Top View Deep Learning Object Detection using Active Perception in Construction Environment

Thesis approved by the Department of Computer Science University of Kaiserslautern-Landau for the award of the Doctoral Degree Doctor of Engineering (Dr.-Ing.)

by

Tanittha Sutjaritvorakul

Date of the defense	:	6 March 2024
Dean	:	Prof. Dr. Christoph Garth
Reviewer	:	Prof. Dr. Karsten Berns
Reviewer	:	Prof. Dr. Paul Lukowicz

Acknowledgement

Throughout the writing of this dissertation I have received a great deal of support and assistance. I would first like to thank my supervisor, Prof. Dr. Karsten Berns, whose expertise was invaluable in formulating the research questions and methodology.

I would like to acknowledge Asst. Prof. Dr. Kittiphan Techakittiroj, Dr. Kong Kritayakirana, and Dr. Panupol Lerssrisuriya for their invaluable advice. It is their kind help and support that have made my study and life smooth.

I would like to extend my sincere thanks to the following RRLab colleagues and students for their technical support on my study including assistance at every stage of the research project—Alexander Köpper, Atabak Nezhadfard, Axel Vierling, Behnoosh Sepehrian, Hannan Ejaz Keen, Jakub Pawlak, Massimo Tosa, Patrick Fleischmann, Patrick Wolf, Qazi Hamzi Jan, Qi Liu, Salah Al-Darraji, Sarwar Hussain Paplu, Songlin Piao, and Zuhair Zafar.

I am thankful to Mrs. Rita Broschart and Ms. Sabine Owens for their administrative support. I would also like to thank the technician, Sascha Steffens and RRLab admin team for their technical help. I want to acknowledge Mahmoud FarshbafDoustar to assist me and fill my gaps of knowledge.

Most importantly, I would like to take the chance to thank Alexandre Hauber da Silva, Carolina Nogueira, Felipe Salgado, Leandro Avila da Silva, Natdanai Promchinavongs, Nelson Junqueira de Andrade, and Willian Doppelreiter for all time support. My life in Germany would not be smooth without them. Finally, my appreciation also goes out to my family and friends for their encouragement and support all through my studies.

> Kaiserslautern, November 2022 Tanittha Sutjaritvorakul

Abstract

To increase situational awareness of the crane operator, the aim of this thesis is to develop a vision-based deep learning object detection from crane load-view using an adaptive perception in the construction area. Conventional worker detection methods are based on simple shape or color features from the worker's appearances. Nonetheless, these methods can fail to recognize the workers who do not wear the protective gears. To find out an image representation of the object from the top view manually or handcrafted feature is crucial. We, therefore, employed deep learning methods to automatically learn those features.

To yield optimal results, deep learning methods require mass amount of data. Due to the data deficit especially in the construction domain, we developed the photorealistic world to create the data in addition to our samples collected from the real construction area. The simulated platform does not benefit only from diverse data types, but also concurrent research development which speeds up the pipeline at a low cost. Our research findings indicate that the combination of synthetic and real training samples improved the state-of-the-art detector. In line with previous studies to bridge the gap between synthetic and real data, the results of preprocessed synthetic images are substantially better than using the raw data by approximately 10%.

Finding the right deep learning model for load-view detection is challenging. By investigating our training data, it becomes evident that the majority of bounding box sizes are very small with a complex background. In addition, we gave the priority to speed over accuracy based on the construction safety criteria. Finally, RetinaNet is chosen out of the three primary object detection models.

Nevertheless, the data-driven detection algorithm can fail to handle scale invariance, especially for detectors whose input size is changed in an extremely wide range. The adaptive zoom feature can enhance the quality of the worker detection. To avoid further data gathering and extensive retraining, the proposed automatic zoom method of the load-view crane camera supports the deep learning algorithm, specifically in the high scale variant problem. The finite state machine is employed for control strategies to adapt the zoom level to cope not only with inconsistent detection but also abrupt camera movement during lifting operation. Consequently, the detector is able to detect a small size object by smooth continuous zoom control without additional training. The adaptive zoom control not only enhances the performance of the top-view object detection but also reduces the interaction of the crane operator with camera system, reducing the risk of fatality during load lifting operation.

Contents

1	Introduction		
	1.1	Overview of Construction	1
	1.2	Construction Automation and Robotics	2
	1.3	Safety in Construction	3
	1.4	Usage of Deep Learning Methods	4
	1.5	Aims and Objectives	5
	1.6	Outline	7
2	Aut	comated Vision-Based Top View Safety Monitoring in Construction	
	Are	a	9
	2.1	Conventional Top View Object Detection	12
		2.1.1 Conventional Vision Techniques	12
		2.1.2 Top View Applications of Traditional Approach	17
	2.2	Deep Learning-Based Top View Object Detection	19
		2.2.1 Deep Learning-Based Detection Techniques	19
		2.2.2 Top View Applications of Deep Learning Approach	20
3	The	esis Overview	25
	3.1	Challenges of Top View Object Detection	25
4	A	husis and Commetium of Deal and Completie Date	กก
4		Crops Operation and Sofety	33 24
	4.1	Crane Operation and Safety 4.1.1 Construction Sofety	04 25
	19	Simulation Usage in Construction	- 30 - 27
	4.2	4.2.1 Simulation Distforms	30 20
	13	A.2.1 Simulation Flation ins	30 //1
	4.0	4.3.1 Cap Botwoon Synthetic and Boal Data	41
	4.4	Finite and Sconario	40
	4.5	Crane Setup for Beal Dataset Concration	45
	4.0 1.6	From Real World to Simulation	40
	4.0	4.6.1 Implementation Concept	40
		4.6.2 Virtual Environment and Scenario	<u>+</u> 0
		4.6.3 Virtual Hardward	чэ 51
	17	Data Collection From Real Environment	51 52
	4.1	471 Data Recording	52 52
		± 1.1 Dava moonuling \ldots \ldots \ldots \ldots \ldots \ldots \ldots	04

		4.7.2 Data Annotation	55
		4.7.3 Public Datasets	55
	4.8	Synthetic Data Generation	56
		4.8.1 Data Recording	56
		4.8.2 Data Annotation	57
	4.9	Dataset Summary	58
	4.10	Evaluation Metrics for Object Detection Algorithms	59
		4.10.1 Average Precision (AP) \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots	62
	4.11	Analysis of Real and Synthetic Data in Frequency Domain	63
	4.12	Experiments	67
		$4.12.1\;$ Exploration Aerial Image DL model With Crane Load-View Data $\;$.	67
		4.12.2 Investigation of Synthetic Data Replacement in Training a NN Model	l 70
	4.13	Discussion	73
5	Тор	View Object Detection Using Deep Learning	75
	5.1	Deep Learning-Based Detection Techniques	75
	5.2	Data Inspection	78
	5.3	Candidate List of Network Models	80
	5.4	Network Configuration and Selection	83
		5.4.1 Network Configuration	84
		5.4.2 Network Selection $\ldots \ldots \ldots$	85
		5.4.2.1 Configuring the Trainer	86
		5.4.3 Result of Network Selection	88
	5.5	Discussion	90
6	Ana	lysis of Motorized Zoom Camera	91
	6.1	Adaptive Zoom Camera and Its Applications	91
	6.2	Implementation of Zoom Control	93
	6.3	Worker Localization in Camera Coordinate	95
		6.3.1 Camera Projection	96
	6.4	Proposed Approach	98
	6.5	System Architecture	100
	6.6	Zoom Controller	101
	6.7	Zoom Controller Verification	104
7	Con	nprehensive Experiment	107
	7.1	Final Network Model Test	109
		7.1.1 Results	111
	7.2	Top View Worker Detection Using Adaptive Zoom	119
		7.2.1 Result	119
	7.3	Discussion	121
8	Con	clusion and Outlook	123
	8.1	Outlooks	125
\mathbf{A}	Acr	onym	127

В	Building Simulation Environment			131						
	B.1 Crane and Sensor									
	B.2	Connee	etion to Finroc	133						
	B.3	Humar	Characters	134						
	B.4	Enviro	nment	135						
\mathbf{C}	Network Configuration 1									
	C.1	FasterI	$\operatorname{RCNN-Resnet50} \dots \dots$	139						
		C.1.1	Model configuration	139						
		C.1.2	Training Configuration	141						
		C.1.3	Evaluation Configuration	141						
	C.2	FasterI	RCNN-Resnet101	142						
		C.2.1	Model configuration	142						
		C.2.2	Training Configuration	143						
		C.2.3	Evaluation Configuration	144						
	C.3 RetinaNet-Resnet 50									
		C.3.1	Model configuration	144						
		C.3.2	Training Configuration	147						
		C.3.3	Evaluation Configuration	147						
	C.4	Input (Configuration	148						
	C.5	Label I	Map	148						
D	Zoo	m Con	troller	149						
\mathbf{E}	E Magnitude Spectrum									
Bi	Bibliography									
In	Index									

1. Introduction

Recently, digitalization using Machine Learning (ML), particularly Deep Learning (DL) methods has been embraced by construction industry. In comparison to other industry sectors such as the automotive industry, entertainment, healthcare, the construction domain is relatively slow in adopting these techniques [1]. The scope of research activity in construction is very broad, ranging from robotics to project planning using Building Information Modeling (BIM). DL approach is used in many applications, specifically as safety assistance or sensory systems for construction automation and robotics. Applying these technologies in construction can increase productivity as well as safety.

In this thesis, we focus on developing a crane operator crane safety assistance system. The adaptability of the load-view zoom camera can improve the safety of operation and help the crane operator to focus on load handling rather than manually adjusting the zoom level of the camera. Indeed, this thesis addresses the challenges in top-view pedestrian detection and introduces the active zoom control to improve the performance of DL networks.

1.1 Overview of Construction

The construction sector plays an important role in the economy. This sector in 2002 contributed approximately 9% and 8% to the gross domestic product (GDP) of the European Union (EU) and United States (US), respectively. As a result, it employs over 18 and 7 million workers in the EU and US [2, 3]. The construction sector plays a vital role in all parts of economic infrastructure such as factories, buildings, offices, schools, houses, hospitals, and roads. They are all byproducts of the construction.

