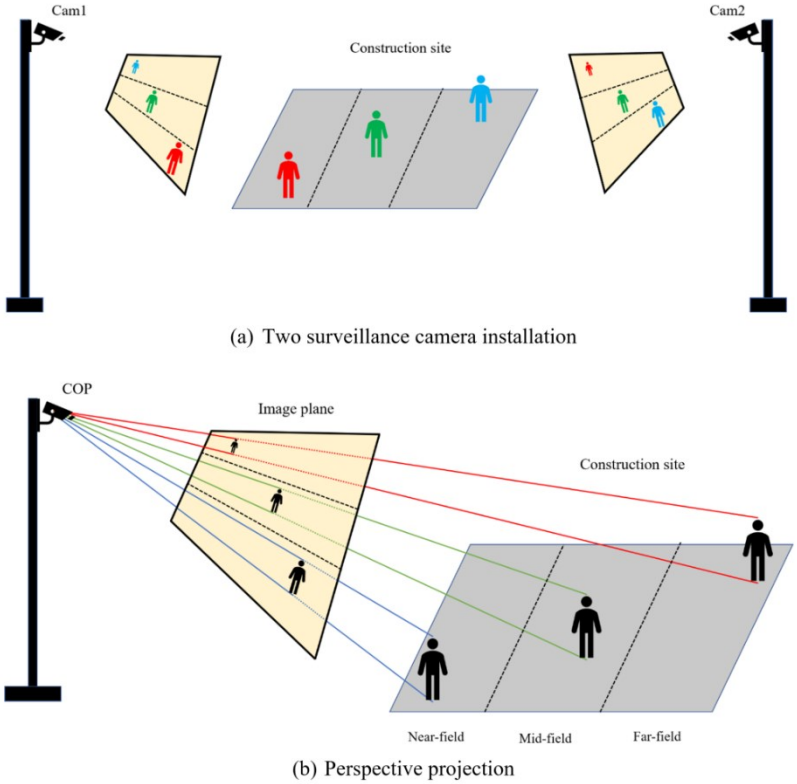


Figure 3-2 Two surveillance camera installation and perspective projection

The PPED module aims to detect and track workers and PPE classes in near, mid, and far-field views. This module is an extended version of [43], a novel frame segmentation technique with a nested deep learning-based network to overcome the challenges of detecting workers and PPE classes on large construction sites. The proposed nested network consists of two Faster R-CNN [44] models to detect workers and PPE classes. The Faster R-CNN model is selected as it has high accuracy and sufficient processing time compared to the other exiting detection algorithms (e.g., SSD, etc.), which have lower detection accuracy. Faster R-CNN models are custom trained based on the transfer learning approach, as will be explained in more detail in the implementation section. The proposed PPED module's overall flow is shown in Figure 3-3, which has two main parts of the worker and PPE detection models.

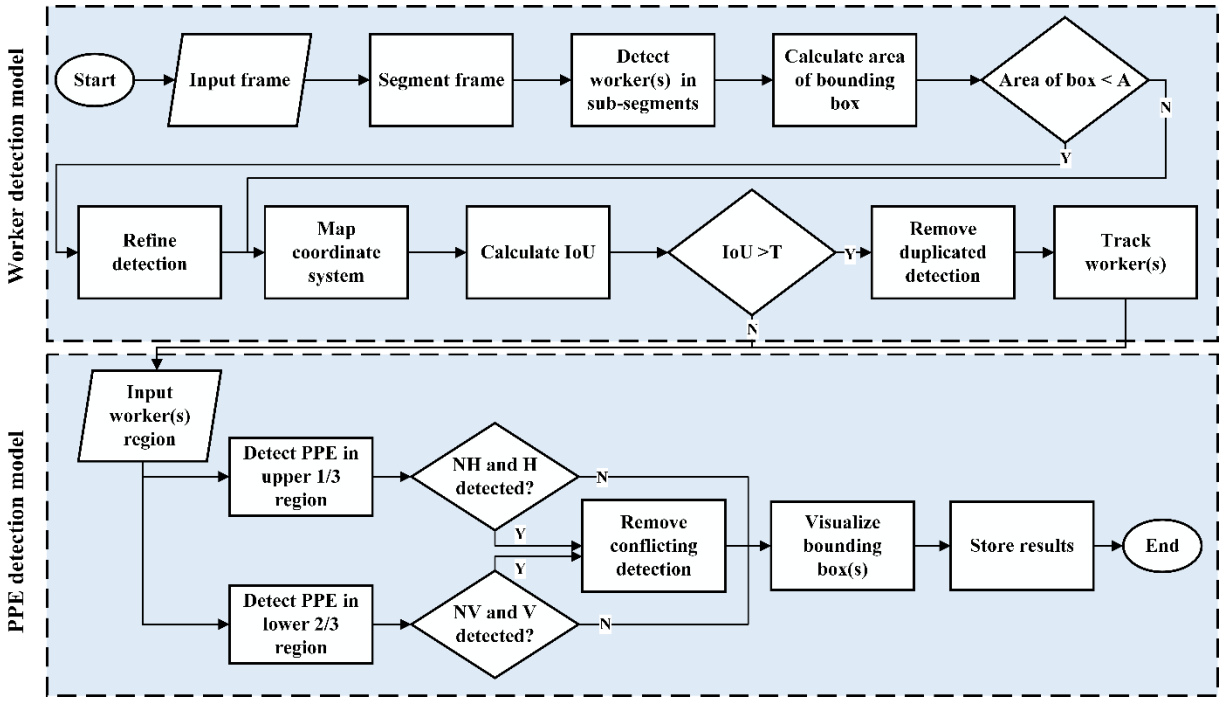


Figure 3-3 The overall flow of the proposed nested network based on frame segmentation

3.2.1. Worker Detection Model

As explained, far-field workers are captured smaller than near and mid-field workers because of the perspective projection, making the worker and PPE detection more challenging. Therefore, the worker detection model aims to detect far-field workers by eliminating the Faster R-CNN model's resizing effect and ensuring that segmentation covers all workers. The worker detection model has four main steps: (a) frame segmentation, (b) worker detection, (c) refine detections, (d) remove duplicated detections, and (e) tracking

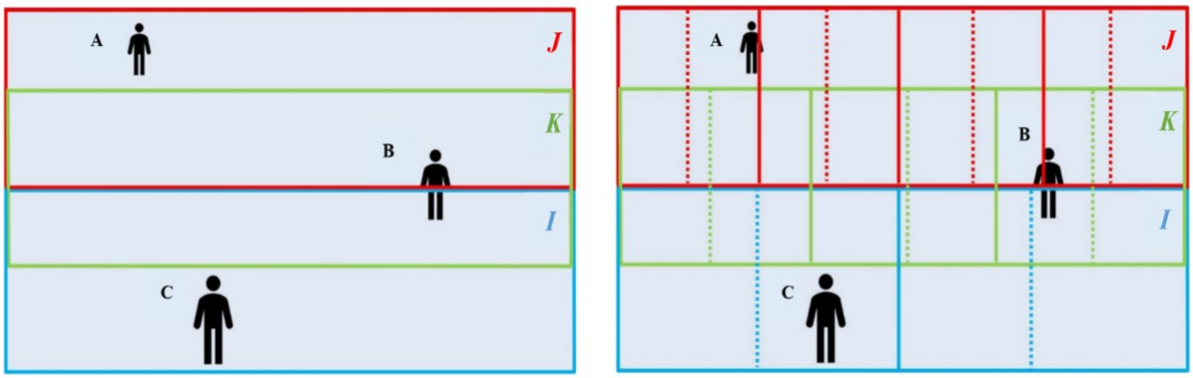
(a) Frame Segmentation

High Definition (HD) surveillance cameras are commonly used on construction sites with a frame size of 1920×1080 pixels. However, the Faster R-CNN model has an input frame size of 1024×600 pixels with an aspect ratio of 1:7. Therefore, the model resizes all the frames greater than the input size in both training and testing stages. Ren et al. [44] considered two factors for resizing images, as shown in Equation 3-1 and Equation 3-2. Both factors are applied to frames that do not fit with the input dimension, and results are compared with the input dimensions. Priority is with the resizing result of large scale factor and will be selected if it fits the input dimensions, otherwise resizing the result of small scale factor will be selected. As an example, a frame with 1920 ×1080 dimensions will have large and small scale factors of 0.56 and 0.53, respectively. Therefore, using the large scale factor, the resized frame will be 1075 × 605 pixels, which does not fit the network input dimensions. On the other hand, the resized frame with a small scale factor will be selected with a size of 1017 × 572 pixels, which is within the input range of the network.

$$Small\ scale\ factor = \frac{Max\ input\ dimension}{Max\ image\ dimension} \quad (3\text{-}1)$$

$$Large\ scale\ factor = \frac{Min\ input\ dimension}{Min\ image\ dimension} \quad (3\text{-}2)$$

Due to the perspective view, workers' size on the image plane changes parallel to the frame's vertical axis, and the average size remains the same on the horizontal axis. Therefore, for simplicity, three main and equal segments are defined parallel to the horizontal axis. Figure 3-4(a) shows an example of main segments where, *I* is the near-field strip where workers are captured in the largest size compared to the whole frame, *K* is where workers are captured in medium-size (mid-field) and *J*, where workers are captured at the smallest size on the image plane (far-field). In some cases, workers could intersect with the defined borders of the main segments, which results in cutting them into two parts. For example, $Worker_B$ is intersecting with the borders of segments *J* and *I,* as shown in Figure 3-4(a). To ensure workers are covered, the *K* segment is defined in a way that it has 50% overlapping with *I* and *J* segments and can fully capture $Worker_B$.



(a) Main segments (b) Adding sub-segments

Figure 3-4 Schematic presentation of main and sub-segments

Additionally, sub-segments are needed within each main segment to meet the network's input size and aspect ratio. Workers' width on the image plane is the smaller dimension of a worker and is used to calculate the sub-segments' width. Equations 3-3 and 3-4 are used in order to get the number of the sub-segments where *N* is defined as a number of workers fitting into a sub-segment. Different N values and aspect ratios are considered to get the best detection. Similarly, $Worker_A$ is intersecting with the cropping lines of sub-segments, as shown in Figure 3-4(b). Overlapping sub-segments are defined with a 50% overlap to ensure that workers in sub-segment are fully covered.

$$Number\ of\ sub\text{-}segments\ = Ceiling\ (\frac{Frame\ width}{N \times Average\ width\ of\ worker}) \tag{3-3}$$

$$Sub\text{-}segment\ width = \frac{Frame\ width}{Number\ of\ sub\text{-}segments} \tag{3-4}$$

(b) Worker Detection

The results of frame segmentation (sub-segments and overlapping sub-segments) are the inputs to the Faster R-CNN worker detection model. The Faster R-CNN model applies a CNN to extract the features and generate a feature map, then a Region Proposal Network (RPN) uses anchor boxes on top of the feature map to find and assign a score for each potential object. Finally, the RPN and Region of Interest (RoI) pooling layer results are merged into a fully connected layer to identify if the detected is a worker or not.

Transfer learning is often used for custom training of DNN models in order to achieve optimal results in a short time [45]. In order to apply the transfer learning technique, a pre-trained model on a related dataset is needed to fine-tune the parameters of the network. Large scale datasets are publicly available that could be used for training or transfer learning of DNN models [46], [47]. One crucial factor for custom training and transfer learning of Faster R-CNN models is adjusting the size, scale, and aspect ratios of anchor boxes. As explained in the Faster R-CNN model's architecture, anchor boxes are used for preliminary prediction of objects in feature maps. The default configuration of Faster R-CNN has nine anchor boxes with sizes of 128×128, 256×256, and 512×512 and aspect ratios of 1:1, 1:2, and 2:1 [44].

The Faster R-CNN model's anchor boxes are modified by changing the scales and aspect ratios based on the custom training dataset. This research proposes using k-means clustering [48] to modify the aspect ratios and scales. Based on the recommendation of [49], worker bounding boxes in the training dataset are clustered based on the IoU of bounding boxes. Considering two bounding boxes of $b_1 = (w_1, h_1)$ and $b_2 = (w_2, h_2)$, where w_1 and w_2 are the widths of the bounding boxes, and h_1 and h_2 are the height of the bounding boxes, Jaccard index can be calculated as shown in Equation 3-5. The Jaccard index return values between 0 and 1, which 1 means that the two bounding boxes' size is equal and 0 for not being equal. Implementing k-means is done by initializing k random boxes for primary means and then assigning each bounding box to a specific cluster.

$$Jaccard\ index_{(b_1,b_2)} = \frac{min(w_1, w_2) \times min(h_1, h_2)}{((w_1 \times h_1) + (w_2 \times h_2)) - min(w_1, w_2) \times min(h_1, h_2)} \qquad (3\text{-}5)$$

(c) Refining Detections

In some cases, small objects similar to workers in far-field views (e.g., traffic cones) might be detected as workers, known as False Positive (FP). This study considers an average area of bounding boxes of workers within each main segment and removes detected bounding boxes smaller than the defined area threshold. Equation 3-6 is used within each main segment to remove FP from detection bounding boxes where, A is the defined average area, W and H are the width and height of the detected bounding boxes, respectively.

$$FP = W \times H < A \tag{3-6}$$

(d) Removing Duplicated Detections

Having a worker detected in a sub-segment and overlapping sub-segment results in generating redundant bounding boxes. The proposed solution to avoid double counting the workers is first to map the bounding box coordinates from the local coordinate system of sub-segments to the whole frame coordinate system. After mapping the coordinates, IoU is calculated between every detection bounding box. A threshold (T) is defined to remove the bounding boxes that overlap above T. Equation 3-7 is used to remove redundant detection bounding boxes and keep one bounding box per detected worker.

$$Dupplicated\ bounding\ boxes = \frac{Area\ of\ overlap_{(box_i, box_j)}}{Area\ of\ union_{(box_i, box_j)}} > T \tag{3-7}$$

(e) Tracking

The final detection bounding boxes coordinates are used as input for the multi-object Centroid Tracking (CT) algorithm to track workers. The CT algorithm is selected because of three main reasons: (1) speed, (2) dealing with low-resolution features, which makes it difficult to use more advanced tracking techniques such as DeepSORT [50], and (3) having a reliable worker detection model, which makes the tracking task easier. The CT algorithm assigns an ID to each detected worker and updates the IDs by the next detection. Tracked bounding boxes and IDs at time *t=0* are compared with newly detected bounding boxes at *t=1*. Euclidean distance is calculated between the centers of bounding boxes in *t=0* and *t=1*. Finally, based on the calculated Euclidean distance, IDs are updated, removed, or new IDs are registered.