In addition to the lack of resources, labor participation in this industry decreases annually due to the great number of accidents. In the EU, more than 20% of all work fatalities appeared in the construction division in 2015 [4]. Similarly, the Occupational Safety and Health Administration (OSHA) reported 19.5% of fatal occupational accidents in the US from 2002 to 2012 were from the construction sector. Construction is a labor-driven industry. It is a field-based, changeable and project-oriented industry. The workers experience a wide range of hazards in daily work which lead to injury. Some examples that show construction as a dangerous job are working with power tools, operating heavy



Figure 1.1: Example of construction robots. (a) SPOT robot observing construction site. [5]© IEEE 2020. (b) Safety assistance on a crane. (c) Bomag Tandem roller for autonomous road compaction and robot collaboration. [6] (d) HRP-5P, a humanoid construction robot. [7]© IEEE 2019.

equipment, constructing sewer/duct, demolition, and high-rise construction. The workers have to cope with risks in every perspective such as airborne diseases from toxic mixtures, unintended collapse, hit by objects, hand-arm vibration syndrome (blue finger) from continuous use of vibratory tools and electrocutions. The causes of accidents on the construction site can occur in any phase of the construction. Small failure can set off the chain reaction to endanger the rest construction phase. With the low construction productivity and the high number of accident problems, *construction automation and robotics* have been introduced to bridge the gap.

1.2 Construction Automation and Robotics

Construction automation and robotics were primarily used to increase productivity and efficiency. They are introduced to eliminate human error while increasing production and optimizing energy consumption. The *construction automation* is considered as the technology which helps the construction process with minimal human intervention. In other words, construction automation is labor-saving technology. The application of construction automation ranges from boiler control of household thermostats to heavy machine control systems. On the other hand, the *construction robotics* is one of the growing fields in construction automation. It is considered as a research extension of field service robots that work in unstructured and harsh environments. The construction robots are generally designed to assist humans or have no human supervision.

Construction robotics was initially started in Japan at the beginning of the 1970s. At that time, Japan mainly experienced the worker's shortage. The automated machines and robots were made to perform an individual task so-called single-task construction robots[8]. The task of processing building material production like handling wall panels or spraying exterior walls was founded in many companies - Kajima, Shimizu, etc [9]. Outdoor construction robotic primarily adopted technology from factory-based manufacturing or aerospace research[10].

The construction robots can range from small manipulators to large-scale construction robots like tower cranes. Since 2002, humanoid robots have been involved in various construction applications. The design of the construction robots is very diverse in different sizes. They can be bipedal, quadruped, vehicle, etc. For example, HRP-5P in Fig. 1.1d is a bipedal robot which is capable of installing drywall [7] or SPOT in Fig. 1.1a, a quadruped robot, is able to record and track construction progress or damage in remote operation with autonomous sensing.

The research activities exist in every element of the construction life cycle i.e., planning and design, construction robotic, intelligent job-site management, operation and maintenance, and others [11]. The robotic technology is not limited to conventional robot manipulators but it can include automated construction with the advancement of digitalization. The capabilities of construction robotics are not framed only by performing an individual simple repetitive task but also include teleoperation, sensory data collection, numerically control, and performing the task autonomously while collaborating with the workers.

1.3 Safety in Construction

Safety is one of the most important aspect in construction. So far not all of the construction works can be replaced by automated machines or robots. In general, the decision-making and reliability in the robots are still very far away from the human. Especially in vehicle construction machine, it rarely finds fully automation especially for the complicated tasks e.g., lifting, in construction starts to have only semi-autonomous which is an assistance system or remote operation.

As automation and robotic technologies are gradually brought into the construction industry, robots and humans start to work side-by-side. Consequently, the construction robots should be functional with harmless interaction in the workplace. In the safety domain, the research also exploits robot technology to assist the machine operator or to observe worker's ergonomic. Thus, the goal is to reduce the number of accidents in addition to increasing productivity.

Falling load or struck-by load are the most common and most dangerous crane-related hazards. The workers can be struck or hit by any moving load while they are working in close proximity to the crane. Beavers et al. [12] reported that 32% of the crane fatalities caused in 1997-2003 are struck-by load. The victims were mainly workers, who were not involved with crane [13]. The human factor is the main root cause of accidents besides the environment, equipment, etc. Operating the crane is a difficult task. Unlike in the



Figure 1.2: Example of deep learning object detection from an unmanned vechicle. This figure is modified from [19][©] IEEE 2018.

street environment, the construction site is complex and unstructured. Lifting operations is a repetitive and exhaustive task. Underestimating the hazardous situation is the most frequent cause of the struck-by accident [14]. The operator may not be able to see any workers who are underneath the load or stay in a swing radius or falling zone. Several of the human casual factors in crane accidents are inadequate training, miscommunication, exhaustiveness, etc [15]. Moreover, a construction worker may work more than five distinct construction sites in a year [16]. The worker may not be familiar with the different construction sites e.g., environments, colleagues (communication), and equipment (cranes).

The crane operators are prone to human errors as they misjudge how risky the current situation will be (or to identify hazards). The operator simultaneously concentrates on many elements while operating e.g., load, controlling the crane, load chart, and wind speed. To reduce such human error, *Situational Awareness (SA)* should be provided. In other words, the operator should actively perceive what is going on around during operation or identify potential hazards and properly respond [17]. Having a good SA is the foundation of good decision making and therefore the fewer failures or errors.

Robotics start to become a part of the construction industry. In the same way (as a human), fatal accidents can happen in robots such as collisions, crushing, and injuries from mechanical parts. In spite of no specific OSHA standards for the robotics industry, construction robotics must have safety features while operating in the environment with human [18]. The robot itself should be able to be aware of the surroundings same as the operator to avoid any accident.

1.4 Usage of Deep Learning Methods

Despite the recent deep learning trend in autonomous driving, the self-driving car, called an Autonomous Land Vehicle In A Neural Network (ALVINN), was created in 1989 [20]. At the same time, the deep learning method also existed in other applications. For instance, handwriting digit recognition is proposed using the back-propagation network of LeCun et al. [21]. Due to the immaturity of internet technology, the deep learning approaches, which require a lot of training data, could not be fully exploited. Another reason was that the hardware technology was yet very expensive and hardware architectures or GPUs that supported the computation of Convolutional Neural Network (CNN) and accelerated the long training process, seldomly existed.

Since the information age, which began in the mid-20th century, has arrived. The sensor technology or powerful hardware is getting cheaper and cheaper. The data resource becomes much easier to access such as Google Images and Gettyimages. As a result, DL methods nowadays become more recognized and have been widely adopted in many research areas e.g., self-driving cars, Unmanned Aerial Vehicle (UAV) (see Fig. 1.2), Human-robot Interaction (HRI).

With the outstanding results of the DL methods, the construction domain also employs these data-hungry learning methods for worker recognition. In spite of the flood of big data, a simulation platform or a game engine platform is taken as data augmentation tools in order to generate more data feeding these learning algorithms. The simulation provides perfect annotations for perception learning tasks including as an experimental platform. There are three primary situations, where the simulation becomes extremely helpful [22]. First, the data is difficult to obtain from the source such as a construction worker catching an ostrich in the middle of a construction area. Second, the data is difficult to label such as to annotate workers from a bird's-eye view or each individual object in a densely crowded situation, see Fig. 1.3a. Lastly, the closed-loop and repetitive behavior have to be performed specifically if the data depends on such actions.

1.5 Aims and Objectives

(Semi-)Automated safety assistance system has been widely adopted in the field of construction domain. For instance, worker detection helps the operator who simultaneously works on many tasks, to be aware of workers nearby the machine. To prevent struck-by-load accidents, observation from the top view should be carried out. The top view offers a wide and advantageous perspective, where the observer stays higher than the observed objects. The top view, which can be perceived from a standard crane zoom camera, basically supports the crane operator during the blindlift. By feeding the top-view camera streaming to worker detection, the location of the worker-on-foot in the proximal distance can be shown to the operator. Although there has been much similar research on aerial image detection from UAV, there has been no study focusing on load-view object detection.

The ultimate goal in this thesis is to develop a vision-based worker detection from top view including zoom active control using the deep learning method. Most of the conventional detection methods are highly dependent on target appearances or background environment. As a result, they usually fail in sophisticated situations like construction areas where it is dynamically reconstructed through each construction phase. Using the deep learning method, the object detection is able to discover the hidden feature patterns in the image which is difficult for a human to find out such as detecting workers from the top view whose size is very small. By showing the result of detection, it can further raise the situational awareness of the operator to reduce struck-by load accidents. However, detecting objects from aerial view remains challenging. Despite remarkable results of data-driven detection methods, the highest accuracy presented in the statistic of top view detection VisDrone benchmarks is merely 30 percent [25].



Figure 1.3: Comparison of top view on different heights. (a) Example of building detection on satellite image at an altitude of higher than 500 km. [23]© IEEE 2018. (b) UAV the range of most commercial drones can fly between 0.5 and 1.5 km depending on the weight and restricted areas such as airports, federal highways, and railways. [24]© IEEE 2019. (c) Example of the view from a mobile crane. (d) Example of building with the height of 22 meters.

The scope of the work mainly studies the situation when the worker-on-foot comes nearby the load while a mobile crane performs lifting. The safety monitoring can be conducted when the work zone can be seen within the camera view and excluded by load obscurity. To develop the top view deep learning worker detection using adaptive zoom, the objectives of this research are the following.

• To generate a substantial amount of data including investigation of the synthetic data for training of deep learning networks:

By employing DL approaches, a large dataset has to be generated to reach the full potential of the DL approach. One of the major objectives of this work was to create a simulation platform especially for the construction area in addition to real-world data collection. The simulation toolchain can be easily adjusted to the new environment in future work. The experiment should be conducted to check whether the synthetic data can either replace or augment the real-world data. To train the DL models, the dataset will be generated in a significant amount. The analysis of training data should be studied to have a better understanding of the data characteristics affecting the training process.