3.2.2. PPE Detection Model

The second Faster R-CNN model in the proposed nested network is the PPE detection model that aims to detect PPE classes within the bounding boxes of detected workers in near, mid, and far-field views. Detecting PPE classes in the far-field view is more challenging than worker detection due to their small sizes on the video frame. Additionally, having a perspective view makes them appear even smaller on the frame. The PPE detection model is expected to have better PPE results in nearer fields, but the defined segments are continuous with a 50% overlap, and some workers can be very close to the border between the main segments with different postures and lighting conditions, which may affect the results.

A Faster R-CNN model is custom trained for detecting four PPE classes of H (hardhat), V (safety-vest), NH (no-hardhat), and NV (no-safety-vest); when NH or NV detected, it will be considered safety noncompliance behaviour. Moreover, to ensure that PPE is appropriately worn, potential regions are defined for each class, which H and NH must be detected in the upper 1/3 region of the detected worker's bounding box and V and NV in the lower 2/3 region. Besides, this research considers the possible conflicting detections caused by the low resolution of far-field workers. The PPE detection model might return both H and NH or V and NV at the same time, which is considered as conflicting detection results. The proposed solution for conflicting detections is to compare the detected classes within each potential zone based on the confidence level of detection and eliminate the one with a lower confidence level. The final step of PPED is to save worker and PPE detection results with the assigned tracking IDs, which will be used in the PPESRG module.

3.3 PPE Safety Report Generation (PPESRG) Module

The purpose of this module is to benefit from the PPED results of two surveillance cameras that have overlap in their FoV in order to generate accurate and reliable PPE safety reports considering practical and privacy aspects. The overall flow of the proposed PPESRG method is shown in Figure 3-5. The proposed module has two main parts of worker matching and safety report generation. The worker matching method by [51] is adopted in this study to match the detected workers under two camera views. The results from the PPED module and matching are combined to generate detailed safety reports. The PPED results for matching workers from each camera view are compared, and PPE classes with a higher confidence level of detection are selected. Additionally, tacked ID from two camera view is merged into a new unique ID. On the other hand, for the nonmatching workers, PPED results and tracking IDs are directly logged into detailed safety reports.

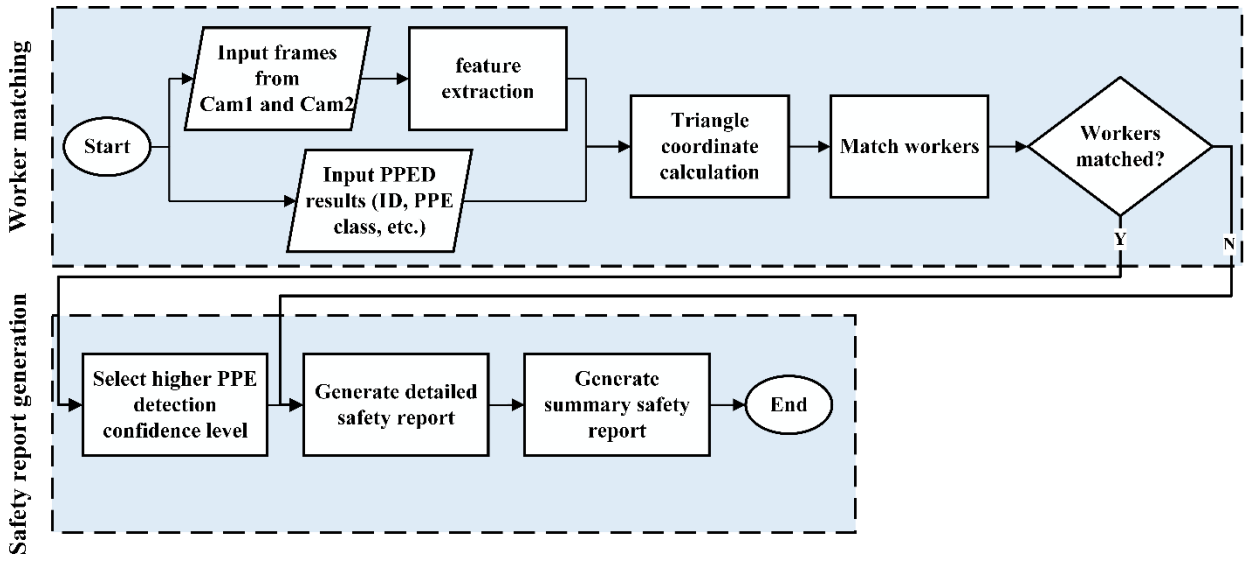


Figure 3-5 The overall flow of PPESRG module

The detailed safety reports contain worker IDs, confidence levels of detections, and PPE classes, which are not convenient for safety managers to get a prompt understanding of the safety status on a specific day on the construction site. Additionally, worker IDs are listed in the detailed safety reports, which leads to privacy concerns. Based on the privacy regulations, employers have to notify the workers officially that surveillance cameras are installed to monitor the construction site, and worker's faces must not be visible in the videos [52]. In order to have more practical safety reports while protecting the privacy of construction workers, summary safety reports are generated. The generated summary safety reports are practical for safety managers to get the safety status on a specific day instantly while protecting worker's privacy. However, in some cases, authorized employees could access the detailed safety reports to extract more details about a specific noncompliance incident.

3.4 Single-camera Localization (SL) Module

The proposed Single camera Localization (SL) module aims to monitor workers' safety based on their location on the construction sites. Based on safety regulations or specific conditions on the construction sites, some areas are identified as high-risk areas for accidents (i.e., close to mobile equipment, etc.). Visual monitoring of workers' presence in specific areas is time-consuming and effortful. Therefore, the SL module aims to locate workers on 2D real-world coordinate systems, which can be used to find the distance between workers and mobile equipment or identify if a worker is in specific zones using a single surveillance camera. The overall of the SL module is shown in Figure 3-6.

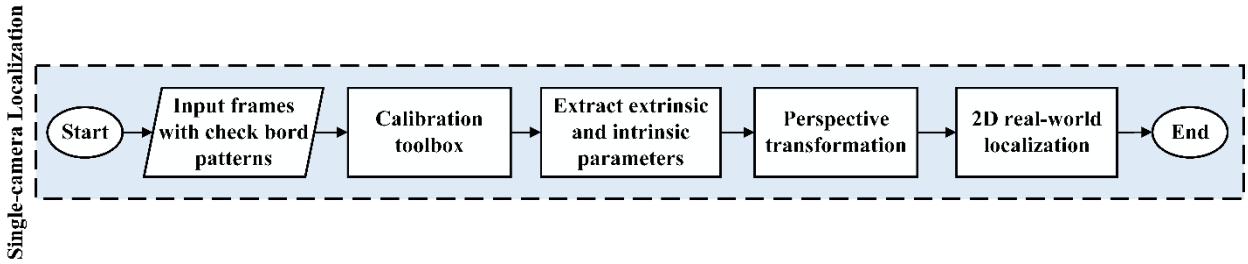
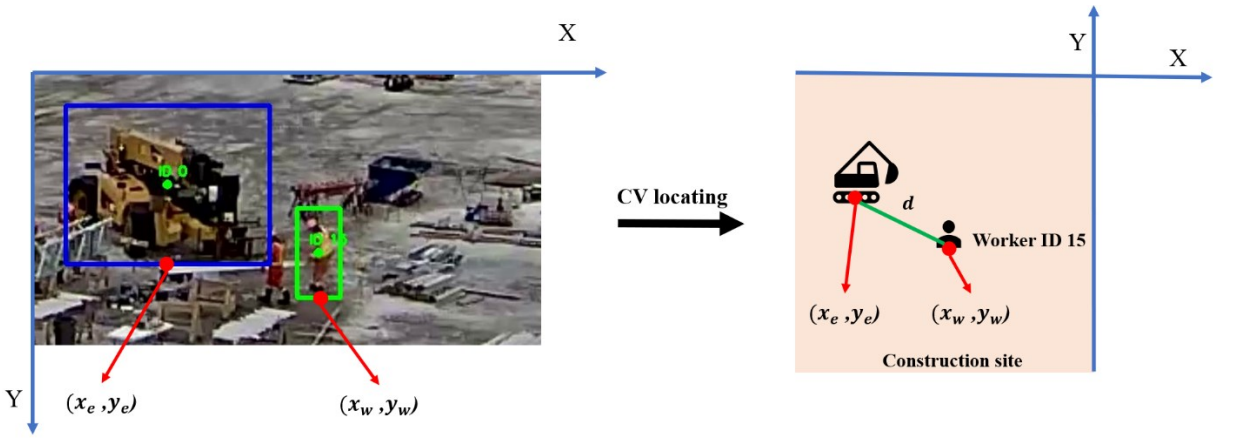


Figure 3-6 The overall flow of the proposed SL module

Camera calibration is necessary for the quantitative analysis of video frames. In order to calibrate the camera, the proposed method by [53] is used, which uses a checkboard pattern in order to extract the features from images. The frames with the pattern are fed into the Matlab camera calibration Toolbox by Bouguet [54] to get the camera's intrinsic and extrinsic parameters. Having the calibration parameters and workers' location on the image frame, perspective transformation [55] in Equation 3-8 is used to find the world metric coordinate of the workers, where s is the scale factor, u and v are the pixel coordinates, M is the camera's intrinsic parameters, R is the rotation matrix of the camera, and t is the translation vector. It is assumed that all the objects (i.e., workers or equipment) are at the same height ($Z_{constant}$), which is considered the ground with Z=0.

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = M(R\begin{bmatrix} X \\ Y \\ Z_{const} \end{bmatrix} + t) \tag{3-8}$$

The bottom center of the detection bounding boxes is used for locating workers or equipment from the image coordinate system to real-world 2D coordinate, as shown in Figure 3-7(a). The location of detected workers and equipment in the world coordinate system after perspective transformation is shown in Figure 3-7(b). Finally, having the world coordinates, the distance between worker and equipment is calculated using Equation 3-9, which identifies the safety status of a worker based on the distance to equipment.



(a) Worker and equipment location on image frame (b) 2D real-world location of worker and equipment

Figure 3-7 CV based locating concept

$$distance = \sqrt{(x_w - x_e)^2 + (y_w - y_e)^2} \tag{3-9}$$

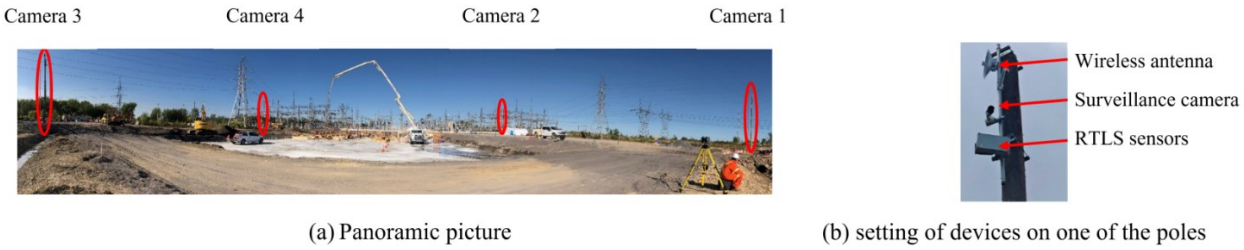
3.5 Summary

This section proposed an automated framework for enhancing safety on the construction sites. There are three main modules of the proposed framework are (1) PPED module, which proposed a novel frame segmentation technique based on the size of workers captured on the image frame. Defined segments are inputs to the worker detection model, and then detected workers are used as inputs for the PPE detection model. (2) PPESRG module utilizes two cameras' PPE detection results and matched workers under two overlapping camera views. Based on worker matching results, PPE detections with the highest detection confidence are used to generate more reliable safety reports. (3) SL module aims to locate workers on the real-world 2D coordinate system of the construction sites. The perspective transformation method uses the camera calibration parameters and worker detection results to locate workers on a real-world coordinate system. The location of detected workers can be used to capture the near-miss events, group workers, or equipment working together and monitor the commuting into a specific zone on construction sites.

Chapter 4 IMPLEMENTATION AND CASE STUDIES

4.1 Implementation Environment and Data Collection

Three main parts of the proposed framework are implemented and tested in order to validate the feasibility and effectiveness. The framework is implemented on the Python platform with the support of Tensorflow Object Detection API [56] and OpenCV library [57] that are providing the required algorithms and tools for image processing. Validation is done on the Compute Canada cluster [58] with six CPUs (Central Processing Unit) and one GPU (Graphical Processing Unit) NVIDIA P100 Pascal and 12 GB memory. Data is collected from a construction site located in Montreal, Canada, where Axis P1425-E surveillance cameras with HD resolution (1920×1080 pixels) were installed on four poles at about 10 *m* height, as shown in Figure 4-1 (a). Also, on each pole, there are three devices: wireless communication antenna, surveillance camera, and RTLS sensors that are used in another research, as shown in Figure 4-1(b). Wireless antennas are used for transferring the data to the office, where a receiver antenna is installed to connect with the server. Videos are recorded for four months based on the working schedule from Monday to Thursday from 7 AM to 5 PM under a variety of weather and lighting conditions.



(a) Panoramic picture (b) setting of devices on one of the poles

Figure 4-1 Panoramic picture showing surveillance camera setup

4.2 PPE Detection (PPED)

As explained in Section 3.2, detecting workers and PPE in far-field is a challenging task, and therefore, the PPED method is proposed based on the frame segmentation technique in order to detect workers and PPE on large construction sites. The proposed method is validated under different conditions, and the results are summarized in the following sections.

4.2.1. Training Dataset and Annotation

The training for worker detection and PPE detection was created using two primary datasets containing 2200 and 1000 images, respectively. In the worker detection dataset, the main object of interest is human. PPE detection dataset has four objects to detect, which are (1) hardhat, (2) safety-vest, (3) no-hardhat, and (4) no-safety-vest. The worker detection dataset contains images extracted from surveillance videos of the same construction site. The gathered images are cropped in small segments with a size of 960×540, which has the same aspect ratio of the Faster R-CNN network to make sure there is no resizing happening in the training phase.

The publicly available datasets contain images taken from the street level and close to the workers, making them unsuitable for the construction application. Also, the proposed camera setup defined in this research can be reused on other construction sites, and therefore, the dataset can be reused. Moreover, for long-lasting construction projects adding images from the construction site to a training dataset is supposed to improve the detection result, which is considered a positive point.

Since the region of interest for the PPE detection model is the human body, the PPE dataset contains persons' cropped images. The PPE dataset is created by combining the CUHK01 dataset [59] that contains people captured from a high angle of view, as negative examples of workers with no PPE, and the image dataset of workers with PPE from the site. The images in both datasets are annotated using open-source software [60] using PASCAL (pattern analysis, statistical modelling, and computational learning) format [61]. Examples of worker and PPE annotations are shown in Figure 4-2(a) and 4-2(b), respectively.

(a) Workers' annotations (b) PPE annotations

Figure 4-2 Examples of workers and PPE annotations

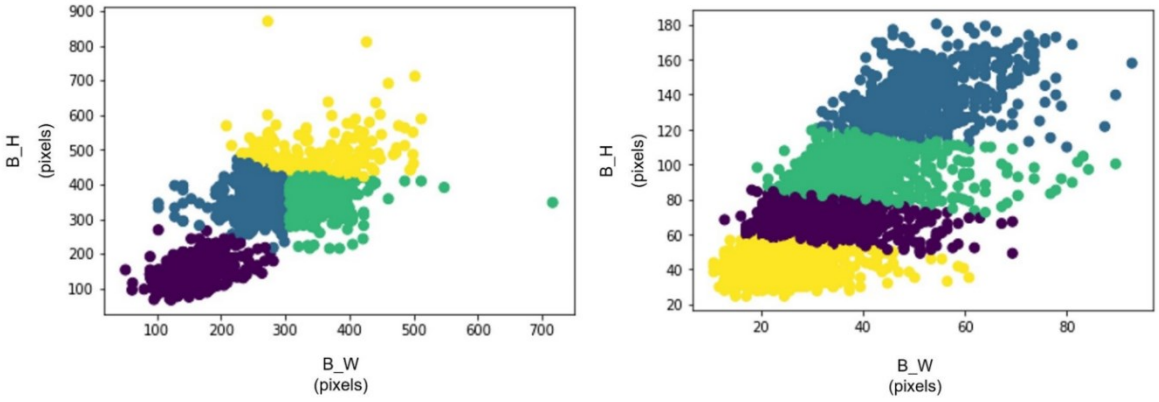
4.2.2. Hyperparameters Adjustment

As explained in Section 3.2.1, the K-means algorithm is used for clustering the training datasets for worker and PPE detection. In order to cluster the dataset, only the normalized width and height of the boxes are needed. The input data for the K-means algorithm is normalized by *B-W*, *B-H*, which are calculated using Equations 4-1, and 4-2 [49], where *w* and *h* are the width and height of the box, respectively, and (x_{min}, y_{min}) and (x_{max}, y_{max}) are the top left and lower right corners of annotation bounding boxes. Faster R-CNN network, by default, considers four scales and three aspect ratios.

The Faster R-CNN Inception Resnet v2 Atrous version model is used for worker and PPE detection, which, by default, considers four scales and four aspect ratios for anchor boxes. As a result of worker dataset clustering, the aspect ratios and scales for the worker detection model are adjusted to (0.40, 0.38, 0.43, 0.48) and (0.15, 0.16, 0.22, 0.32), respectively. Additionally, the PPE detection dataset contains cropped worker bounding boxes where PPE classes are relatively small. The default Faster R-CNN anchors are defined for detecting general objects such as cars, bicycles, cats, etc. Based on the Faster R-CNN model's initial training results on the PPE detection dataset with default anchors, the training could not learn PPE classes' features, and the training loss continued to increase. The clustering PPE detection dataset with four clusters resulted in four aspect ratios and scales that are (0.76, 0.85, 0.92, 1.1) and (0.51, 0.62, 1.2, 1.4), respectively. Figures 4-3(a)and (b) illustrate the clusters for worker and PPE detection datasets, respectively.

$$B - W = \frac{X_{max} \times X_{min}}{w} \tag{4-1}$$

$$B - H = \frac{Y_{max} \times Y_{min}}{h} \tag{4-2}$$



(b) Bounding box clustering for workers dataset (a) Bounding box clustering for PPE dataset

Figure 4-3 Bounding box clustering for worker and PPE detection datasets

4.2.3. Case Study

The three main segments of *I*, *K*, and *J*, are defined with an equal size of 1920×540 parallel the horizontal axis of the image frame. However, the main segments' size does not fit with the Faster R-CNN network's input size. As explained in Section 3.2.1, in order to find the optimum number of sub-segments considering the accuracy and detection time, using Equation 3-3, three values of 5, 15 and 25 are considered for *N*. The average widths of workers is measured on the image plane. The average width is 55 pixels in segment *I*, 35 pixels in segment *K*, and 20 pixels in the main segment *J*. The total number, size, and aspect ratio of sub-segments with different *N* are summarized in Table 4-1.

Table 4-1 Details of sub-segments based on different N values

N values	Specifications	Main segment *I*	Main segment *K*	Main segment *J*
5	No. sub-segments with overlaps	13	21	39
	W x H (pixels)	274×540	174×540	96×540
	Aspect ratio	0.51	0.32	0.17
15	No. sub-segments with overlaps	5	7	13
	W x H (pixels)	640×540	480×540	274×540
	Aspect ratio	1.19	0.89	0.51
25	No. sub-segments with overlaps	3	5	7
	W x H (pixels)	960×540	640×540	480×540
	Aspect ratio	1.78	1.19	0.89

(a) Worker Detection Module

Based on the surveillance camera installation and construction work, Camera 3 is selected as the closest camera for detecting workers and PPE, as shown in Figure 4-4(c). Four 5-minute validation videos recorded with 30 frames per second are selected from different phases of the project, and detection is performed every second. Precision, recall, and accuracy are calculated to evaluate detection results. The results are based on assuming the value of 50% for the IoU. Tables 4-2 to 4-5 show the three test videos' results for the first, second, and third evaluation videos. Additionally, custom trained worker detection Faster R-CNN model is tested on the validation videos without the frame-segmentation technique (i.e., N=1) to compare the results.

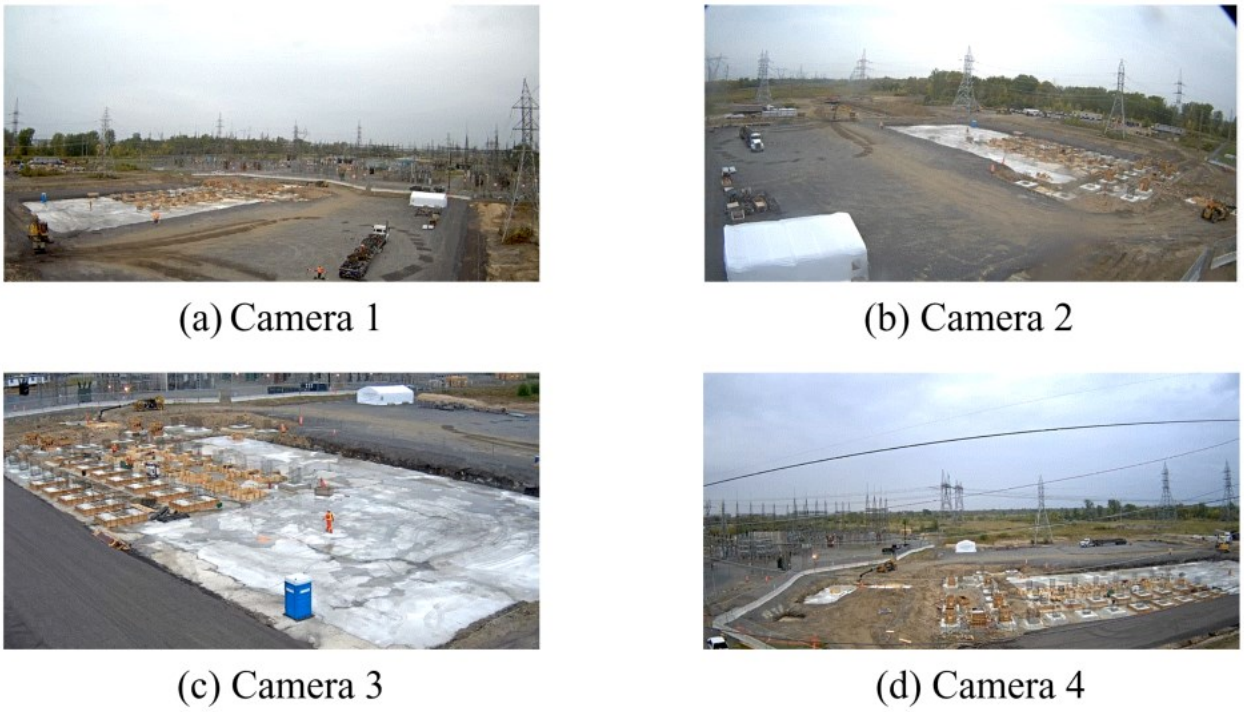


Figure 4-4 Multiple camera views of the construction site

Table 4-2 Sensitivity analysis of the different N with 50 percent IoU first evaluation video

N values	Full-frame precision (%)	Full-frame recall (%)	Full-frame accuracy (%)	Accuracy segment I (%)	Accuracy segment K (%)	Accuracy segment J (%)	Time (sec)
1	90.25	21.77	17.88	N/A	N/A	N/A	193
5	99.26	92.33	91.70	92.96	94.23	91.41	9,364
15	99.53	90.00	89.62	94.56	89.99	87.31	4,029
25	100	91.59	91.59	97.36	92.26	86.45	2,433

Table 4-3 Sensitivity analysis of the different N with 50 percent IoU second evaluation video

N values	Full-frame precision (%)	Full-frame recall (%)	Full-frame accuracy (%)	Accuracy segment I (%)	Accuracy segment K (%)	Accuracy segment J (%)	Time (sec)
1	88.85	23.40	22.73	N/A	N/A	N/A	194
5	95.67	95.51	91.56	98.33	96.09	92.38	8,546
15	83.79	97.75	82.21	98.33	91.08	84.45	3,448
25	96.15	92.83	89.36	100	93.32	90.60	1,932

Table 4-4 Sensitivity analysis of the different N with 50 percent IoU third evaluation video

N values	Full-frame precision (%)	Full-frame recall (%)	Full-frame accuracy (%)	Accuracy segment I (%)	Accuracy segment K (%)	Accuracy segment J (%)	Time (sec)
1	97.97	51.77	49.90	N/A	N/A	N/A	187
5	98.26	96.84	95.20	81.34	99.19	97.24	8,897
15	96.80	97.76	94.70	99.11	98.85	93.73	3,572
25	100	99.01	99.01	100	99.88	98.95	1,916

Table 4-5 Sensitivity analysis of the different N with 50 percent IoU fourth evaluation video

N values	Full-frame precision (%)	Full-frame recall (%)	Full-frame accuracy (%)	Accuracy segment I (%)	Accuracy segment K (%)	Accuracy segment J (%)	Time (sec)
1	97.49	44.44	43.94	N/A	N/A	N/A	193
5	98.79	93.35	92.29	93.93	92.74	91.16	8532
15	93.72	95.48	89.74	99.16	98.26	93.71	3446
25	100	97.09	97.09	99.44	98.25	92.94	1928

Based on the evaluation results of the worker detection model, N=25 has achieved the best detection accuracy, where the image frame is divided into a total of 15 sub-segments that fit the input dimensions and have an aspect ratio close to the Faster R-CNN model's input aspect ratio. The worker detection model achieved an average precision of 99.04%, an average recall of 95.13%, and average accuracy of 94.26% based on the full-frame results. Furthermore, each section's accuracy is calculated in Tables 4-2 to 4-5. The average accuracies of the far-field segment (J), mid-field segment (K), and near-field segment (I) are 99.20%, 95.93%, 92.24%, respectively, for N=25. Based on each segment's average accuracy, near-filed has achieved higher accuracy than mid and far-field, since more workers' features can be captured in the closer fields, making the worker detection easier.