- To choose the competent deep learning network for the load-view detection: The main objective is to determine suitable deep learning methods for worker detection using a load-view camera. Detecting objects from top view can perform in various methods based on different altitudes as the amount of the information and the appearance of the object is completely non-identical in each level as depicted in Fig. 1.3. Several state-of-the-art network models which were commonly adopted in the aerial image object detection e.g., Faster Region Based Convolutional Neural Networks (R-CNN), Single Shot MultiBox Detector (SSD), and RetinaNet, should be explored. In addition, the choices of network parameters that are suitable to the load-view input data should be examined. The time performance of the chosen network should be appropriate for safety measurement. In other words, the detected output should present to the operator within safety response time in order to allow the operator to respond and handle the incoming risk such as halting the lifting operation.
- To preliminary analyze a capability of the adaptive perception:
- Recognizing an object from the top view is very difficult, even for a human because the object size is very small and the perspective is completely changed relative to the frontal view. In detecting workers from the load-view camera, most detection methods could fail due to a big range of scale changing from either crane boom movement or zooming camera despite scale invariance characteristic. The top view DL-based detection indeed struggles from extracting the feature out of the small number of pixels. Thus, the objective is to devise and implement an adaptive perception via zoom crane camera for increasing the quality of the chosen DL detector. To utilize the zoom mechanism of the load-view camera, it is necessary to examine the sensor properties in particular the zoom control function. Instead of raw sensor data, the step of zoom control verification should be carried out using a reliable and accurate reference target such as a fiducial marker.

1.6 Outline

To address the safety awareness of the crane operator, the purpose of this thesis is to implement a robust top view DL-based object detection via an active perception in a construction environment. The worker detection is a part of an operator assistance perception system which can increase situational awareness of the crane operator. The crane is a heavy equipment vehicle. Small mistakes can lead to huge consequences such as fatal accidents.

In this chapter, we briefly provided the importance of construction and accident statistics. To reduce accidents and increase productivity in construction, automation and robotics including safety are introduced. The aim and objectives are then discussed to set out what we expect to achieve and steps corresponding to each goal.

In Chapter 2, the previous studies of automated vision-based top view safety monitoring in construction areas are discussed into three main categories based on risk identification. Different types of image sensors, which are typically employed in safety monitoring, are listed. Later, we presented the literature review of the top view object detection and its development. The important concept and the limitations of the existing research were pointed out. The contributions behind this thesis have been introduced in Chapter 3. The significance and advantages of this research including the scope of work are indicated.

Chapter 4 demonstrates the usage of the synthetic data to enhance the quality of the load-view worker detection using the deep learning approach. The construction safety and the typical crane operation that potentially causes the struck-by load accident is described. The application of the simulation platform in the previous studies such as dataset generation is elaborated. Afterward, we showed the development of our simulation platform and the real crane setup of the test environment where the experiment of this thesis took place. After the great amount of data was collected from both real and virtual platforms, two experiments were conducted at the end of the chapter to validate the usage of synthetic data for top view worker detection.

Chapter 5 presents the finding of the suitable deep learning network model for the load-view worker detection. Several most primarily deep learning object detection models in the literature were first discussed. Based on the selection criteria and the analysis of our collected dataset, the chapter contains the experiment for choosing the final network model that fits our application from the candidate list.

The analysis of the crane load-view zoom camera is presented in Chapter 6. The zoom control mechanism of this off-the-shelf image sensor is employed to support the top view worker detection using deep learning. Without zoom function, the detector suffers from extracting the image feature due to the very small size of the bounding box. In this chapter, the proposed zoom controller logic is demonstrated including the zoom verification using the fiducial marker.

Eventually, Chapter 7 integrates all proposed components from the previous chapters as a comprehensive experiment. The content in the last chapter, Chapter 8, comprises the conclusion and follows by a future work that we plan to work on.

The acronyms is declared in Appendix A. The making of simulation environment in Unreal Engine (UE) such as virtual characters and UE-FINROC connection is elaborated in Appendix B. The detail of network configurations used in the network selection procedure is given in Appendix C. Appendix D further expands the experimental results regarding zoom controller FSM from Chapter 6 and Chapter 7. Lastly, Appendix E additionally presents Magnitude Spectrum (MS) result for each dataset.

2. Automated Vision-Based Top View Safety Monitoring in Construction Area

Identifying the risk in the construction site is manually conducted by observers and highly depends on their perceptual capability. The observer including the operator and personnel involved require experiences to understand and evaluate the scenario by comparing to the safety checklist, guideline, or rules to spot potential hazards e.g., a working and load radius indicator with audible alarm is installed, the ground condition should be stable and ensuring only competent or trained personnel are allowed in the working area [26, 27]. Likewise, vision-based safety monitoring approaches should consider not only how to recognize construction entities in an image, but also how to apply expert knowledge regarding unsafe conditions to the perceived information.

There are numbers of image sensor which are adopted for vision-based perception of the safety monitoring in construction. For imagery data collection, camera(s) can be installed either on the vehicle or on-site. Camera sensors are used in research including a monocular or surveillance camera, stereo camera (e.g., Bumblebee XB3, FLIR Systems, Inc.), Pan-tilt-zoom camera (PTZ) camera, and motorized zoom camera, etc. Browatzki et al. [28] found that 3D-based and the combination of 2D and 3D-based approaches provide higher classification accuracy than 2D-based. For construction indoor tasks such as finishing work (e.g., facing, plastering, flooring, painting, wallpapering, and glazing), KinectTM RGB-D camera is used to sense the motion and posture of workers. The raw perceptual information is later analyzed for the proper ergonomic properties [29, 30]. Unlike stereo vision, the 2D camera is inexpensive. The monocular 2D imaging system is typically used in construction. More specification comparison of imagery devices—Light Detection and Ranging (LiDAR), flash LiDAR, monocular and stereo vision cameras, can be found in [31].

For the vision-based safety monitoring system, workers and construction equipment are the targets in the images. The target is localized by different computer vision techniques. The recognized target can be further interpreted and identified the risk in different dimensions—scene, location, and action according to Seo et al. [32].



Figure 2.1: Comparison among object recognition techniques based on risk identification criteria. From the left, object detector identifies the location of workers-on-foot nearby the mobile crane. Object tracker in the center figure estimates both location and trajectory of the objects which also gives a potential direction where the worker is working to move next. The activity recognition in the right figure provide the pose of the objects that can be further intepreted if the safety is violated such a worker is running toward the crane.

First, *scene*-based risk identification refers to the understanding and estimating any likelihood of harm in a static scene by observing the overall site condition in the safety context such as high noise or congested area [33, 34], Personal Protective Equipment (PPE) non-compliance [35, 36]. The scene-based risk; therefore, can be determined by the number of construction resources, tools, or safety equipment (e.g., hard helmets, safety vest) presented. *Object detection* method can be used to identify the object's location in an image scene by detecting its features e.g., shape, texture, and color. *Object classification*, which part of object detection's pipeline, later classifies the detected objects into different categories.

Second, *location*-based risk identification points out the hazard based on the location and movement of the construction resources such as proximity warning for the struck-by accident, anti-collision, motion planning, and speed violation [37, 31, 38, 36, 34, 39, 40]. *Object tracking* predicts and estimates the location and trajectory of one or more objects over time by distinguishing image features, motion, etc. Each tracking target has its own identification. Object tracking can be initialized automatically by the output of the detector or manually by the user. In addition, object tracking does not only provide temporal information but also improves the performance of object detection [41]. Tracking stabilizes the accuracy and precision of the detector by linking noisy targets on a new image frame based on previously tracking targets.

Third, *action*-based risk identification aims on detecting unsafe activity between worker and job conditions. Working in awkward positions can lead the worker to the risk of developing a work-related musculoskeletal disorder (WMSD) [42]. *Activity recognition* techniques mainly focus on injury prevention i.e., what action between worker and equipment is, and evaluate if it is an ergonomic violation such as improper lifting [43] or unsafe during ladder climbing [30]. The techniques make sequentially observations on the detected targets' actions in the environment. In particular, most activity recognition methods attempt to extract the skeleton data of the human or parts of the vehicle, like joints, and determine



Figure 2.2: An illustration of blind spot situation. From the crane cabin, the operator is unable to notice the workers, who are in red BBox due to the obstruction. As a result, the workers can be hit by the load.

the target pose. The pose is subsequently converted into action. In addition, the location and action-based approach can be used for performance and productivity analysis [44, 45].

Fig. 2.1 illustrates the difference of computer vision techniques mentioned above. Detecting objects are the basic categories of vision-based safety monitoring. Tracking and Action recognition is beyond the scope of this thesis. A complete survey of the topic tracking and action-based approaches can be found in [46, 47].

Top view vision-based object detection plays a crucial role in the research field. It has been widely used in many domains such as construction, traffic, sport, social etiquette analysis [48], search and rescue. Detecting object from the top provides overall information instead of a specific one. The goal of such application e.g., safety monitoring and crowd analysis requires to perceive merely the existence of the objects as many as possible in the large viewpoint. Henceforth, it serves the purpose of the applications. In this work, the literature review is mainly provided in the direction of using computer vision techniques in construction sites for safety monitoring or crane operator safety assistance.

Monitoring the safety of the construction site from above can be done by using the surveillance camera mounted on the site or the load-view camera on the crane. Detecting workers from the load-view is challenging. In general, it is very hard for a human to notice the small-sized workers from the top view, see top right of Fig. 2.2 which explains the high fatality rate in crane operation and necessitates the application of load-view worker detection. The construction area is cluttered and dynamically changing over time.



Figure 2.3: Comparison between conventional detection and deep learning methods. The feature extraction in the traditional object classification method (1st row) is essentially hand-crafted, while generating the features in the deep learning method (2nd row) can be obtained from end-to-end learning where human does not need to define how the feature looks like for the method the learn.

There are not much research has been conducted on this topic. In the standard visibility assistance in the crane, the load-view camera feed gives non-semantic information e.g., no identification of the object's position surrounding the crane. In contrast to the frontal view, the load view is advantageous for safety observation as it provides an object viewpoint without occlusion. It is crucial for an operator to understand or be aware of what happens inside the working zone without manual inspection. Despite the fact that the operator perceives a lot of incoming information, such as lift trajectory and wind speed during lifting, the operator has to observe objects surrounding the large load (e.g., a container) from merely a camera stream of the 7-inch standard monitor.

In general, top view object detection techniques in literature can be categorized into two groups as depicted in Fig. 2.3. The following section will first introduce conventional object detection which includes techniques and their applications which were used in the top-view safety monitoring. The basic top view object detection pipeline will break down into blocks and will be discussed in detail. Later, the top view object detection using deep learning approaches are presented in Sec. 2.2. The development of well-known deep learning network architectures are introduced together with the previous studies.