(b) PPE Detection Module

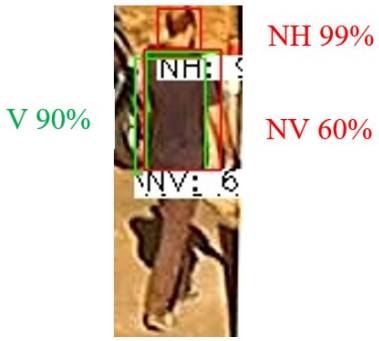
In order to evaluate the PPE detection model, 200 images are selected as a test dataset in which half of them are workers wearing PPE and the other half not wearing PPE. The PPE detection model is evaluated on the full-datasets and for near, mid, and far-field. The PPE detection results for each class are summarized in Table 4-6 based on precision, recall, and accuracy. Based on the PPE detection results, the average precision among the four classes has achieved 100%, which indicates the effectiveness of the proposed solution for removing conflicting detections.

Table 4-6 Evaluation metrics for PPE detection results

Classes	Full-dataset precision (%)	Full-dataset recall (%)	Full-dataset accuracy (%)	Near-field accuracy (%)	mid-field accuracy (%)	Far-field accuracy (%)
H	100	99.98	99.98	100	97.62	97.22
NH	100	93.20	93.20	93.93	90.63	94.74
V	100	99.01	99.01	100	100	97.37
NV	100	93.00	93.00	87.88	93.33	97.30
Average	100	96.30	96.30	95.45	95.40	96.66

The PPE detection model has achieved better results in some further segments compared with nearer segments (e.g., NV in mid-field). However, as explained in the methodology, defined regions are continuous, and different poses and lighting conditions affect the detection results. An example of bad light conditions is shown in Figure 4-5(a), in which the worker is in the near segment. The bad light condition results in a lower confidence level for the actual PPE class (i.e., NV=60%), and it is considered a conflicting detection, as shown in Figure 4-5(b).

(a) Bad light condition for worker region



(b) Conflicting PPE detection results

Figure 4-5 Example of conflicting PPE detection due to bad light conditions

Overall, the most critical evaluation criteria for the constriction sites' safety monitoring is the recall. Equations 4-3 to 4-5 are used to calculate the precision, recall, and accuracy. As shown in Equation 4-4, False Negative (FN) is in the denominator. FN represents the cases where the object detection model failed to detect the exiting ground truth class, and an example is a worker with no hardhat annotated in the test dataset, which is not detected by the object detection model. Additionally, True Positive (TP) shows the detections that are correctly matching the ground truth bounding boxes. The worker detection model and PPE detection model results are combined to illustrate the overall precision, recall, and accuracy of the proposed PPED module. Therefore, the PPED module has an overall 99.04% precision, 91.61% recall, and 90.77% accuracy in detecting workers in near, mid, and far-filed with four different classes of PPE.

$$Precision = \frac{TP}{TP + FP} \tag{4-3}$$

$$Recall = \frac{TP}{TP + FN} \tag{4-4}$$

$$Accuracy = \frac{TP}{TP + FP + FN} \tag{4-5}$$

The worker detection model is custom trained using the video frames with a perspective view of the construction site in the training phase. The trained worker detection model is reusable on similar construction sites, with cameras installed at the height and workers wearing the full-body orange safety vest. On the other hand, the PPE detection model is custom trained using bounding boxes of workers captured from a high angle of view. The trained PPE detection model can be used for similar PPE detection scenarios to detect if workers are wearing the full-body safety vest and hardhat or not. Examples of the worker and PPE training dataset are shown in Appendix E.

4.3 PPE Safety Report Generation (PPESRG) Module

Safety reports are used by the safety or project managers to understand construction sites' safety status and behaviour. As explained in Section 3.3, in order to generate accurate and practical safety reports, the PPESRG method is proposed. The proposed method is validated on the surveillance videos from the construction site, and the steps are explained in the following sections.

4.3.1. Data Collection

Among the available four surveillance cameras discussed in Section 4.1, Camera 1 and Camera 2 are selected to validate the proposed method. Figure 4-6 shows examples of video frames from the selected cameras. Having overlaps in views is an essential factor in achieving accurate worker matching results. The robustness of the matching method under different environmental conditions is validated by Zhang et al. [51].

Camera 1 Camera 2

Figure 4-6 Example frames for matching workers in two views

4.3.2. Matching Visual Features

In order to find the appropriate feature extraction threshold, the matching accuracy of the construction workers is tested with different threshold values. The frames of the selected Camera 1 and Camera 2 are used to investigate the proper threshold value of the feature matching. The best worker matching accuracy was achieved at a threshold equal to 0.8. The generated triangle meshes for Camera 1 under the 0.8 threshold are shown in Figure 4-7. The centroid of detected workers' bounding boxes is used to locate them on the generated triangle meshes; then potential workers are matched using the epipolar information and workers' location on triangle meshes.

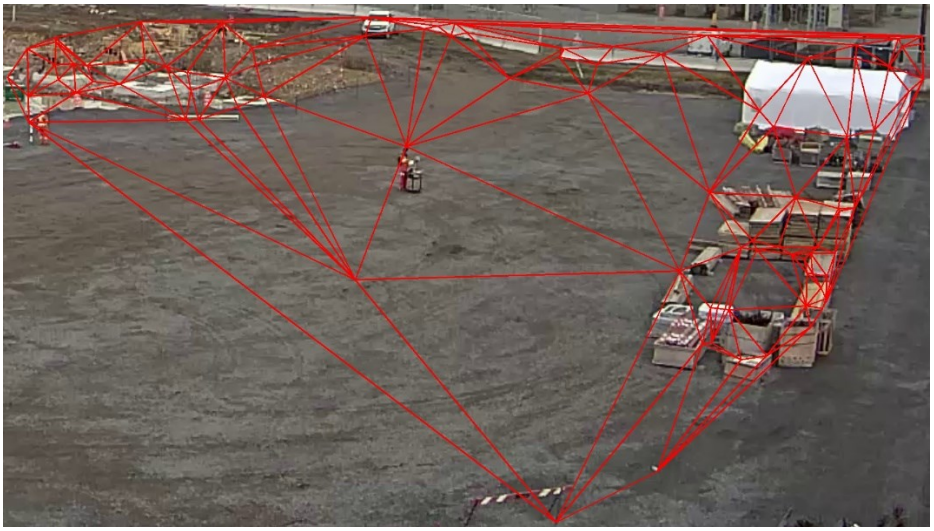
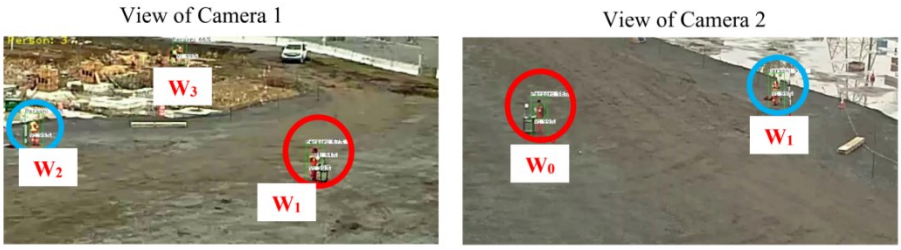


Figure 4-7 Triangle meshes generated for Camera 1 at 0.8 threshold

4.3.3. Case Study

As discussed in Section 3.3, the detailed and summary safety reports are generated for matching and nonmatching workers based on the worker matching and PPE detection results. The PPE detection results are compared for each matching worker, and the detected PPE classes with higher confidence are logged into the detailed safety report. On the other hand, for nonmatching workers, the available PPE detection results from one camera are logged into the detailed safety reports.

The detailed safety reports are further processed to generate summary safety reports, giving an overview of the safety status while protecting workers' privacy. A short video is used to validate the proposed method, where two workers are detected by the PPED module from Camera 1 and Camera 2. The detected workers' location on the image frame is then used as input to the worker matching technique to find the matching workers in two camera views, shown in the same colored circle in Figure 4-8(a).



(a) Matching workers from Camera 1 and Camera 2

(b) PPE detection results and the matched worker from Camera 1 and Camera 2

Figure 4-8 An example of PPE detection results combined with worker matching

Finally, the PPE detection results for nonmatching, and matching workers are logged into the detailed safety report. W_1 from Camera 1 and W_0 from Camera 2 are matched correctly. Additionally, the worker is located near Camera 1, as shown in Figure 4-8(a), which resulted in correctly detecting the safety noncompliance by the PPE detection model, as shown in Figure 4-9(b). The detailed safety reports are generated in Excel sheets containing the PPED information from a single camera or two cameras. However, in order to protect the privacy and have practical safety reports for safety managers, detailed safety reports are processed into summary safety reports. Examples of detailed and summary safety reports are shown in Figures 4-9(a) and 4-9(b), respectively. The matching colours in Figure 4-9(a) for Camera 1 and Camera 2 indicate the matched workers. The summary safety report indicates the number of PPE noncompliances for not wearing hardhats or safety vests. The safety managers validated the generated safety reports as being practical for safety monitoring on the construction sites.

Camera 1

Frame NO	Date	Time	Worker ID	Hardhat	No-hardhat	Vest	No-vest
0	20/9/2019	10:10 AM	0	98%	0	99%	0
0	20/9/2019	10:10 AM	1	85%	0	99%	0
30	20/9/2019	10:10 AM	0	0	0	99%	0
30	20/9/2019	10:10 AM	1	99%	0	99%	0
60	20/9/2019	10:10 AM	0	0	0	99%	0
60	20/9/2019	10:10 AM	1	0	94%	99%	0
60	20/9/2019	10:10 AM	2	0	0	99%	0

Camera 2

Frame NO	Date	Time	Worker ID	Hardhat	No-hardhat	Vest	No-vest
0	20/9/2019	10:10 AM	0	99%	0	0	0
0	20/9/2019	10:10 AM	1	0	0	99%	0
0	20/9/2019	10:10 AM	2	99%	0	99	0
0	20/9/2019	10:10 AM	3	99%	0	0	0
30	20/9/2019	10:10 AM	0	0	0	99%	0
30	20/9/2019	10:10 AM	1	0	0	99%	0
30	20/9/2019	10:10 AM	2	99%	0	97%	0
60	20/9/2019	10:10 AM	0	0	0	99%	0
60	20/9/2019	10:10 AM	1	0	0	99%	0

Matching camera 1 and camera 2

Frame NO	Date	Time	Worker ID	Hardhat	No-hardhat	Vest	No-vest
0	20/9/2019	10:10 AM	0	98%	0	99%	0
0	20/9/2019	10:10 AM	1	99%	0	99%	0
0	20/9/2019	10:10 AM	2	99%	0	99%	0
0	20/9/2019	10:10 AM	3	99%	0	0	0
30	20/9/2019	10:10 AM	0	0	0	99%	0
30	20/9/2019	10:10 AM	1	99%	0	99%	0
30	20/9/2019	10:10 AM	2	99%	0	0	0
60	20/9/2019	10:10 AM	0	0	0	99%	0
60	20/9/2019	10:10 AM	1	0	94%	99%	0
60	20/9/2019	10:10 AM	2	0	0	99%	0

(a) Example of detailed safety report

Date	No. of workers	No. of No-Hardhat	No. of No-Vest
25/9/2019	4	1	0

(b) Example of summary safety report

Figure 4-9 An example of technical and high-level safety reports

4.4 Single-camera Localization (SL) Module

The surveillance videos are used to validate the proposed method in Section 3.4 for locating workers in a real-world 2D coordinate system, and the steps are explained in the following sections.

4.4.1. Data Collection

In order to validate the proposed module, surveillance video in which workers are closely working with equipment is selected. Different equipment types are working at the construction sites (e.g., excavators, cranes, etc.), which could be considered potential struck-by accident sources. The video is used to detect the workers by the proposed worker detection model explained in Section 3.2.1 and other available models trained to detect construction sites' equipment [23]. An example of the defined condition is shown in Figure 4-10, where workers work close to the equipment.

Figure 4-10 An example of workers close to the equipment

4.4.2. CV Based Localization

The proposed CV based localization technique must be validated first in order to validate the proposed SL module. Validation data is collected by defining and measuring three areas with known dimensions (i.e., ground truth) using traffic cones, as shown in Figure 4-11. The pixel coordinates of each corner of defined rectangles are selected and transformed into real-world measures, and then the distance between the two points is measured and compared to known dimensions.

Figure 4-11 The defined rectangle zones on the construction site

In order to perform the perspective transformation, the parameters of Equation 3-8 must be extracted from the camera. The surveillance camera is calibrated using two check boards shown in Figure 4-12 with sizes of 10×10 *cm* and 20×20 *cm* and Matlab's camera calibration toolbox [53]. The first step after getting the calibration parameters is to find the scale factor (*s*). In order to calculate the *s,* a point is selected on the image frame, and knowing that the *z=0, s* is calculated using the last row of the Equation 3-8.

Figure 4-12 Example of image frames with two check boards for calibration

Finally, having the camera's parameters and *s*, pixel coordinates from each corner of defined rectangles are transferred into the real-world coordinates, and the distance between them is measured and compared with the known distance. For each rectangle, six dimensions (i.e., four sides and two diameters) are measured. The average error and standard deviations are 1.58 *m* and 1.03, respectively. The standard error of average [62] of the proposed SL module is 0.24 *m,* which indicates how much variation in the average to expect from the ground truth if the experiment were to be repeated n times, assuming that the measurements are unbiased.

4.4.3. Case Study

In order to illustrate the potential usage of the proposed SL module, a short video is selected from surveillance cameras where workers are working close to the equipment. Workers and equipment are detected in the videos, and the bottom center of the detection bounding box is used as the main point for localization. The distance between workers and equipment is calculated, and the results are summarized in Figure 4-13.

a) Equipment 0 - worker 1:

1.25m

b) Equipment 0 - worker 8:

2.05m

Figure 4-13 Example of captured near-miss event

4.5 Summary

The implementation environment, data collection procedure, and evaluation of each proposed module are explained in this section. The proposed framework is implemented in Python programming language using state-of-the-art deep learning CV algorithms. The evaluation and training data is collected from a construction site located in Montreal, Canada. The evaluation results of the PPED module illustrates the effectiveness of the proposed frame segmentation method in improving the worker and PPE detection result at far-field views of the construction sites.

Similarly, the PPESRG module is also validated using two cameras with overlapping FoV, and evaluation results illustrated the necessity of using multiple cameras for generating reliable safety reports. Finally, the SL module is evaluated on surveillance videos of the construction site. The results indicated that the average error for locating workers or equipment is 1.58 *m,* which indicates the practicality of the proposed technique in monitoring workers' safety based on their location, classifying different groups of workers and equipment working together on construction sites.

Chapter 5 SUMMARY, CONCLUSIONS AND FUTURE WORK

This chapter starts with a review of the overall proposed framework, followed by discussion and conclusions, and finally, the limitations and recommendations for future works are discussed.

5.1 Summary and Conclusion of the Proposed Framework

The proposed framework consists of three main parts of the PPED module, PPESRG module, and SL module. The PPED module is a nested network based on the frame segmentation technique, aiming to detect workers and their PPE in near, mid, and far-field views. The proposed frame segmentation technique divides the video frames into main and then sub-segments based on the workers' average width in pixels so that there are more sub-segments defined in the far-field segment and fewer in the near-field segment. The defined sub-segments are fed into the custom trained, Faster R-CNN model to detect workers. The CT algorithm tracks the detected workers, and the detected workers' bounding boxes are input to the second custom trained Faster R-CNN model, which detects four PPE classes. The PPED results are logged and are further used to generate PPE safety reports.

The worker detection model in the PPED module is validated on four videos from different construction work phases with an average precision of 99.04%, an average recall of 95.13%, and average accuracy of 94.26%. Also, the PPE detection model is validated on 200 images with four classes of PPE (i.e., hardhat, no-hardhat, safety-vest, no-safety-vest), with an average precision of 100%, an average recall of 96.30%, and average accuracy of 96.30%. The PPED module has an overall precision of 99.04%, a recall of 91.61%, and an accuracy of 90.77%. The results indicated that the proposed PPED module overcomes the far-field PPE detection challenge when the workers are only captured in a single surveillance camera view.

In order to generate accurate and practical safety reports, the proposed method by [49] is adopted in this study to match workers from two camera views. The results of the PPED module from two camera views are used to match the detected workers in synchronized frames, after matching the workers, detected PPE classes are compared for matched workers, and the ones with higher confidence level are selected. On the other hand, for not matching workers, the available PPE detection results from one camera are considered in the PPE safety reports.

Detailed and summary safety reports are generated based on the PPE detection and worker matching results. Detailed safety reports contain more technical details about detection such as frame number, detection confidence level, tracking ID, etc. However, in order to protect workers' privacy and have practical safety reports for safety managers, summary safety reports are generated. The summary safety reports give an overview of safety status for a specific day while protecting privacy. The PPESRG module is validated using videos of two surveillance cameras from the construction site. The test results indicate the proposed module's effectiveness for generating accurate and practical safety reports benefiting from two cameras when the workers are occluded or far from the other camera view. The safety managers validated the generated safety reports as being practical for monitoring safety on the construction sites.

Construction workers' safety status can also be identified based on their location on the construction site. The proposed SL module uses the detection and tracking results of the PPED method and camera calibration results to locate workers from image coordinate to a real-world 2D coordinate system. Finally, by having workers' location on a 2D real-world coordinate system, the workers' safety status is identified as the distance to mobile equipment or defined high-risk zones on the construction sites. The SL module is validated using surveillance videos in which areas are defined with known dimensions. The validation results indicate the average error of 1.58 *m* in locating workers on a real-world 2D coordinate system. The proposed method can locate workers or other construction entities in different zones and define a group of workers or equipment working together.

5.2 Limitations and Future Work

This research focused on detecting PPE and generating safety reports that can be used by safety managers. Test results guide future researchers in improving PPE detection and transferring them into more detailed safety reports by benefiting from other data sources. The adopted worker matching technique highly depends on the overlaps between the two camera views; therefore, having video data with high overlap improves worker matching reliability and generated PPE safety reports. In order to achieve more accurate worker localization results, a stereo camera setup can be used on the construction site to locate workers and other entities. Additionally, to improve safety monitoring in the future, Pan-Tilt-Zoom (PTZ) cameras can be used to have the ability to orient the cameras based on the work schedule. Furthermore, investigating the use of more than two cameras to overcome the far-field detection challenge and having more overlap between the views.

References

[1] Association of Workers' Compensation Boards of Canada (AWCBC)-Statistics, http://awcbc.org/?page_id=14.

[2] CCOHS: Canadian Centre for Occupational Health and Safety, Oct. 02, 2019. https://www.ccohs.ca.

[3] Construction workers: 3 or 4 times more accidents - SPI Health and Safety, https://www.spi-s.com/en/blog/item/construction-workers-3-or-4-times-more-accidents.

[4] The 5 most common workplace accidents on construction sites | Aviva, https://www.aviva.ca/en/business/blog/5-most-common-workplace-accidents-on-construction-sites.

[5] K. Arunkumar and J.Gunasekaran, "Causes and Effects of Accidents on Construction Site," International Journal of Engineering Science and Computing, June 2018, https://doi:10.30880/ijscet.2019.10.02.003.

[6] Infrastructure Health & Safety Association, https://www.ihsa.ca/Homepage.aspx.

[7] National Safety Council-Near-Miss-Reporting," https://www.nsc.org/work-safety/tools-resources/near-miss-reporting.

[8] Employment and Social Development Canada, "Workplace Safety," *aem*, Feb. 11, 2009. https://www.canada.ca/en/employment-social-development/services/health-safety/workplace-safety.html.

[9] Occupational Safety and Health Administration-Personal Protective Equipment - Head Protection, https://www.osha.gov/SLTC/etools/logging/manual/logger/head_protection.html.

[10] CCOHS: Canadian Centre for Occupational Health and Safety, "High-Visibility Safety Apparel: OSH Answers," Aug. 14, 2019. http://www.ccohs.ca.

[11] U.S. Bureau of Labor Statistics. https://www.bls.gov/

[12] S. Dong, Q. He, H. Li, and Q.Yin, "Automated PPE Misuse Identification and Assessment for Safety Performance Enhancement," *ICCREM 2015*, pp. 204–214, https://doi: 10.1061/9780784479377.024.

[13] A.Kelma, L.Laußata, A, Meins-Beckera, D.Platza, M. J.Khazaeea, A.M.Costinb, M.Helmusa, J.Teizerb, "Mobile passive Radio Frequency Identification (RFID) portal for automated and rapid control of Personal Protective Equipment (PPE) on construction sites," *Automation in Construction*, vol. 36, pp. 38–52, Dec. 2013, https://doi: 10.1016/j.autcon.2013.08.009.

[14] H. Li, M. Lu, S.-C. Hsu, M. Gray, and T. Huang, "Proactive behavior-based safety management for construction safety improvement," *Safety Science*, vol. 75, pp. 107–117, Jun. 2015, https://doi: 10.1016/j.ssci.2015.01.013.

[15] J. Lucas, J. Burgett, A. Hoover, and M. Gungor, "Use of Ultra-Wideband Sensor Networks to Detect Safety Violations in Real Time," *ISARC Proceedings*, pp. 250–257, Jul. 2016.

[16] H. Siddiqui, F. Vahdatikhaki, and A. Hammad, "Case study on application of wireless ultra-wideband technology for tracking equipment on a congested site," *Journal of Information Technology in Construction (ITcon)*, vol. 24, no. 10, pp. 167–187, May 2019.

[17] H. Zhang, X.Yan, H. Li, R. Jin, and H.F. Fu, "Real-Time Alarming, Monitoring, and Locating for Non-Hard-Hat Use in Construction," *Journal of Construction Engineering and Management*, vol. 145, no. 3, p. 04019006, Mar. 2019, https://doi: 10.1061/(ASCE)CO.1943-7862.0001629.

[18] H. Siddiqui, UWB RTLS for Construction Equipment Localization: Experimental Performance Analysis and Fusion with Video Data. Master's Thesis, Dept. of Information Systems Engineering, Concordia University, Montréal, QC, Canada, 2014.

[19] CCOHS: Canadian Centre for Occupational Health and Safety, "Construction Worker - General: OSH Answers," Jun. 30, 2020. https://www.ccohs.ca.

[20] CCOHS: Canadian Centre for Occupational Health and Safety, "Effective Workplace Inspections: OSH Answers," Jun. 30, 2020. https://www.ccohs.ca.

[21] Occupational Safety and Health Administration,1926.100 - Head protection," https://www.osha.gov/laws-regs/regulations/standardnumber/1926/1926.100.

[22] A.Kelma, L.Laußata, A, Meins-Beckera, D.Platza, M. J.Khazaeea, A.M.Costinb, M.Helmusa, J.Teizerb, "Mobile passive Radio Frequency Identification (RFID) portal for automated and rapid control of Personal Protective Equipment (PPE) on construction sites," *Automation in Construction*, vol. 36, pp. 38–52, Dec. 2013, https://doi: 10.1016/j.autcon.2013.08.009.

[23] C. Chen, Z. Zhu, and A. Hammad, "Automated excavators activity recognition and productivity analysis from construction site surveillance videos," *Automation in Construction*, vol. 110, p. 103045, Feb. 2020, https://doi: 10.1016/j.autcon.2019.103045.

[24] X. Luo, H. Li, D. Cao, Y. Yu, X. Yang, and T. Huang, "Towards efficient and objective work sampling: Recognizing workers' activities in site surveillance videos with two-stream convolutional networks," *Automation in Construction*, vol. 94, pp. 360–370, Oct. 2018, https://doi: 10.1016/j.autcon.2018.07.011.

[25] H. Kim, K. Kim, and H. Kim, "Vision-Based Object-Centric Safety Assessment Using Fuzzy Inference: Monitoring Struck-By Accidents with Moving Objects," *Journal of*

Computing in Civil Engineering, vol. 30, no. 4, p. 04015075, Jul. 2016, https://doi: 10.1061/(ASCE)CP.1943-5487.0000562.

[26] M. Zhang, Z. Cao, Z. Yang, and X. Zhao, "Utilizing Computer Vision and Fuzzy Inference to Evaluate Level of Collision Safety for Workers and Equipment in a Dynamic Environment," *Journal of Construction Engineering and Management*, vol. 146, no. 6, p. 04020051, Jun. 2020, https://doi: 10.1061/(ASCE)CO.1943-7862.0001802.

[27] X. Yan, H. Zhang, and H. Li, "Computer vision-based recognition of 3D relationship between construction entities for monitoring struck-by accidents," Computer-Aided Civil and Infrastructure Engineering, vol. n/a, no. n/a, https://doi: 10.1111/mice.12536.

[28] S. Han and S. Lee, "A vision-based motion capture and recognition framework for behavior-based safety management," *Automation in Construction*, vol. 35, pp. 131–141, Nov. 2013, https://doi: 10.1016/j.autcon.2013.05.001.

[29] Centers for Disease Control and Prevention - NIOSH Program Portfolio, Musculoskeletal Disorders: Program Description, https://www.cdc.gov/niosh/programs/msd/default.html.

[30] M.-W. Park and I. Brilakis, "Continuous localization of construction workers via integration of detection and tracking," *Automation in Construction*, vol. 72, pp. 129–142, Dec. 2016, https://doi: 10.1016/j.autcon.2016.08.039.

[31] Z. Zhu, M.-W. Park, C. Koch, M. Soltani, A. Hammad, and K. Davari, "Predicting movements of onsite workers and mobile equipment for enhancing construction site safety," *Automation in Construction*, vol. 68, pp. 95–101, Aug. 2016, https://doi: 10.1016/j.autcon.2016.04.009.

[32] M.-W. Park and I. Brilakis, “Construction worker detection in video frames for initializing vision trackers,” *Automation in Construction*, vol. 28, pp. 15–25, Dec. 2012, https://doi: 10.1016/j.autcon.2012.06.001.

[33] M.-W. Park, N. Elsafty, and Z. Zhu, “Hardhat-Wearing Detection for Enhancing On-Site Safety of Construction Workers,” *Journal of Construction Engineering and Management*, vol. 141, no. 9, p. 04015024, Sep. 2015, https://doi: 10.1061/(ASCE)CO.1943-7862.0000974.

[34] K. Shrestha, P. P. Shrestha, D. Bajracharya, and E. A. Yfantis, “Hard-Hat Detection for Construction Safety Visualization,” *Journal of Construction Engineering*, 2015. https://www.hindawi.com/journals/jcen/2015/721380.

[35] B.E. Mneymneh, M. Abbas, and H. Khoury, “Vision-Based Framework for Intelligent Monitoring of Hardhat Wearing on Construction Sites,” *Journal of Computing in Civil Engineering*, vol. 33, no. 2, p. 04018066, Mar. 2019, https://doi: 10.1061/(ASCE)CP.1943-5487.0000813.

[36] Q.Fang, H.Li, X.Luo, L.Ding, H.Luo, T.M.Rose, W.An, “Detecting non-hardhat-use by a deep learning method from far-field surveillance videos,” *Automation in Construction*, vol. 85, pp. 1–9, Jan. 2018, https://doi: 10.1016/j.autcon.2017.09.018.