2.1 Conventional Top View Object Detection

2.1.1 Conventional Vision Techniques

Traditional top view object detection pipeline consists of three main steps i.e., candidate generation, feature selection or extraction and classification, see Fig. 2.4.



Figure 2.4: Object detection pipeline.

Candidate generation: Any different objects may randomly appear in any location in the image with different sizes. To find out the potential regions, the intuitive method is scanning through the image completely with multi-scale *sliding window*. a sliding window is a rectangular region of predefined width and height that strides across an image.

Teutsch and Krüger [49] used the sliding windows to generate targets, which are moving vehicles from aerial image frames. After sliding window, Non-Maximum Suppression (NMS) is commonly applied as the post process to select the most appropriate BBoxes. In other words, NMS suppresses the smaller overlapping BBoxes and outputs only the larger ones as shown in Fig. 2.5a. Unlike person detection which rescales the sliding window into 50 different scales, the authors introduced three different sliding window scales for the vehicles based on typical vehicle size. They were small scale (buses and trucks whose length are between 15-20 m), mid scale (large cars, small trucks), and original scale (standard cars whose length are between 4-5 m). As the result, the search space of sliding window became much more less. The authors finally demonstrated that the detection using this exhaustive candidate generation. Nevertheless, the *slide* detection window has to go over all scales and positions as the sliding window methods attempt to search for all possible candidates. This results expensive computation and causes region duplication [50, 51].

Another alternative candidate generation method is *background modeling*. The background subtraction results in the blob or rectangular foreground regions due to the change of motion between frames. Kim et al. [34] used an on-site camera to detect worker-on-foot and construction machines using background subtraction and estimate the risk of each entity on the field. The Gaussian mixer model (GMM) or Mixture of Gaussians (MoG) is used to distinguish the moving objects from the background. MoG models the background pixel values as a mixture of multiple Gaussian distributions. The authors [34] mentioned the background subtraction can increase the reliability of the safety monitoring. The performance of the method can be optimized by parameter tuning. Lingam et al. [52] improved the quality of foreground extraction for real time UAV tracking, shown in Fig. 2.5b and 2.5c. The authors addressed the noise and ghosting effect of traditional background subtraction technique, which can be occured due to dynamic environment from the aerial surveillance. They proposed an adaptive Gaussian-based background subtraction approach. The tuning factor is statistically optimized ranging from 0 to 1.

Despite the background subtraction is computationally efficient, there are several limitations in detecting construction resources. For instance, it is unable to detect static objects.



Figure 2.5: Different candidate generation methods. (a) Sliding window. This figure is modified from [49]© IEEE 2015. (b) and (c) MoG Background subtraction© Reprinted by permission from Springer Nature: Springer [52], Copyright (2014).

Additionally, the object type of the extracted foreground object is unable to be identified such as finding the difference between pedestrian and construction worker. Furthermore, it may consider several objects which partially overlap as a single object. The extensive literature related in background subtraction techniques can be found in [53].

Identifying a region can be beneficial to find multiple candidates which potentially exist in an image. This image processing, which partitions an image into multiple regions, is called *image segmentation*. The representation of the image becomes more meaningful and easier to analyze. Segmentation does not require making an assumption about the object size or shapes compared to an exhaustive search like the sliding window approach. Felzenszwalb and Huttenlocher [54] proposed an efficient image segmentation algorithm using a graph theory. An image is represented by a undirected graph G = (V, E). A pixel is defined as a node $v_i \in V$, while an edge $e = (v_i, v_j) \in E$ links between vertices (v_i, v_j) . The pixel difference or weight score $w(v_i, v_j)$ can be distinguished by pixel's intensity, color or location, etc. The more different the two pixels are, the higher the weight is. Eventually, segmentation is a partition of vertices into multiple linked components C. Similar pixels are assigned to the same components. Nevertheless, segmentation is difficult because sometimes it is unclear what separates an object. In other words, it is hard to define the pixel similarity. Selective search (SS) [55] is a region proposal method that offers better segmented candidates via a bottom-up hierarchical grouping. The method recursively combines similar regions into larger ones, see Fig. 2.6. This method is initially developed based on the image segmentation algorithm [54]. Tewari et al. [56] employed the selective search to generate targets for deep learning based classifier for detecting vehicles in aerial images, see Fig. 2.7. The authors indicated the simplicity to generalize object proposal to any vehicle aerial images by adjusting the parameters of the selective search such as initial segmentation size, minimum and maximum proposal size.

In general, candidate generation methods are used to define the object boundary for the feature extraction and is later classified.

Feature extraction: Given an image from top view perspective, a feature extraction algorithm generates a set of feature descriptors or feature vectors. An image feature is a point of interest for describing an image for detection. The top view images typically have less features to detect pedestrian since they usually return a horitzontal plane unlike frontal view that presents a wide range of features that can be used to detect human or



Figure 2.6: Selective Search, one of the candidate generation methods[©] Reprinted by permission from Springer Nature: Springer [55], Copyright (2013).

pedestrian. For example, the silhouette of a human from the frontal view is much more complex (rich feature) compared to the top view. The feature descriptor should carry sufficient information of the object in the image and should not have any unnecessary data, like an image noise from the extraction. Importantly, the descriptor should cost low computation to ease for a large collection of images and quick extraction. They should be invariant to clutter, illumination change, size, posture, etc. In other words, the descriptor should generalize the same object type, even from the different views, it should produce the identical feature as much as possible. The feature descriptor should contain well-organized information of the image for the detection tasks. Image feature extraction can be ranged from low level (e.g., color, motion, gradient) to mid-level (e.g., edge, regions, corners) and high level (e.g., background, object model) [57]. The image feature is considered to represent a target. Constructing handcraft visual features requires expertise in order to process the raw data into a suitable internal representation form. The features should be robust and semantic.

Image gradients or Gradient vectors is one of the most basic concepts in computer vision. It measures how the image is changing. The change is in pixel value along the x-direction and the y-direction around each pixel. The image gradient ∇I has two properties; (a) magnitude $|\nabla I(x,y)| = \sqrt{I_x^2 + I_y^2}$ is L2-norm. It shows how quickly the image is changing and (b) direction $\theta_{\nabla I} = \arctan(I_y/I_x)$ tells us the orientation in which image is changing. The advantages of the image gradients are invariance in brightness and scale. Given I(x,y) is a pixel color value at location (x, y), the image gradient computation can be defined as follows.

$$\nabla I = \begin{bmatrix} \nabla I_x \\ \nabla I_y \end{bmatrix} = \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{bmatrix} = \begin{bmatrix} I(x+1,y) - I(x-1,y) \\ I(x,y+1) - I(x,y-1) \end{bmatrix}$$
(2.1)

 $\frac{\partial I}{\partial x}$ and $\frac{\partial I}{\partial y}$ are the partial derivative on the x-direction and y-direction respectively. The well-known applications of image gradients are edge detection and feature extraction of Histogram of oriented gradients (HOG) descriptor [58]. Garcia et al. [59] exploited HOG features to localize person from aerial view to boost disaster search. The study showed that



Figure 2.7: Example of selective search from top view images from VEDAI dataset[60][©] Reprinted by permission from Springer Nature: Springer [56], Copyright (2019).

using HOG as image descriptor provide general representation of the object in different domains e.g., pose and angle of view. Consequently, this is suitable for the aerial view because the local feature such as face, which is invisible from the aerial view. In addition, HOG is tolerant to different brightness.

Another examples of the prominent visual features are Local Binary Pattern (LBP) [61], Scale-Invariant Feature Transform (SIFT) [62], Haar-like [63], etc. LBP feature is relatively similar to HOG which are built on gradient of a surrounding pixel. As the name implied, LBP extracts the local patterns from eight directions for each pixel, while the HOG only used one direction of the greatest magnitude for each pixel. Therefore LBP has better local representation. On the other hand, HOG outperforms in capturing edges and corners. Moranduzzo et al. [64] used LBP features to classify the agriculture region to support archaeological analysis from extremely high resolution (EHR) aerial images. The Chi-square distance is used as similarity measurement. Instead of pixel-based, the authors used the tile-based which turned out to be simpler and more efficient.

To track a single object from drone, Jabar et al. [65] employed SIFT feature for Kanade–Lucas–Tomasi (KLT) tracking method to overcome similar objects in the video frames and cluttered background. The authors demonstrated that using SIFT feature was able to provide sufficient information to represent the object shape. Although the generated visual features are associated based on how the human brain interprets, it remains difficult to have general features which can be used for any objects [62]. It is challenging to recognize objects in different appearance, scale, brightness, pose, or obscure, etc.

Classification: The classic classification appraches are used intensively in satellite imagary and drone vision, which will be elaborated later on after introducing common classification approaches. Classification is a machine learning process that generates a prediction model. The classifier model determines the class label of subregions in the image based on the feature representation. Once the visual feature of the candidate is generated, those features have to be categorized into a class. There are three main learning paradigms in machine learning. In this thesis, only supervised learning will be mainly discussed. The rest are beyond the scope of this work. *Supervised learning* is the most common form of machine learning such as Support Vector Machine (SVM) [66]. A large dataset of images is collected. Each object in an image is labeled by hand. The model learns from the given dataset including its label. While the *unsupervised learning* does not



Figure 2.8: Vision-based crane load monitoring. The upper row depicts the load monitoring simulation in indoor, while the bottom row illustrates load monitoring in the actual construction area. [70]

have any training dataset or has only an incompleted dataset, the model itself has to find the unknown patterns in the non-labeled dataset. Lastly, the *reinforcement learning* (RL)is the trial-error method where the agent learns how to achieve a goal on their own under an uncertain environment. The agent gets either rewards (positive) or penalties (negative) feedback based on the actions it performs. The goal is to maximize the reward. Neural Network (NN) can be used in all these three learning methods depends on the learning tasks.