[37] W. Fang, L. Ding, H. Luo, and P. E. D. Love, “Falls from heights: A computer vision-based approach for safety harness detection,” *Automation in Construction*, vol. 91, pp. 53–61, Jul. 2018, https://doi: 10.1016/j.autcon.2018.02.018.

[38] Z. Xie, H. Liu, Z. Li, and Y. He, “A convolutional neural network based approach towards real-time hard hat detection,” in *2018 IEEE International Conference on Progress in*

Informatics and Computing (PIC), Dec. 2018, pp. 430–434, https://doi: 10.1109/PIC.2018.8706269.

[39] J. Wu, N. Cai, W. Chen, H. Wang, and G. Wang, "Automatic detection of hardhats worn by construction personnel: A deep learning approach and benchmark dataset," *Automation in Construction*, vol. 106, p. 102894, Oct. 2019, https://doi: 10.1016/j.autcon.2019.102894.

[40] N. D. Nath, A. H. Behzadan, and S. G. Paal, "Deep learning for site safety: Real-time detection of personal protective equipment," *Automation in Construction*, vol. 112, p. 103085, Apr. 2020, https://doi: 10.1016/j.autcon.2020.103085.

[41] L. Wang, L. Xie, P. Yang, Q. Deng, S. Du, and L. Xu, "Hardhat-Wearing Detection Based on a Lightweight Convolutional Neural Network with Multi-Scale Features and a Top-Down Module," *Sensors (Basel)*, vol. 20, no. 7, Mar. 2020, https://doi: 10.3390/s20071868.

[42] J. Jung, S.-Y. Seo, S.-C. Lee, and Y.-S. Ho, "Enhanced Linear Perspective using Adaptive Intrinsic Camera Parameters," 2010.

[43] M. Akbarzadeh, Z. Zhu, and A. Hammad, "Nested Network for Detecting PPE on Large Construction Sites Based on Frame Segmentation," in *Proceedings of the Creative Construction e-Conference 2020*, Online, 2020, pp. 33–38, https://doi: 10.3311/CCC2020-006.

[44] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, https://doi: 10.1109/TPAMI.2016.2577031.

[45] H.C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers, "Deep Convolutional Neural Networks for Computer-Aided Detection: CNN

Architectures, Dataset Characteristics and Transfer Learning," *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1285–1298, May 2016, https://doi: 10.1109/TMI.2016.2528162.

[46] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009, pp. 248–255, https://doi: 10.1109/CVPR.2009.5206848.

[47] X. Chen, H. Fang, T.-Y. Lin, R. Vedantam, S. Gupta, P. Dollar, C. L. Zitnick, "Microsoft COCO Captions: Data Collection and Evaluation Server," *arXiv:1504.00325 [cs]*, Apr. 2015, Available: http://arxiv.org/abs/1504.00325.

[48] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, Mar. 1982, https://doi: 10.1109/TIT.1982.1056489.

[49] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," *arXiv:1612.08242 [cs]*, Dec. 2016, Available: http://arxiv.org/abs/1612.08242.

[50] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *2017 IEEE International Conference on Image Processing (ICIP)*, Sep. 2017, pp. 3645–3649, https://doi: 10.1109/ICIP.2017.8296962.

[51] B. Zhang, Z. Zhu, A. Hammad, and W. Aly, "Automatic matching of construction onsite resources under camera views," *Automation in Construction*, vol. 91, pp. 206–215, Jul. 2018, https://doi: 10.1016/j.autcon.2018.03.011.

[52] Surveillance in the Workplace, Employment & Human Rights Law in Canada, Mar. 12, 2020. https://www.canadaemploymenthumanrightslaw.com/2020/03/surveillance-in-the-workplace

[53] Z. Zhang, “Flexible camera calibration by viewing a plane from unknown orientations,” in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, Sep. 1999, vol. 1, pp. 666–673 vol.1, https://doi: 10.1109/ICCV.1999.791289.

[54] J. Y. Bouguet, 2004. Camera calibration toolbox for MATLAB. Santa Clara, CA: Intel Corporation.

[55] Camera Calibration and 3D Reconstruction — OpenCV 2.4.13.7 documentation, https://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html.

[56] J.Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, K. Murphy, “Speed/accuracy trade-offs for modern convolutional object detectors,” *arXiv:1611.10012 [cs]*, Apr. 2017, Accessed: Jul. 13, 2020. [Online]. Available: http://arxiv.org/abs/1611.10012.

[57] M. Beyeler, *Machine Learning for OpenCV*. Packt Publishing Ltd, 2017.

[58] Compute Canada - Calcul Canada, *Compute Canada - Calcul Canada*. https://www.computecanada.ca.

[59] W. Li, R. Zhao, T. Xiao, and X. Wang, “DeepReID: Deep Filter Pairing Neural Network for Person Re-Identification,” 2014, pp. 152–159, Available: https://openaccess.thecvf.com/content_cvpr_2014/html/Li_DeepReID_Deep_Filter_2014_CVPR_paper.html.

[60] T. Zutalin, https://github.com/tzutalin/labelImg.

[61] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The Pascal Visual Object Classes (VOC) Challenge,” *Int J Comput Vis*, vol. 88, no. 2, pp. 303–338, Jun. 2010, https://doi: 10.1007/s11263-009-0275-4.

[62] M. P. Barde and P. J. Barde, “What to use to express the variability of data: Standard deviation or standard error of mean?,” *Perspect Clin Res*, vol. 3, no. 3, pp. 113–116, 2012, https://doi: 10.4103/2229-3485.100662.

Appendices

Appendix A. Procedure for Running Nested Network Detection

In order to run the proposed nested network, modifications must be made only in the "main.py" script (Appendix B), which initiates the detection. The default setting for the detection are: (1) 1detection/sec, (2) N=25. The other detection parameters, such as workers' size on the image plane, could be relatively changed in the "nested_detection.py" script. Additionally, the CT tracking algorithm is imported in the "nested_detection.py," which can be downloaded from opensource repositories online. The final results of the detection, which are the output video with the bounding boxes and detection results as a "jason" file, are stored in the output directory.

Appendix B. Python Code of Developed Nested Network (Main.py)

```
# Author Mohammad Akbarzadeh
# https://github.com/mohammadakz

# Importing nested detection class
import os
from nested_detection import Nested_Detection

# Model and video file names and path
CWD_PATH = os.getcwd()

model_name = 'inference_graph_50000_balanced'
video_name = 'construction.mp4'
out_put = 'out_put/results_{}.avi'.format(video_name[:-4])
label_path = 'label_map_person.pbtxt'

# Performing the detection
dt = Nested_Detection()
dt.worker_detection(model_name, label_path, video_name, out_put)
```

Appendix C. Python Code of Nested Detection (nested_detection.py)

```
# Author Mohammad Akbarzadeh

# Import packages
import os
import sys
import cv2
import math
from shapely.geometry import Polygon
from pyimagesearch.centroidtracker import CentroidTracker
import numpy as np
import tensorflow as tf
import json

sys.path.append("..")

# Import utilities
from utils import label_map_util
from utils import visualization_utils as vis_util


class Nested_Detection():

    def __init__(self):
        self.NUM_WORKER_CLASSES = 1
        self.frame_index = 0
        self.small_worker = 20
        self.mid_worker = 35
        self.large_worker = 55
        self.n_small = 25
        self.n_mid = 25
        self.n_large = 25
        self.skip_frames = 30
        self.font = cv2.FONT_HERSHEY_PLAIN
        self.rectangle_bgr = (255, 255, 255)
        self.detection_results = []
        self.track_list = []
        self.ct = CentroidTracker()
        self.util_match_table = {}
        self.util_detection_results = {}

    def sub_regions(self, img, width, height):

        num_small = int((width / self.small_worker) / self.n_small) + 1
        self.small_width = int((width / num_small))
```

```
        num_mid = int((width / self.mid_worker) / self.n_mid) + 1
        self.mid_width = int((width / num_mid))

        num_large = int((width / self.large_worker) / self.n_large) + 1
        self.large_width = int((width / num_large))

        # (x,y) # (0,0) till (1920, 540)
        up_left_small = (0, 0)
        down_right_small = (width, int((height / 2)))

        # (0,270) till (1920, 810)
        up_left_mid = (0, int((height / 4)))
        down_right_mid = (width, height - int(height / 4))

        # (0,540) till (1920, 540)
        up_left_large = (0, int((height / 2)))
        down_right_large = (width, height)

        # (y:y, x:x)
        main_small_region = img[up_left_small[1]:down_right_small[1],
up_left_small[0]:down_right_small[0]]
        main_mid_region = img[up_left_mid[1]:down_right_mid[1],
up_left_mid[0]:down_right_mid[0]]
        main_large_region = img[up_left_large[1]:down_right_large[1],
up_left_large[0]:down_right_large[0]]

        small_sub_regions = []
        small_sub_regions_overlap = []

        mid_sub_regions = []
        mid_sub_regions_overlap = []

        large_sub_regions = []
        large_sub_region_overlap = []

        for i in range(0, width, self.small_width):
            small = main_small_region[:, i:i + self.small_width]
            small_sub_regions.append(small)

        for i in range(int(self.small_width / 2), (width - int(self.small_width / 2)), self.small_width):
            small_overlap = main_small_region[:, i:i + self.small_width]
            small_sub_regions_overlap.append(small_overlap)

        for i in range(0, width, self.mid_width):
            mid = main_mid_region[:, i:i + self.mid_width]
```

```
            mid_sub_regions.append(mid)

        for i in range(int(self.mid_width / 2), (width - int(self.mid_width / 2)), self.mid_width):
            mid_overlap = main_mid_region[:, i:i + self.mid_width]
            mid_sub_regions_overlap.append(mid_overlap)

        for i in range(0, width, self.large_width):
            large = main_large_region[:, i:i + self.large_width]
            large_sub_regions.append(large)

        for i in range(int(self.large_width / 2), (width - int(self.large_width / 2)), self.large_width):
            large_overlap = main_large_region[:, i:i + self.large_width]
            large_sub_region_overlap.append(large_overlap)
        return small_sub_regions, small_sub_regions_overlap, mid_sub_regions,
mid_sub_regions_overlap, \
               large_sub_regions, large_sub_region_overlap

    def calculate_iou(self, box_1, box_2):
        poly_1 = Polygon(box_1)
        poly_2 = Polygon(box_2)
        iou = poly_1.intersection(poly_2).area / poly_1.union(poly_2).area
        return iou

    def remove_duplicates(self, detected_workers):
        duplications = []
        i = 0
        while i <= len(detected_workers):
            j = i + 1
            while j < len(detected_workers):
                iou = self.calculate_iou(
                    [[detected_workers[i][0], detected_workers[i][2]], [detected_workers[i][1],
detected_workers[i][2]],
                     [detected_workers[i][1], detected_workers[i][3]],
                     [detected_workers[i][0], detected_workers[i][3]]],
                    [[detected_workers[j][0], detected_workers[j][2]], [detected_workers[j][1],
detected_workers[j][2]],
                     [detected_workers[j][1], detected_workers[j][3]],
                     [detected_workers[j][0], detected_workers[j][3]]])

                if iou > 0.50:
                    # or abs(detected_workers[i][0] - detected_workers[j][0]) <= 40 and abs(
                    # detected_workers[i][2] - detected_workers[j][2]) <= 40 and abs(
                    # detected_workers[i][1] - detected_workers[j][1]) <= 40:
                    duplications.append(detected_workers[j])
                j += 1
            i += 1
```

```
        return [x for x in detected_workers if x not in duplications]

    def area_checking(self, refined_workers):
        for worker in range(0, len(refined_workers)):
            height = abs(refined_workers[worker][3] - refined_workers[worker][2])
            width = abs(refined_workers[worker][1] - refined_workers[worker][0])
            center_x = int(width / 2) + refined_workers[worker][0]
            center_y = int(height / 2) + refined_workers[worker][2]
            area = width * height
            refined_workers[worker].append(area)
            refined_workers[worker].append(center_x)
            refined_workers[worker].append(center_y)

        for i in refined_workers:
            for j in range(len(i)):
                if j == 2:
                    if i[j] > 540 and i[-3] < 1200:
                        refined_workers.remove(i)
        return refined_workers

    def worker_tracking(self, tracking_list):
        object_tracker = self.ct.update(tracking_list)
        for (objectID, centeroid) in object_tracker.items():
            text_track = "W {}".format(objectID)
            cv2.putText(self.frame, text_track, (centeroid[0], centeroid[1] + 60),
cv2.FONT_HERSHEY_SIMPLEX, 0.75,
                        (0, 0, 255), 2)
            cv2.circle(self.frame, (centeroid[0], centeroid[1] + 50), 4, (0, 0, 255), -1)

        for (objectID, centeroid) in object_tracker.items():
            worker_match = []
            worker_min_distance = 50
            for worker_location in tracking_list:
                # distance = math.sqrt(
                #     ((worker_location[-2] - centeroid[0]) ** 2) + ((worker_location[-1] - centeroid[1])
** 2))
                # if distance < worker_min_distance:
                #     worker_min_distance = distance
                worker_match = worker_location

            self.util_match_table[objectID] = {"worker_location": worker_match}
        self.util_detection_results[self.frame_index] = self.util_match_table;

    def draw_bounding_box_workers(self, final_refined_workers):
```

```
        for i in range(0, len(final_refined_workers)):
            cv2.putText(self.frame, "Person: {}".format(str(len(final_refined_workers))), (10, 20),
self.font, 2,
                        (0, 255, 255), 2,
                        cv2.FONT_HERSHEY_SIMPLEX)

            text = "Person: {}%".format(str(final_refined_workers[i][4]))
            (text_width, text_height) = cv2.getTextSize(text, self.font, 1, thickness=1)[0]

            box_coords = (
                (final_refined_workers[i][0] - 10, final_refined_workers[i][2] - 10),
                (
                    final_refined_workers[i][0] + text_width,
                    (final_refined_workers[i][2] - text_height - 10)))

            cv2.rectangle(self.frame, box_coords[0], box_coords[1], self.rectangle_bgr,
cv2.FILLED)

            cv2.putText(self.frame, "Person: {}%".format(str(final_refined_workers[i][4])),
                        (final_refined_workers[i][0] - 10, final_refined_workers[i][2] - 10),
                        self.font, 1,
                        (0, 0, 0), cv2.FONT_HERSHEY_PLAIN)

            roi_person = self.frame_rgb[
                         abs(final_refined_workers[i][2] - 10):abs(final_refined_workers[i][3] + 10),
                         abs(final_refined_workers[i][0] - 10):abs(final_refined_workers[i][1]) + 10]

            self.track_list.append([final_refined_workers[i][0], final_refined_workers[i][2],
                                    final_refined_workers[i][1], final_refined_workers[i][3]])

            cv2.rectangle(self.frame,
                          (abs(final_refined_workers[i][0] - 10), abs(final_refined_workers[i][2] - 10)),
                          (final_refined_workers[i][1] + 10, final_refined_workers[i][3] + 10),
                          (0, 255, 0), 1)

    def worker_detection(self, model_name, label_path, video_name, out_put):
        video = cv2.VideoCapture(video_name)
        frame_width = int(video.get(3))
        frame_height = int(video.get(4))
        out = cv2.VideoWriter(out_put, cv2.VideoWriter_fourcc(*'XVID'),
                              30, (frame_width, frame_height))

        while video.isOpened():
            ret, self.frame = video.read()
            if ret == True:
                self.frame_rgb = cv2.cvtColor(self.frame, cv2.COLOR_BGR2RGB)
```

```
            if self.frame_index % self.skip_frames == 0:
                s_regions, s_regions_overlap, m_regions, m_regions_overlap, \
                l_regions, l_regions_overlap = self.sub_regions(self.frame_rgb, frame_width,
frame_height)

                path_to_check = os.path.join(model_name, 'frozen_inference_graph.pb')
                categories = label_map_util.convert_label_map_to_categories(
                    label_map_util.load_labelmap(label_path),
                    max_num_classes=self.NUM_WORKER_CLASSES,
                    use_display_name=True)
                category_index = label_map_util.create_category_index(categories)
                # Load the Tensorflow model into memory.
                detection_graph = tf.Graph()
                with detection_graph.as_default():
                    od_graph_def = tf.GraphDef()
                    with tf.gfile.GFile(path_to_check, 'rb') as fid:
                        serialized_graph = fid.read()
                        od_graph_def.ParseFromString(serialized_graph)
                        tf.import_graph_def(od_graph_def, name='')

                    sess = tf.Session(graph=detection_graph)

                image_tensor = detection_graph.get_tensor_by_name('image_tensor:0')
                detection_boxes = detection_graph.get_tensor_by_name('detection_boxes:0')
                detection_scores = detection_graph.get_tensor_by_name('detection_scores:0')
                detection_classes = detection_graph.get_tensor_by_name('detection_classes:0')
                num_detections = detection_graph.get_tensor_by_name('num_detections:0')

                for idx, subregion in enumerate(s_regions):
                    frame_expanded_1 = np.expand_dims(subregion, axis=0)
                    # Perform the actual detection by running the model with the image as input
                    (boxes, scores, classes, num) = sess.run(
                        [detection_boxes, detection_scores, detection_classes, num_detections],
                        feed_dict={image_tensor: frame_expanded_1})

                    coordinates_1 = vis_util.return_coordinates(
                        subregion,
                        self.frame_index,
                        np.squeeze(boxes),
                        np.squeeze(classes).astype(np.int32),
                        np.squeeze(scores),
                        category_index,
                        use_normalized_coordinates=True,
                        line_thickness=8,
                        min_score_thresh=0.50)
```

```
                for coordinate in coordinates_1:
                    (ymin, ymax, xmin, xmax, acc, classification) = coordinate
                    self.detection_results.append(
                        [(xmin + (idx * self.small_width)), (xmax + (idx * self.small_width)), ymin,
ymax,
                         int(acc),
                         classification])

            for idx, subregion in enumerate(s_regions_overlap):
                frame_expanded_1 = np.expand_dims(subregion, axis=0)
                # Perform the actual detection by running the model with the image as input
                (boxes, scores, classes, num) = sess.run(
                    [detection_boxes, detection_scores, detection_classes, num_detections],
                    feed_dict={image_tensor: frame_expanded_1})

                coordinates_1 = vis_util.return_coordinates(
                    subregion,
                    self.frame_index,
                    np.squeeze(boxes),
                    np.squeeze(classes).astype(np.int32),
                    np.squeeze(scores),
                    category_index,
                    use_normalized_coordinates=True,
                    line_thickness=8,
                    min_score_thresh=0.50)

                for coordinate in coordinates_1:
                    (ymin, ymax, xmin, xmax, acc, classification) = coordinate
                    self.detection_results.append(
                        [(xmin + (idx * self.small_width) + (int(self.small_width / 2))),
                         (xmax + (idx * self.small_width) + (int(self.small_width / 2))), ymin, ymax,
int(acc),
                         classification])

            for idx, subregion in enumerate(m_regions):
                frame_expanded_1 = np.expand_dims(subregion, axis=0)
                # Perform the actual detection by running the model with the image as input
                (boxes, scores, classes, num) = sess.run(
                    [detection_boxes, detection_scores, detection_classes, num_detections],
                    feed_dict={image_tensor: frame_expanded_1})

                coordinates_1 = vis_util.return_coordinates(
                    subregion,
                    self.frame_index,
                    np.squeeze(boxes),
                    np.squeeze(classes).astype(np.int32),
```

```
            np.squeeze(scores),
            category_index,
            use_normalized_coordinates=True,
            line_thickness=8,
            min_score_thresh=0.50)

        for coordinate in coordinates_1:
            (ymin, ymax, xmin, xmax, acc, classification) = coordinate
            self.detection_results.append(
                [(xmin + (idx * self.mid_width)), (xmax + (self.mid_width * idx)),
                 (ymin + int(frame_height / 4)),
                 (ymax + int(frame_height / 4)), int(acc),
                 classification])

    for idx, subregion in enumerate(m_regions_overlap):
        frame_expanded_1 = np.expand_dims(subregion, axis=0)
        # Perform the actual detection by running the model with the image as input
        (boxes, scores, classes, num) = sess.run(
            [detection_boxes, detection_scores, detection_classes, num_detections],
            feed_dict={image_tensor: frame_expanded_1})

        coordinates_1 = vis_util.return_coordinates(
            subregion,
            self.frame_index,
            np.squeeze(boxes),
            np.squeeze(classes).astype(np.int32),
            np.squeeze(scores),
            category_index,
            use_normalized_coordinates=True,
            line_thickness=8,
            min_score_thresh=0.50)

        for coordinate in coordinates_1:
            (ymin, ymax, xmin, xmax, acc, classification) = coordinate
            self.detection_results.append(
                [(xmin + (idx * self.mid_width) + (int(self.mid_width / 2))),
                 (xmax + (idx * self.mid_width) + (int(self.mid_width / 2))),
                 (ymin + int(frame_height / 4)),
                 (ymax + int(frame_height / 4)), int(acc),
                 classification])

    for idx, subregion in enumerate(l_regions):
        frame_expanded_1 = np.expand_dims(subregion, axis=0)
        # Perform the actual detection by running the model with the image as input
        (boxes, scores, classes, num) = sess.run(
            [detection_boxes, detection_scores, detection_classes, num_detections],
```

```
            feed_dict={image_tensor: frame_expanded_1})

        coordinates_1 = vis_util.return_coordinates(
            subregion,
            self.frame_index,
            np.squeeze(boxes),
            np.squeeze(classes).astype(np.int32),
            np.squeeze(scores),
            category_index,
            use_normalized_coordinates=True,
            line_thickness=8,
            min_score_thresh=0.50)

        for coordinate in coordinates_1:
            (ymin, ymax, xmin, xmax, acc, classification) = coordinate
            self.detection_results.append(
                [(xmin + (idx * self.large_width)), (xmax + (self.large_width * idx)), (
                      ymin + int(frame_height / 2)),
                 (ymax + int(frame_height / 2)), int(acc),
                 classification])

    for idx, subregion in enumerate(l_regions_overlap):
        frame_expanded_1 = np.expand_dims(subregion, axis=0)
        # Perform the actual detection by running the model with the image as input
        (boxes, scores, classes, num) = sess.run(
            [detection_boxes, detection_scores, detection_classes, num_detections],
            feed_dict={image_tensor: frame_expanded_1})

        coordinates_1 = vis_util.return_coordinates(
            subregion,
            self.frame_index,
            np.squeeze(boxes),
            np.squeeze(classes).astype(np.int32),
            np.squeeze(scores),
            category_index,
            use_normalized_coordinates=True,
            line_thickness=8,
            min_score_thresh=0.50)

        for coordinate in coordinates_1:
            (ymin, ymax, xmin, xmax, acc, classification) = coordinate
            self.detection_results.append(
                [(xmin + (idx * self.large_width) + (int(self.large_width / 2))),
                 (xmax + (idx * self.large_width) + (int(self.large_width / 2))),
                 (ymin + int(frame_height / 2)),
                 (ymax + int(frame_height / 2)), int(acc),
```

```
                    classification])

            if len(self.detection_results) == 0:
                print("Workers are not detected!")
            else:
                refine_workers_detection = self.remove_duplicates(self.detection_results)
                refined_area_workers = self.area_checking(refine_workers_detection)
                store_refined_detection = refined_area_workers
                self.draw_bounding_box_workers(refined_area_workers)
                self.worker_tracking(self.track_list)
                out.write(self.frame)
                del self.track_list[:]
                self.frame_index += 1
        else:
            out.write(self.frame)
            self.frame_index += 1
    else:
        break

with open("out_put/util_detection{}.json".format(video_name), "w") as file:
    j = json.dumps(self.util_detection_results)
    file.write(j)
```