In supervised learning, the machine is shown an image during training and produces an output in the form of a vector of scores, one for each category. SVM and Artificial Neural Network (ANN) are the most robust classifier option in supervised learning. SVM is a non-probabilistic binary linear classifier. It builds a model which represents the training examples as points in space. The model separates a new example into one class or other. Park et al. [67] obtained the foreground blob using the median filter as background subtraction. The small bounding boxes wrapping around the blob are fed to feature extraction. The HOG features are fed to SVM classifier whether the candidate is a person or not. Gleason et al. [68] proposed vehicle detection from drone image which is used for visual inspection in patrol service. The authors made the comparison with different image descriptors (HOG and Histogram of Gabor coefficients) and classifier (k-Nearest Neighbors (kNN), random forests (RF) and SVM). The dataset was collected from both indoor and outdoor. The top performer was the detector which used Gabor histogram features and classified them with RF classifier. Than Noi et al. [69] showed the comparison of different classifier for land use classification for satellite images. Overall, SVM outperformed RF and kNN. The authors observed that there was no significant difference among the three classifiers when the training sample size was large (≥ 500 pixels) apart from one specific dataset whose size was approximately 750 pixels with kNN classifier.

2.1.2 Top View Applications of Traditional Approach

Traditional worker detection methods are based on simple features like helmet and color of high visibility vest. Neuhausen et al. [71] proposed worker detection and tracking from the bird's eye view using the color of high-visibility safety vests as the image feature. The



(a) Drone top view

(b) Load top view

Figure 2.9: Top view comparison between load and drone [73].

authors initially extracted the target worker from the foreground using the color histogram of the emergency vest and pass them to the soft cascaded classifier [72]. Two color spaces, namely RGB and HSV were compared. In contrast to RGB color space, HSV is more robust in brightness changes. After the targets were generated, the soft cascaded classifier which is based on decision tree. At the end of the object detection pipeline, Kalman Filter is used to smoothen the detection result. However, this work fails to consider complex environment of the construction area including majority of the workers who did not wear safety apparel which is high visibility vest because the method highly relied on the shape and color of the target. In addition, the background subtraction method to generate target is limited to non-stationary camera.

To increase spatial awareness of the operator, Fang et al. [70] proposed real-time monitoring of load motion for an offshore platform. For load sway monitoring, the vision-based method is adopted to track the load position in addition to the crane motion monitoring using Inertial Measurement Unit (IMU), see Fig. 2.8. A monocular camera mouted at the crane boom looking downward to the ground. With the monocular camera, the authors used a color-based segmentation approach to track the large-colored marker on the load. To obtain discriminative color segmentation, the authors chose CIELAB color space instead of RGB. After the position of the load is obtained in form of the bounding box, the 3D position with respect to the camera is calculated by using camera project method, given the wire rope length and camera focal length. Nevertheless, this study would be more interesting if the authors assess the different load shapes as only in the shape of box or containers existed despite of many types of loads in actual construction.

Takahashi et al. [74] measured the three-dimensional position of the crane hook. Given a load-view monocular CCD camera, the hook region can be obtained by optical flow. The part of the detected hook region at the center is tracked based on Orientation Code Matching (OCM) [75]. According to the experiments, the authors set the size of the tracked region template is 21×21 pixels. To speed up the processing time, the authors suggested to increase the searching region to 51×51 pixels. Consequently, they obtained approximates 33 frames per second (FPS). The study was limited to the static camera mounted at the boom.

For vision-based Search and Rescue (SAR), Andriluka et al. [76] showed the comparison of victim detection from a quadcopter among five classic detectors which were based on

monolithic and part-based models. Two monolithic detector models were HOG-based with the full bodies of pedestrian and the other with only upper bodies. As the name implies, these types of detector provides global descriptor of the target. Therefore, these methods are very sensitive to pose variation which could worsen their performance. On the other hand, the three different part-based model detectors were Deformable Part Model (DPM), pictorial structures with discriminant part detector [77], and poselet based detector [78]. The authors concluded that the part-based features have better performance and more robust with partial occlusion.

Nevertheless, the above mentioned conventional detection, methods which are built on handcrafted feature such as high visibility colors, parts of object and shape, can fail to PPE noncompliant scenario. According to a survey about worksite accidents and injuries conducted by the Bureau of Labor Statistics (BLS), 84% of all workers who suffered head injuries were not wearing head protection [79]. This shows most workers do not regularly wear protective clothes. Thus using PPE information, the high-visibility color of helmet or vest, as a feature to detect workers may not be adequate. Moreover, the methods which rely on static environment or background can fail because the construction site keeps changing through the project phases.

2.2 Deep Learning-Based Top View Object Detection

As an emerging trend, the research on Deep Learning for top view object detection is swiftly expanded. Localizing the objects from very high altitude is very challenging. The target size is very small which yields insufficient image feature extraction of target.

Deep Learning (DL) is a class of machine learning techniques based on ANNs. For decades, handcraft feature extraction is limited. Traditional machine learning methods require expert knowledge to extract the features out of raw input data. To train a model of these conventional object detection, the expert indicates each part of the object rather than a big picture. For example, the face detector [63] features the differences of eye-cheeks or eye-nose bridge. DPM detector models an object based on the global and local appearance [51]. The global appearance consists of local appearances which have spatial connections as a spring among them. DPM is a graph-based method. Each part considers as a single node and each node has a cost. To illustrate, the global appearance is the whole body of a person while the local appearances are parts of the body (e.g., torso, head, arms, and legs). On the contrary, DL object detection approach is learned by showing a vast amount of image data including labels without guiding through each part of the object. Hence, it is end-to-end learning which includes complete detection pipeline, namely, localization, features extraction and classification, see Fig. 2.3 at the bottom.

2.2.1 Deep Learning-Based Detection Techniques

The notable DL-based object detector algorithms can be categorized into two classes, namely two-stage detector and one-stage detector (see, Fig. 5.1). The classification of the detectors are based on Convolutional Neural Network (CNN) or ConvNets instead of manual engineered features. The well-known CNN object classification architecture are



Figure 2.10: Mask R-CNN [80]. (a) Raw image. (b) Mask R-CNN is applied.

AlexNet [81], VGG [82], and Residual Neural Network (ResNet) [83]. More comparison of CNN-based object detectors regarding speed and accuracy can be found from [84, 85].

Similar to the traditional object detection pipeline, the *two-stage* detectors treat the detection problem in two steps. First, the object proposal generates potential candidates. The candidates are later sent to the top of the classification and regression network or their network head for further class prediction and bounding box regression, respectively. Region-based CNN series including Region Based Convolutional Neural Networks (R-CNN) [86], Fast R-CNN [87], Faster R-CNN [88] and Mask R-CNN [89] are the renown two-stage detectors.

On the other hand, *single-stage* detectors are rapid detection models such as You Only Look Once (YOLO) series [90, 91, 92, 93, 94, 95], Single Shot MultiBox Detector (SSD) [96] and RetinaNet [97]. They proposed the candidates from the input image directly without the region proposal step. This leads to simpler and faster model architecture while lessening the performance slightly. Further discussion of DL network architecture is elaborated in Sec. 5.

2.2.2 Top View Applications of Deep Learning Approach

With the outstanding results of DL methods, a researcher in the construction domain started to employ these data-hungry learning methods. Besides construction domain, it is worth to mention the literature of object detection from Unmanned Aerial Vehicle (UAV) because the aerial view and crane load view are nearly identical, see the comparison in Fig. 2.9.

Yang et al. [80] presented the automated safety distance identification system from a tower crane based on Mask R-CNN method. By monitoring the distance between worker and falling zone area, the system gives a warning to the operator if any worker accesses the hazard area which was marked in yellow and black stripe area, see Fig. 2.10. To detect worker-on-foot, the training data was first collected. Due to the difficulty in finding data from the actual crane platform, the authors created an internal crowdsourcing platform for the workers to collect the data from the real construction site. Unlike Mask R-CNN detector, the bounding box-based detectors such as HOG, YOLO, or SSD, output each recognized target as a bounding box. These detectors can have large errors when the camera is rotated or tilted. This results the target to be heavily deviated. Mask R-CNN



Figure 2.11: RetinaNet model to localize transmission line defects[©] Reprinted from [103], Copyright (2020), with permission from Elsevier.

is an instance segmentation which compute pixel-wise mask of individual object instance. This allows the operator to have better risk estimation of the worker.

Ahmed et al. [98] proposed multiple people tracking via surveillance IP camera under 5G infrastructure. The authors employed YOLOv3 as the detector with tracking algorithm called Deep Simple Online and Realtime Tracking (SORT) [99] to gain better detection performance, see Fig. 5.1b. The comparison between two YOLO detectors were made, namely original pre-trained model with MS COCO dataset and one with top view image fine-tune. With the pre-trained model, the detector was able to achieve 92% accuracy while it gained 4% better with transfer learning. Kim et al. [100] used YOLOv3 to localize objects to further monitor proximity between mobile constructions resources.

Zhu et al. [101] applied a two-stage detector to detect person from UAV. The authors improved Faster R-CNN to detect small-size targets and reduce overlapped bounding boxes among targets. To improve small target size detection, the authors extended the anchor size to cover smaller target size. Two additional anchor scales, namely 32×32 and 64×64 were added in RPN to match small object size. To handle overlapped or proximal targets, RoI pooling was replaced by RoI align [89].

Kim et al. [40] developed a prediction model for trajectory of mobile construction resources using UAV image data. The Socially Acceptable Trajectories with Generative Adversarial Networks (Social GAN) model is able to predict for more than five seconds in advance. The trajectory is further used to monitor struck-by accidents.

Barekatain et al. [102] proposed an action detection model to detect person using SSD[96] as a base architecture. The author trained the detector with different daily actions e.g., lying, walking, pushing, and calling. However, the worker activities on the construction site are quite different than the actions of the pedestrian. Some examples of worker activities are lay brick, nailing, welding, etc. Evidently, the worker's action basically appears as squatting, bending, dragging objects, etc. Such approaches, however, failed to address the unstructured background, change in object appearance, or viewpoint.

Kapania et al. [104] combined both YOLOv3 and RetinaNet as detectors for multiple object tracking with drones. The authors reason of using the two detectors is to bring out the best properties of each detector and compensate the drawbacks of both. While YOLOv3 usually fails to small object size or objects in close proximity, RetinaNet performs well in small and dense objects. NMS was later applied to remove redundant BBoxes. In the same manner as [98], the tracking algorithm is based on Deep SORT. For transmission line defect recognition via UAV, Liu et al. [103] increased the performance gain of the RetinaNet-based



Figure 2.12: Examples of AV Dataset. (a) ApolloScape [109]© IEEE 2018. (b) Waymo Open Dataset[110]© IEEE 2019. (c) Cityscape [111]© IEEE 2016 and (d) KITTI [112].

detector, see Fig. 2.11. There were four common defected on the transmission line which are tower, fittings, insulator, and ground wire. The authors made several modification on the RetinaNet model for the improvement. For instance, the original backbone of RetinaNet (i.e., FPN) was replaced by DenseNet [105]. As a result, the frame rate and the accuracy become higher. In addition, the anchor box generation mechanism of YOLOv3 was included to obtain better quality of anchor box. In the end, the authors demonstrated the improved RetinaNet outperformed Faster R-CNN and YOLOv3.

Golcarenarenji et al. [106] proposed CraneNet network architecture for top view worker detection for a mobile crane in safety monitoring. The network was trained with Visdrone 2019 and Stanford Drone Dataset (SDD). The CraneNet was based on Path Aggregation Network (PANet) [107] which contains deeper network layer to gain more meaningful information and improves small object detection and Spatial Pyramid Pooling (SPP) [108] which expands the receptive field of CraneNet backbone network. Despite the good accuracy, it would be more interesting that the authors considers target without high visibility emergency vest. Because the targets are usually blended to the background which result more challenging in worker localization, especially at very high altitude.

The performance of deep learning methods is highly dependent on the existence of ample training samples. There is an intensive shortage of training data and pretrain models in the construction domain. Laflamme et al. [113] present the self-driving car datasets publicly exist in great amount such as KITTI [112], Waymo [110], Cityscape [111], ApolloScape [109] and Mapillary Vista [114], see Fig. 2.12. Each dataset has its own strengths and weaknesses.

Unfortunately, these datasets do not apply to load-view object detection due to the frontal viewpoint. The street environment is quite structured, unlike the construction site which



Figure 2.13: Drone dataset. (a) Stanford Drone Dataset© Reprinted by permission from Springer Nature: Springer [48], Copyright (2016). (b) VCI-CITR and VCI-DUT dataset [24]© IEEE 2019. (c) Okutama [102]© IEEE 2017.

is always rapidly changing. The objects that appear in between two scenarios are entirely different.

In addition, UAV datasets (e.g., SDD [48], VCI-CITR and VCI-DUT dataset [24], Okutama [102], see Fig. 2.13, could not be used as an alternative because of an uncluttered and static background. The pose or activity of the worker and the pedestrian are not identical, which can lead to different image features. Lastly, there is another solution of data gathering for the data-driven approaches is to exploit the simulation platform which will be further discussed in Sec. 4.3.

3. Thesis Overview

Top view vision-based object detection attract researchers' attention in recent years. It is used in surveillance, crowd analysis, safety monitoring, etc. These applications require observing multiple objects simultaneously in a large area. Top view image is taken vertically from an elevated position such as aircraft, buildings, and huge construction machines. The name of the top view can also be seen in different terms such as bird's-eye view, aerial view, overhead view, and top-down perspective. The view can provide the existence of the objects in the large viewpoint rather than the detail of every single object. On the contrary, the front view, such as a dashboard camera view, is taken where the observer is at the same elevation as the object. Unlike the top view, there are fewer visible objects can be captured and more inter-object occlusion.

Detecting objects from the top view is significantly used in a wide range of application domains, from safety to disaster relief. In general, such application areas are inaccessible or hard for ground vehicles or humans to reach. Additionally, the idea of top view perception is gathering the information in a big picture scheme instead of detail specific. Therefore detecting objects from the top view is a suitable approach for these applications. In traffic monitoring, the surveillance camera mounting on the light pole can observe the vehicle movement or indicate the density of the vehicles on the road. Regarding the rescue mission, a camera is mounted on Unmanned Aerial Vehicle (UAV) to recognize any victims in serious disruption areas. Another example would be the construction area. The information from the surveillance camera is used to estimate the construction asset. For safety monitoring, a crane operator is unable to see any worker who is immediate to a load from the cabin because the workers could be obscured by the load. Instead, the camera attached at the top of the crane arm is used to facilitate the operator during lifting operation and importantly notice if any worker comes nearby the load to avoid struck-by load accidents or a situation where a worker can be hit by a load.

3.1 Challenges of Top View Object Detection

Detecting objects using a monocular image sensor from the top view is extensively employed to perform autonomous operation [115]. The position of the identified target is later used



Figure 3.1: MC5200 on crane. [118]

in higher-level image processing for semantic understanding such as pose estimation, activity recognition. Despite remarkable results, there is a big room for improvement due to different challenges, such as changes in scale and viewpoint. As previously stated, top view observation offers a larger viewpoint and higher numbers of object existence, comparing to front view. However, the bigger the area covers, the smaller the object size becomes. Consequently, the traditional object detections suffer from manually defining the image features or environment constraints such as object apparel [71], background [34]. Bhattarai et al. [116] employed a binary classifier on Haar-feature to detect a person from UAV. The authors mentioned the tradeoff between object size and accuracy. Such approaches, however, failed to address the unstructured background, change in object appearance, or viewpoint. The previous survey[117] showed the comparison of top view detection methods using UAV in three different height intervals. The eye level is the lowest drone flying range which is at the same level as a human viewpoint. The low-medium range is the most common flying zone i.e., 5 to 120 m above the ground. The rest falls into the aerial level which is higher than 120 m. The authors mentioned the changes in size and perspective result in various and specific detection techniques. Besides the impressive outcome of data-driven detection methods, the top view object detection using Deep Learning (DL) approach is far behind, comparing to the front view. The difference of the best Average Precision (AP) achievement in object detection benchmark between the front view and the top view is very large. In spite of different evaluations, the highest AP of front view object detection benchmarks is approximately 96 percent [119] whereas the statistic of top view detection benchmarks is merely 30 percent [25]. Zhu et al. [120] points out the serious and open problem in data inadequacy of aerial image. The data generation for the top view is problematic due to the unreachable area and small object size which is very difficult to annotate. In the absence of data, the DL methods are not able to reach the promising outcome. There are numerous test platforms for self-driving


Figure 3.2: Overview workflow of this thesis.

cars [121]. To conduct the experiment, the test platform for top view autonomous operation is indeed necessary. Nevertheless, the simulation platforms for top view object detection are limited [122, 123].

The aim of this thesis is to develop a robust vision-based 2D object detection with adaptive zoom camera from top view images using a deep learning algorithm. To solve the challenges of top view object detection, we opt to investigate the problems in the construction area as the use case. The motorized zoom camera on the mobile crane is employed as an image sensor to recognize the object from above. The camera is attached at the boom end, shown in Fig. 3.1 and always pointing downward to the ground. During lifting operation, it moves straight up and down along the movement of the crane arm. The target object is a worker-on-foot who works on the ground around the load. The arguments of choosing the construction area as the use case are following. First, the environment of the crane camera to the ground is frequently changing during the lifting operation. Lastly, research activities of the construction industry are fairly small relative to autonomous passenger vehicles. Ultimately, detecting objects from the top view in a construction area contains most of the mentioned challenges.

The research objectives fall into the following steps. To overcome the drawback of the traditional methods, we investigate the suitable DL network models and their parameters that are competent for the complex environment with several viewpoints. The speed performance should be assessed, as well as the accuracy because most standard image sensors to obtain top view images are typically small, compact, and low resolution especially in industrial machines. The space and weight of the sensor to install on the platform are very restricted.

Regarding the data deficit in data-driven methods and limitation of test platforms, the simulated environment is developed using a game engine, Unreal Engine (UE). The environment is built resembling the selected use case. A good amount of synthetic datasets



Figure 3.3: Summary of thesis contributions.

are generated and expected to increase the quality of the detection method. UE has been commonly used in many applications e.g., film industry, architecture, automobile, robotics. It can provide the realistic simulation in various perspectives such as texture and physical properties. Therefore, we want to analyze the realism effect of training using synthetic data whether it can improve the accuracy of the detector in the real world. Lastly, we inspect the zoom mechanism of the crane camera to address the scaling problem. The analysis of the zoom controller should be executed to see how much it can maintain the object size.

The workflow to fulfill the research objectives is briefly shown Fig. 3.2. The concept of this work can be adapted to other application areas e.g., using a surveillance camera to detect objects in an urban area instead of using a crane camera on a construction site. To bridge the gaps in the aforementioned challenges, the following contributions are made. Each problem will be first described in detail and followed by the associated contribution. The contribution overview of this thesis is depicted in Fig. 3.3.

• Generating and Analysis of synthetic data to enrich the quality of top view detection [124, 118]:

The data-driven object detection methods require good quality of data in great amount [125]. Despite the vast number of the public dataset, the groundtruth are faulty due to interpolation among frames and human errors. To the best of our knowledge, there is no public dataset for top view worker detection in the construction area. On the other hand, data generation is tedious, especially for huge construction machines.

In this work, *simulation platform* [126] for a mobile crane is developed using UE, shown in Fig. 3.4. This platform has the advantage of a more comparable environment



Figure 3.4: Simulation platform of Steil Kranarbeiten, Trier, Germany

to the actual construction including surrounding objects. The environment can be modified in a fast manner. Another practical advantage of the platform is that it can be used as a lab-based experiment for a mobile crane which does not exist in other works. The simulation platform is necessary for construction research, especially automation and robotics. The construction site is hazardous. The simulation platform contains an immediate connection to the robotic framework, FINROC¹. By using the platform, many experiments or algorithms can be achieved concurrently without any risk and high costs.

The simulated environment is used for *dataset generation* to support object detection using DL. Numbers of synthetic datasets are generated using the developed platform [124]. The crane camera recorded the data in different heights, weathers, zoom levels, and camera movement. The benefit of using this platform is expected to reproduce the zoom function or crane maneuverability to collect the identical data as from the real crane. There is no compatible public dataset that is suitable for worker detection using a load-view zoom camera. Therefore, the author believes that these synthetic crane datasets will be valuable to the construction research domain whose dataset is hardly available. Lastly, the *evaluation of synthetic data* in [118] demonstrated the possibilities of using synthetic data in training. In addition, there is a significant improvement of more than 10% in average precision between the pre-processing and non-preprocessing synthetic dataset.

• Top view object detection using Deep Learning [118]:

Most of the traditional object detection methods are based on simple feature extraction and image processing. Object regions are given by background subtraction[34].