Appendix D. Python Code of Safety Report (safety_report_generation.py)

```
# MA
''' This scripts uses worker matching results and PPE detection log from two cameras to generate detailed and summary safety reports '''

import json

with open('a/util_detection1.mp4.json') as f:
    camera_1 = json.load(f)

with open('a/util_detection1.mp4.json') as f:
    camera_2 = json.load(f)

# Worker matching results
matching_result = [(0, 0), (1, 1), (2, 3), (3, 2), (4, 4), (5, 5), (6, 6)]
matching_list = [list(ele) for ele in matching_result]
c1 = []
c2 = []
for i in matching_list:
    c1.append(i[0])
    c2.append(i[1])

# Detailed safety report
with open("a_safety/Detailed_report.csv", "w") as file:
    file.write("Date, hour, minutes, seconds, frame_number, worker_ID, H or NH, V or NV \n")

# Summary safety report
with open("a_safety/Summary_report.csv", "w") as file:
    file.write("Date, No_of_workers, No_of_no-hardhats, No_of_no-vest \n")

# Safety violences
NH = 0
NV = 0

# Time adjustment
Date = "2020/10/09"
hour = 7
minutes = 0
seconds = 0
for frame_number in range(0, 120, 30):
    print(frame_number)
    if frame_number % 30 == 0:
        seconds += 1
        if seconds % 60 == 0:
            minutes += 1
            if minutes % 60 == 0:
                hour += 1

    workers_camera_1 = (camera_1["{}".format(frame_number)])
    workers_camera_2 = (camera_2["{}".format(frame_number)])

    for workers_c1 in c1:
        for workers_c2 in c2:
            if workers_c1 == workers_c2:
                if len((workers_camera_1["{}".format(workers_c1)])["hat_location"]) > 0 and len(
                        (workers_camera_2["{}".format(workers_c2)])["hat_location"]) > 0 and len(
                    (workers_camera_1["{}".format(workers_c1)])["vest_location"]) > 0 and len(
                    (workers_camera_2["{}".format(workers_c2)])["vest_location"]) > 0:
                    if
(((workers_camera_1["{}".format(workers_c1)])["vest_location"][5]).split(":")[0]) == "NV":
                        NV += 1
                    if (((workers_camera_1["{}".format(workers_c1)])["hat_location"][5]).split(":")[0])
== "NH":
```

```
                NH += 1

            if ((workers_camera_1["{}".format(workers_c1)])["vest_location"][4]) >= (
                (workers_camera_2["{}".format(workers_c2)])["vest_location"][4]) and (
                (workers_camera_1["{}".format(workers_c1)])["hat_location"][4]) >= (
                (workers_camera_2["{}".format(workers_c2)])["hat_location"][4]):

                with open("a_safety/Detailed_report.csv", "a") as file:
                    file.write(Date + ',' + str(hour) + ',' + str(minutes) + ',' + str(seconds) + ',' + str(
                        frame_number) + ',' + str(workers_c1) + ',' + str(
                        (workers_camera_1["{}".format(workers_c1)])["hat_location"][5]) + ',' + str(
                        (workers_camera_1["{}".format(workers_c1)])["vest_location"][5]) + "\n")
            else:
                with open("a_safety/Detailed_report.csv", "a") as file:
                    file.write(Date + ',' + str(hour) + ',' + str(minutes) + ',' + str(seconds) + ',' + str(
                        frame_number) + ',' + str(workers_c2) + ',' + str(
                        (workers_camera_2["{}".format(workers_c2)])["hat_location"][5]) + ',' + str(
                        (workers_camera_2["{}".format(workers_c2)])["vest_location"][5]) + "\n")

#############################################################################################
        elif len((workers_camera_1["{}".format(workers_c1)])["hat_location"]) > 0 and len(
            (workers_camera_2["{}".format(workers_c2)])["hat_location"]) > 0 and len(
            (workers_camera_1["{}".format(workers_c1)])["vest_location"]) == 0 and len(
            (workers_camera_2["{}".format(workers_c2)])["vest_location"]) == 0:

            if (((workers_camera_1["{}".format(workers_c1)])["hat_location"][5]).split(":")[0])
== "NH":
                NH += 1
            if (((workers_camera_2["{}".format(workers_c2)])["hat_location"][5]).split(":")[0])
== "NH":
                NH += 1

            if ((workers_camera_1["{}".format(workers_c1)])["hat_location"][4]) >= (
                (workers_camera_2["{}".format(workers_c2)])["hat_location"][4]):
                with open("a_safety/Detailed_report.csv", "a") as file:
                    file.write(Date + ',' + str(hour) + ',' + str(minutes) + ',' + str(seconds) + ',' + str(
                        frame_number) + ',' + str(workers_c1) + ',' + str(
                        (workers_camera_1["{}".format(workers_c1)])["hat_location"][5]) + ',' + str(
                        "N/A") + "\n")
            else:
                with open("a_safety/Detailed_report.csv", "a") as file:
                    file.write(Date + ',' + str(hour) + ',' + str(minutes) + ',' + str(seconds) + ',' + str(
                        frame_number) + ',' + str(workers_c2) + ',' + str(
                        (workers_camera_2["{}".format(workers_c2)])["hat_location"][5]) + ',' + str(
                        'N/A') + "\n")

#############################################################################################
        elif len((workers_camera_1["{}".format(workers_c1)])["hat_location"]) == 0 and len(
            (workers_camera_2["{}".format(workers_c2)])["hat_location"]) == 0 and len(
            (workers_camera_1["{}".format(workers_c1)])["vest_location"]) > 0 and len(
            (workers_camera_2["{}".format(workers_c2)])["vest_location"]) > 0:

            if
(((workers_camera_1["{}".format(workers_c1)])["vest_location"][5]).split(":")[0]) == "NV":
                NV += 1
            if
(((workers_camera_2["{}".format(workers_c2)])["vest_location"][5]).split(":")[0]) == "NV":
                NV += 1

            if ((workers_camera_1["{}".format(workers_c1)])["vest_location"][4]) >= (
                (workers_camera_2["{}".format(workers_c2)])["vest_location"][4]):
                with open("a_safety/Detailed_report.csv", "a") as file:
                    file.write(Date + ',' + str(hour) + ',' + str(minutes) + ',' + str(seconds) + ',' + str(
                        frame_number) + ',' + str(workers_c1) + ',' + str("N/A") + ',' + str(
```

```
                    (workers_camera_1["{}".format(workers_c1)])["vest_location"][5]) + "\n")
            else:
                with open("a_safety/Detailed_report.csv", "a") as file:
                    file.write(Date + ',' + str(hour) + ',' + str(minutes) + ',' + str(seconds) + ',' + str(
                        frame_number) + ',' + str(workers_c2) + ',' + str("N/A") + ',' + str(
                        (workers_camera_2["{}".format(workers_c2)])["vest_location"][5]) + "\n")

#############################################################################
        elif len((workers_camera_1["{}".format(workers_c1)])["hat_location"]) > 0 and len(
                (workers_camera_2["{}".format(workers_c2)])["hat_location"]) == 0 and len(
            (workers_camera_1["{}".format(workers_c1)])["vest_location"]) == 0 and len(
            (workers_camera_2["{}".format(workers_c2)])["vest_location"]) > 0:

            if
(((workers_camera_2["{}".format(workers_c2)])["vest_location"][5]).split(":")[0]) == "NV":
                NV += 1
            if (((workers_camera_1["{}".format(workers_c1)])["hat_location"][5]).split(":")[0])
== "NH":
                NH += 1

            with open("a_safety/Detailed_report.csv", "a") as file:
                file.write(Date + ',' + str(hour) + ',' + str(minutes) + ',' + str(seconds) + ',' + str(
                    frame_number) + ',' + str(workers_c1) + ',' + str(
                    (workers_camera_1["{}".format(workers_c1)])["hat_location"][5]) + ',' +
                        str((workers_camera_2["{}".format(workers_c2)])["vest_location"][5]) +
"\n")

#############################################################################
        elif len((workers_camera_1["{}".format(workers_c1)])["hat_location"]) == 0 and len(
                (workers_camera_2["{}".format(workers_c2)])["hat_location"]) > 0 and len(
            (workers_camera_1["{}".format(workers_c1)])["vest_location"]) > 0 and len(
            (workers_camera_2["{}".format(workers_c2)])["vest_location"]) == 0:

            if
(((workers_camera_1["{}".format(workers_c1)])["vest_location"][5]).split(":")[0]) == "NV":
                NV += 1
            if (((workers_camera_2["{}".format(workers_c2)])["hat_location"][5]).split(":")[0])
== "NH":
                NH += 1

            with open("a_safety/Detailed_report.csv", "a") as file:
                file.write(Date + ',' + str(hour) + ',' + str(minutes) + ',' + str(seconds) + ',' + str(
                    frame_number) + ',' + str(workers_c1) + ',' + str(
                    (workers_camera_2["{}".format(workers_c2)])["hat_location"][5]) + ',' +
                        str((workers_camera_1["{}".format(workers_c1)])["vest_location"][5]) +
"\n")

#############################################################################
with open("a_safety/Summary_report.csv", "a") as file:
    file.write(Date + ',' + str(max(max(c1), max(c2))) + ',' + str(NH) + ',' + str(NV) + "\n")
```

Appendix E. Python Code for Worker Detection Evaluation

```
# MA
# This script is for evaluating the worker detection model
import csv
import glob
import xml.etree.ElementTree as ET
from shapely.geometry import Polygon
import os
import cv2

def calculate_iou(box_1, box_2):
    poly_1 = Polygon(box_1)
    poly_2 = Polygon(box_2)
    iou = poly_1.intersection(poly_2).area / poly_1.union(poly_2).area
    return iou


worker_properties_test = {}
worker_properties_detection = {}
with open('Eval_thesis/Eval4_15/Person_Detection_Coordniates_thesis_Eval_4_15.csv', 'r') as
file:
    reader = csv.reader(file, delimiter=",")
    next(reader)
    for i, row in enumerate(reader):
        x_min = row[1]
        x_max = row[2]
        y_min = row[3]
        y_max = row[4]
        # frame_NO = row[0][23:].split(".")[0]
        frame_NO = row[0].split(".")[0][7:]  # [7:]#[23:] #[0]
        worker_properties_detection[i] = {'frame_number': int(frame_NO), 'x_min': int(x_min),
                                          'y_min': int(y_min),
                                          'x_max': int(x_max), 'y_max': int(y_max)}

with open('Eval_thesis/Eval4_25/GT_labels.csv', 'r') as file:
    reader = csv.reader(file, delimiter=",")
    next(reader)
```

```
    for i, row in enumerate(reader):
        x_min = row[4]
        x_max = row[6]
        y_min = row[5]
        y_max = row[7]
        frame_NO = row[0].split(".")[0][3:]
        worker_properties_test[i] = {'frame_name': int(frame_NO), 'x_min': int(x_min), 'y_min': int(y_min),  # [3:]
                                     'x_max': int(x_max), 'y_max': int(y_max)}
TP = 0
FP = 0
FN = 0
iou_TP = 0

# Calculating the FP based on height of bounding box
for no, detection in worker_properties_detection.items():
    if detection['y_min'] >= 270:
        if (abs(detection['y_max'] - detection['y_min']) < 50):
            FP += 1
# Calculating TP in the ground truth for getting accuracy in each field
m = 0
for k, t in worker_properties_test.items():
    if t['y_max'] <= 540:  #Far_field (J)
    #if t['y_max'] <= 810 and t['y_min']>=270: # Mid_field (K)
    #if t['y_min'] >= 540:  # Near_field (I)
        m += 1

print('Field_GT', m)

# Overall validation
for key, test in worker_properties_test.items():
    for no, detection in worker_properties_detection.items():
        # print(test)
        # print(detection)

        if test['frame_name'] == detection["frame_number"]:
```

```
            iou = calculate_iou(
                [[test['x_min'], test['y_min']], [test['x_max'], test['y_min']],
                 [test['x_max'], test['y_max']],
                 [test['x_min'], test['y_max']]],
                [[detection['x_min'], detection['y_min']], [detection['x_max'], detection['y_min']],
                 [detection['x_max'], detection['y_max']],
                 [detection['x_min'], detection['y_max']]]
            )

            if iou > 0.5: # Overall full frame TP
            #if iou >0.5 and test['y_max'] <= 540:   # Far_field TP (J)
            # if iou >0.5 and test['y_max'] <= 810 and test['y_min']>=270: # Mid_field TP (K)
            #if iou > 0.5 and test['y_min'] >= 540:  # Near-field TP (I)
                iou_TP += 1


print("total_test", key + 1)
print("total_detection", no + 1)

print('TP: ', iou_TP)

print("FN", (key + 1) - iou_TP)
print("FP", FP)

print("precision", round((iou_TP / (iou_TP + FP) * 100), 2))
print("recall", round((iou_TP / (iou_TP + ((key + 1) - iou_TP)) * 100), 2))
print("accuracy", round((iou_TP / (iou_TP + ((key + 1) - iou_TP) + FP) * 100), 2))
```

Appendix F. Python Code for PPE Detection Evaluation

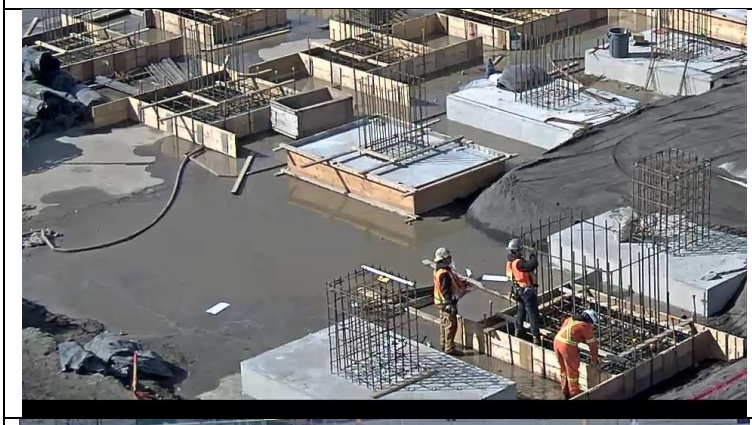
```
# MA
# This script is for evaluating the PPE detection model
import csv
import glob
import xml.etree.ElementTree as ET
from shapely.geometry import Polygon
import os
import cv2

def calculate_iou(box_1, box_2):
    poly_1 = Polygon(box_1)
    poly_2 = Polygon(box_2)
    iou = poly_1.intersection(poly_2).area / poly_1.union(poly_2).area
    return iou

PPE_test = {}
PPE_detection = {}
with open('PPE_Detection_Coordniates_thesis_Eval_4_15.csv', 'r') as file:
    reader = csv.reader(file, delimiter=",")
    next(reader)
    for i, row in enumerate(reader):
        x_min = row[1]
        x_max = row[2]
        y_min = row[3]
        y_max = row[4]
        # frame_NO = row[0][23:].split(".")[0]
        frame_NO = row[0].split(".")[0][7:]  # [7:]#[23:] #[0]
        PPE_detection[i] = {'frame_number': int(frame_NO), 'x_min': int(x_min),
                            'y_min': int(y_min),
                            'x_max': int(x_max), 'y_max': int(y_max)}

with open('PPE_GT_labels.csv', 'r') as file:
    reader = csv.reader(file, delimiter=",")
    next(reader)
```

```
    for i, row in enumerate(reader):
        x_min = row[4]
        x_max = row[6]
        y_min = row[5]
        y_max = row[7]
        frame_NO = row[0].split(".")[0][3:]
        PPE_test[i] = {'frame_name': int(frame_NO), 'x_min': int(x_min), 'y_min': int(y_min),  #
[3:]
                                'x_max': int(x_max), 'y_max': int(y_max)}
TP = 0
FP = 0
FN = 0
iou_TP = 0

# Calculating the FP based on height of bounding box
for no, detection in PPE_detection.items():
    if detection['y_min'] >= 270:
        if (abs(detection['y_max'] - detection['y_min']) < 50):
            FP += 1



# Overall validation
for key, test in PPE_test.items():
    for no, detection in PPE_detection.items():
        # print(test)
        # print(detection)

        if test['frame_name'] == detection["frame_number"]:

            iou = calculate_iou(
                [[test['x_min'], test['y_min']], [test['x_max'], test['y_min']],
                 [test['x_max'], test['y_max']],
                 [test['x_min'], test['y_max']]],
                [[detection['x_min'], detection['y_min']], [detection['x_max'], detection['y_min']],
                 [detection['x_max'], detection['y_max']],
                 [detection['x_min'], detection['y_max']]]
```

```
        )

        if iou > 0.5:

            iou_TP += 1

print("total_test", key + 1)
print("total_detection", no + 1)

print('TP: ', iou_TP)

print("FN", (key + 1) - iou_TP)
print("FP", FP)

print("precision", round((iou_TP / (iou_TP + FP) * 100), 2))
print("recall", round((iou_TP / (iou_TP + ((key + 1) - iou_TP)) * 100), 2))
print("accuracy", round((iou_TP / (iou_TP + ((key + 1) - iou_TP) + FP) * 100), 2))
```

Appendix G. Examples of Worker and PPE Training Datasets

Worker detection training dataset



PPE detection training dataset

Positive samples

Negative samples