¹Framework for Intelligent RObot Control (FINROC) is a real-time robotic framework based on a systematic design. It has been developed at the RRLab of TU Kaiserslautern, Germany since 2008[127]



(a) load-view camera



(b) surveillance view camera

Figure 3.5: (a) shows the yellow-black color combination area is defined as physical hazard zone according to OSHA [80]. (b) PPE non-complaint workers working nearby the heavy machine in yellow locating on the right of the figure.

Additionally, the image features of the traditional worker detectors are limited to PPE [71] or the marked danger zone, see Fig. 3.5a. Such protective gears have high visibility colors and specific shapes i.e., circle (helmet). Color histograms such as HSV, YUV, or RGB color space are incorporated to represent the workers. Nevertheless, the methods do not account for manifold of PPE noncompliance, see Fig. 3.5b or dynamic changes in the construction environment.

With the DL breakthrough, object detection using a data-driven approach is employed in this work. In contrast to traditional methods, this method has the advantage of an end-to-end learning concept which allows the Neural Network to automatically find out the most descriptive features which are suitable for each object class. The approach can overcome the conventional methods which are highly committed to PPE presence [71].

A further question is whether person detection from UAV image using in typical outdoor environments (e.g., university, city, sports ground) could be an alternative solution [102]. In fact, the detection algorithms in UAV are limited to a plain background, invariant height, and importantly restricted to objects which differ on appearances and activities. Therefore, the study of worker detection via load-view crane camera particularly remains to be explored.

Detecting workers from load-view is challenging. In particular, no study, to the author's knowledge, has considered the worker detection from the load-view zoom camera. This remains an open problem in the area. Such the industrial crane camera has low pixel resolution. It also swings with pendulum motion which adds more degree of difficulty, see Fig. 3.1. Selecting network architecture is another challenging problem due to time and accuracy trade-off [84]. In this work, the studies will further discuss the network option in Chapter 5 concerned hardware resource, accuracy, etc. The analysis of parameter tuning including the training process is investigated and properly customized to the application in this task. Based on the evaluation, the optimal network is chosen.

• Analysis of adaptive zoom to maintain the quality of top view detection [128]:



Figure 3.6: Snap from load-view Orlaco. The red bounding boxes are annotated the person in the image frame. [131]

Data acquisition is one of the essential stages in the object detection pipeline. To the author's knowledge, no previous research has investigated the zoom mechanism on a mobile crane for object detection. Alternatively, the previous research on the safety monitoring system using object detection attempt to exploit additional particular sensors to augment the perception information. Unlike a crane zoom camera, these non-standard crane sensors, like laser scanners, significantly increase the hardware cost. The undesired cost explains the small usage of the LiDAR in the industries [129]. Hence, more specific research to investigate the existing crane sensor, which is the load-view camera, is needed.

The in-depth literature review in Sec. 6.1 shows that the zoom function of the camera can be used to improve accuracy. Nevertheless, the methods presented solely the problem of zoom constraints. For example, the camera is zoomed in or out to hold the certain range of pixel ratio between the recognized object and the image width [130]. These studies would have been more useful if they include how to control the zoom. In practice, the unsteady pixel of the object can affect zoom oscillation. Zoom control is a challenging subject. To zoom to a certain condition e.g., the defined specific pixel size, it requires stable zoom control to maintain the state. Thus, the problem regarding handling zoom control remains to be addressed.

In this work, the *control adaptation* in optical zoom function of the crane camera is investigated. The zoom level adjustment of the camera is mathematically modeled in the finite state machine under the safety criteria. The state transition is handled continuously. The (semi-)automatic zoom can be changed based on the recognized worker-on-foots. During operation, the crane operators have to simultaneously concentrate on every surrounding and can be a lack of situational awareness. Hence, this adaptive zoom gives a significant advantage for the crane operators.

Besides operator support, the adaptive zoom feature gives a favor to the DL worker detection. The DL detection algorithm can fail to insufficient training various sizes. Although DL is scale-invariant, working crane radius are very wide range and it is different from one crane to another, see Fig. 3.6. To avoid extensive retraining and re-optimization, the zoom adaptability of the load-view camera supports the DL algorithm in a similar manner as the operator. In other words, the operator can see or the camera can automatically detect the workers closer by the adaptive zoom function.

In general, DL methods neither can yet solve every computer vision problem nor are widely accepted among industries [132]. Such algorithms can be deceived by simple tricks [133]. So far, there are failures of the data-driven approaches in autonomous vehicle car accidents [134]. Hence, the adaptive zoom feature can be another advantage compared to applying the DL method alone.

4. Analysis and Generation of Real and Synthetic Data

To address the problems of top view object detection, we investigated the construction domain as the use case. The definition of the construction area in this thesis work is an area where construction operation is carried out, see Fig. 4.1.

As mentioned in Sec. 3.1, we aim to increase the situational awareness of the crane operator by proactively recognize the struck-by load hazard i.e., the workers can be hit by the crane load during the lifting operation. To warn the operator, any workers nearby the load radius are visualized and highlighted on smart glasses, Microsoft Hololens, see Fig. 4.4b. Furthermore, we had an opportunity to work with the real setup including the experts, who are crane lift planning designers and camera solution system provider for construction, as part of this research project. With the close collaboration with crane industries, we gained knowledge about the crane safety regulation in practice, how crane trajectory should be especially during the blind lift, the suitable hardware including their position should be mounted, etc. As our contribution in this project, *Top view worker detection* provides meaningful information to the crane operator to be aware of the surrounding risk.

The structure of this chapter will be first discussed on how the construction operation task and brief safety regulation are. Secondly, the literature review related to simulation usage in construction domain and dataset generation will be elaborated in Sec. 4.2 and 4.3. Sec. 4.4 describes the scenario and the environment where usually causes accidents and also were used for running the experiments in this thesis. The hardware setup in the real world will be later presented in Sec. 4.5 then followed by the transition from the real world to the proposed simulation platform in Sec. 4.6. Importantly, it is necessary to investigate the simulation platform for construction automation and robotic research experiments such as worker detection or dataset augmentation for Machine Learning.

After the development of the simulation platform including the hardware setup in the real world is presented, Sec. 4.7 and 4.8 present the approach on how the datasets were gathered both in the actual setup and using the virtual setup, including an analysis of the synthetic data. The data generation procedure both in the real world and the virtual platform is first described in two steps i.e., data recording and data annotation. All the



Figure 4.1: The examples of the typical industrial or construction areas where a crane operates. (a) Surveillance camera view. [34]. (b) and (c) illustrate view from crane camera. (d) depicted top view from the 6th floor building using load-view crane camera.

generated datasets including the public datasets will be later summarized and used to refer for the rest of the thesis. The evaluation metric for object detection algorithms for the experiments is introduced in Sec. 4.10. Finally, a part of the generated datasets was adopted in the two experiments for the analysis of the synthetic data in Sec. 4.12.

4.1 Crane Operation and Safety

A crane is a central machine that serves in many construction operations. This powerful machine is the main drive for the construction project. It provides great mechanical support which is moving materials or loads beyond the normal human capability. However, keeping the construction site safe is challenging because the area dynamically changes across the construction phases. The size of the crane including the loads is huge. Any small mistakes can lead to great damage or loss. Among the nine major economic sectors, construction has the third highest fatality rate [12]. Approximately 30 percent of all construction fatalities are related to cranes [16].

For lifting operation, the operator works with several personnel (a)rigger and (b)signaller. The riggers set up and connect the lifting equipment (e.g., a hook) to the load and assure that they are tight and will not fall down during the lifting. They work closely to the load during not only the attachments (see Fig. 4.2b), but also the lifting. Sometimes, the



Figure 4.2: (a) The overview of crane personnel during the operation. (b) The rigger works closely to the load and the crane.

operator needs small adjustments to the load, whose movement is too small for the huge machine to perform. Therefore, the riggers have to either drag load strap or go close by to push/pull the load. On the other hand, the signaller, who acts as additional eyes of the operator, mostly stands nearby the crane cabin, see Fig. 4.2a. He gives a crane signal which provides the proximity information of the crane and the load for the operator. The signaller guides the direction or indicates obstacles via either hand signals or a handheld transceiver (HT) when it is out of the operator's sight.

4.1.1 Construction Safety

In construction operation, safety measures have to be executed in adequate manner. The safety officer is responsible to build a safe environment for all construction workers and assure all the construction workers follow the safety regulation. Traditionally, the safety personnel or supervisors walk through the construction site, observe other workers and notify them if any potential hazards (i.e., unsafe conditions or acts) are found. The job site inspection and observation are performed every one or two weeks depending on the size of the project [135]. Each inspection takes 1-2 hours at an arbitrary time.

Safety monitoring tools or safety assistance systems, *visibility* in particular is important for the situational awareness as it allows crane operators to be able to *see* what is happening around the equipment. The struck-by equipment or vehicle accidents are highly visibilityrelated. Human highly depends on sight. In general, human obtains more than 90% of information transmitted to the brain is via visual sense [136]. Limited or poor visibility such as blind spots or obstruction accounted for 82% of all visibility-related fatalities [137].

In a crane, the operator has to perform blind lift when the operator is unable to see the load or personnel from the cabin, see Fig. 4.2a and 4.3. Without any crane assistance system, the crane operator has to rely on the signaller or the load-view camera; however, the communication can go wrong or be delayed, which can lead to an accident[138]. The crane operator can maneuver the load to strike the workers unintentionally because of invisibility. For instance, the operator in Fig. 4.2a is in a higher position than the worker and the hand signaller are. Apparently, he may not be able to see the people on the ground below directly from the cabin, particularly blind spots. Despite the load-view camera,



Figure 4.3: Blind lift at the test location, Steil. [73]

the operator has to change the zoom level by himself in the way that he can see all the workers.

To avoid struck-by accidents, the operator has to always manually observe where the worker-on-foot and the load area from the small screen (7-inch), see Fig. 4.13. In particular, it is more difficult for the operator to observe the workers especially when the workers do not comply with the safety regulation. The operator may not be able to see the worker who does not wear the highly visible color vest in the construction area because their appearance becomes uniform to the background. Moreover, when the camera zooms in, Field of View (FOV) becomes smaller. As a result, the observable area for the operator is narrower and does not cover the area as large as zooming out at maximum. For example, the worker is visible to the operator when the camera is at maximum zoom out. After the operator zooms the camera in, the worker can not be anymore seen by the operator despite no change in the worker's position. As can be seen, the operator has to change the zoom level by hand during the operation in order to observe the workers.

In general, the off-the-shelf camera video system merely offers visual information but not in a semantic manner. This assistance system, which is the load-view camera feed system, does not provide an automated alert or point out where a hazard is. The operator has to observe and further make decisions based on the given raw information i.e., control feedback or video stream. As a result, they do not remove all of the human factors as the operator yet performs multi-tasking which is vulnerable to getting into an accident. The operators have to ballpark estimate on possible risks that may cause the accident based on their experience.

(Semi-) automated, continuous, and semantic monitoring is considerably useful for the construction operator. To observe the safety surrounded the crane, the *location* of construction resources including their relative locations must be first determined. To achieve the position or location of the construction resources, *top view object detection* from the crane camera is employed in this thesis work to provide the location of the worker



Figure 4.4: (a) The top left image shows the view of the operator which can be seen from the monitor inside the cabin during the blind lift. (b) The crane operator wears Microsoft Hololens, a pair of mixed reality (MR) smartglasses. [73]

surrounding the crane to increase the situational awareness of the operator during the safety monitoring.

The cranes are used everywhere and not limited to construction areas such as on the street right next to the pedestrian path or sports stadium. In this work, we opted to investigate the challenging environments which is an industrial or construction area. The following sections will describe the scenario including the typical setup of the top view object detection in the safety monitoring system.

4.2 Simulation Usage in Construction

A simulation is a tool that a scientist primarily uses for trial-error experiments. This technology builds the mathematical model to estimate the physical or real-world outcome. Based on the outcome, it can be further used to analyze for the optimal solution. The simulation is originally used to increase productivity during World War II [139]. Jon Von Neumann and Stanislaw Ulam simulated the behavior of neutrons to find out how far the neutrons would travel in different materials [140]. The electric roulette wheel is used as a physical random number generator in hit and trial experimentation. As an outstanding result, the method is known as Monte Carlo became popular and is adopted by many applications including industry and business.

Simulation eases humans in problem-solving. It offers graphic and data visualization instead of text form. In general, the human brain can process visual data 60k times faster than text, and more than 90 percent of transmitted information to the brain is visual [141]. Therefore, a human can effectively use this skill to analyze the data pattern and have a better understanding of the problem. As an illustration, it is better to visualize a room temperature or weather forecast as heatmap colors instead of showing the degree numerically e.g., red shows as a high range temperature while blue shows a low range temperature. Some examples of notorious simulation software tools are MATLAB, Simulink, AnyLogic, Ansys, etc.

Besides data visualization and experiment independence, simulation can increase safety while saving cost and time especially in robotics or autonomous vehicles. These applications

Application	Simulator platform
AV : passenger	CARLA, Apollo, Udacity, LGSVL, Virtual KITTI [153, 154], DeepGTAV [155]
AV : racing car	TORCS
Underwater robotics	UUV, UWSim
UAV	AirSim, UdaciDrone
Rescue robotic	USARSim [156]
Commercial vehicles	RRLAB [157, 126]
Computer vision	UE4Sim [158], Sim4CV [159], UnrealCV [160]
Vocational training	LiSIM

 Table 4.1: Examples of robotic simulator.

essentially need simulation for research and development. The simulation allows the researchers to concurrently work on different parts e.g., vehicle control, vehicle perception, and assess their particular algorithms. In addition, the researcher can work on a physical robot without modifying the actual robots. A large number of robot simulators are available such as Gazebo [142], Virtual Robot Experimentation Platform (V-REP) [143] and CoppeliaSim [144], see Fig. 4.5. Initially, the goal of the robot simulator targets merely on the physical properties and the main robot function itself. However, the test environment of these robot simulators is limited or invariant. Further comparative studies of robotic simulators can be found in [145, 146]. Table 4.1 lists the robotic simulator examples based on the application.

Simulating diverse environments for testing a robot is another challenging problem. Despite the ability to create complex manipulator robot arms or quadruped robots, the traditional robot simulators suffer from building large-scale complex virtual environments which are nearly identical to the real world [122]. Game engine (e.g., Unity [147], Unreal Engine (UE) [148]) started to draw a great deal of interest among scientific research. The game engine platform is a software development environment for people who aim to create video games. Likewise, the game engine shares the same goal as the robot simulator, which is having a realistic environment. It also delivers good accurate and precise modeling such as Microsoft Flight Simulator (MSFS) [149]. The highlight of the game framework is a real-time rendering engine or renderer [150]. It can generate a photorealistic for 2D or 3D graphic models. Other game engine features are a physics engine, collision response, character animation, acoustic, networking, artificial intelligence, etc. [151]. Furthermore, the game engine software architecture contains a handy interface and libraries including plugins. The plugins can be obtained from the e-commerce platform e.g., UE4 Marketplace [152] where other developers provide game-ready content and code.

4.2.1 Simulation Platforms

Robotic researchers specifically in the autonomous vehicle area exploit the game engine features to develop autonomous vehicle simulation platforms. The architecture of an open-source game engine allows the developers to interface with their robotic framework.

Numerous game engine-based simulation platforms for autonomous driving research are available. Car Learning to Act (CARLA) is a UE-based platform that not only supports the diverse sensors but also validation features of the autonomous driving algorithms with perception and controls [161]. The architecture is scalable with a client-server model. It



(c) Sim4CV [159]

(d) Gazebo [142]© IEEE 2004

Figure 4.5: Robot Simulation platforms.

offers simulated application programming interfaces (APIs) which allow the user a high degree of flexibility such as traffic control, map generation, and pedestrian behavior.

Additional self-driving car simulators are Apollo simulation [162], Udacity Self-driving Car Simulator [163] and LGSVL Simulator [164] which are made in Unity, a game engine. TORCS (The Open Racing Car) is designed for AI racing game [165]. Reviewing of the self-driving car simulators can be found from [121]. For underwater robotics, there are UUV Simulator (Unmanned Underwater Vehicle Simulator) [166] and UWSim (The UnderWater Simulator) [167] while AirSim [122] and UdaciDrone [123] are the UAV simulators. Most of the simulator platform are based on Unity or UE with connections to the ROS robot framework interface [168].

Despite extensive research on Autonomous Vehicle (AV) simulators, there is a lack of simulator platforms for commercial vehicles (e.g., excavator, mobile cranes, drum roller) whose dynamics are different than a passenger car. Commercial vehicles started to employ AV technology; however, not everything can be reiterated. To have reliable robotic systems, the robot must be able to operate in the actual environment. As previously mentioned, one of the advantages of the game engine over a traditional robotic simulator is being able to create a realistic and complex environment. The difference between driving on the street and driving on a construction site or forest are at the opposite poles, see Fig. 4.6. To illustrate, the street view is plain, well-structured including the surrounding objects (e.g., traffic lights, pedestrians, trees, buildings). The road path are clearly defined with many landmarks and references to facilitate the navigation algorithm. In constrast,



(a) Road [112]

(b) Construction site [34]

Figure 4.6: Comparison between different scenarios: road vs construction site.

non-street environments are changed rapidly. The surrounding objects appear and behave differently. Wolf et al. [157] demonstrate a complete simulated robotic framework for complex commercial vehicles which are used in an unstructured environment such as off-road vehicles and construction machines. The author implemented the interface between UE and FINROC, a framework for intelligent robot control of RRLAB [127]. There exist a variety of simulated sensors (e.g., IMU, GNSS, cameras, 3D lasers) and environments (e.g., quarry, forest, construction area).

In construction, simulation can improve safety and worker's health quality in a less cost and time-efficient manner. The simulation is generally used in the main construction phases e.g., vocational training, planning, and operation. For *training*, Virtual Reality (VR) recently becomes a trend in vocational education. Krafft [169] presented how Unity-based VR platform helps the worker to improve the safety, see Fig. 4.7a and 4.7b. The VR provides more reality and essentially affects the decision-making of the trainee during training compared to conventional training which is lecture structure or watching the class from the screen and answers the questions. The VR platform allows the worker to experience and physically interact with the current situation which happens right in front of the trainee. Zhao and Lucas [170] developed a simulation for the virtual reality-based safety training program. The program allows the users to do the safety practice for electrical hazards repetitively without actual danger. For crane operation, crane simulators LiSIM (Liebherr Simulator) in Fig. 4.7c which is developed by Vortex Studio, a simulation software [171]. Liebherr offers several location solutions i.e., classroom, cabin, and container. The scenarios and environmental conditions, which the tool can generate, are diversified.

In *planning*, the lifting trajectory has to be planned before the incoming operation to avoid the possible hazards. AlBahnassi and Hammad [172] present the framework for motion planning to avoid collision among cranes. Many key factors have to be considered such as site conditions, load, equipment, and types of lifting. The lifting routes and basic obstructions are simulated and visualized. During *operation*, Fang et al. [173] use a game engine, Unity to acquire sensor data real-time and visualize their application interface. The application is a framework that eases a crane operator specifically for blind lift.





(c)

Figure 4.7: Different VR Simulation platforms. (a) VR in Unity. [169]. (b) VR in UE. [174] and (c) Crane simulator, LiSIM (Liebherr Simulator) for vocational training. [175]

4.3 Dataset Generation

To yield high accuracy, deep learning-based object detection algorithms require large-scale image datasets as learning samples. There are great amount of self-driving car datasets mentioned in Sec. 2.2 including synthetic data like SYNTHIA dataset [176].

Dataset acquisition consists of two main steps, *data recording*, and *data labeling*. *Data recording* is a process that collects the data from the different environments where the robot should operate including multiple sensor installations, calibration, etc. Next, the collected data has to be *labeled* as the important information in the scenario for operating of AV system and decision-making algorithm of the vehicles e.g., traffic light, road lane, traffic sign, static and dynamic surrounding objects.

For non-street or off-road areas, recording data is not straightforward. It is limited due to the legal action. For example, driving through a forest in Germany is only permitted with the consent of the forest owners [177]. In the same way, a drone or UAV is not allowed to fly 100 meters above the ground without a permit and only 50 meters high is allowed in controlled airspace [178]. The aircraft pilot is mandatory to have a license to fly a drone weighing more than 2 kilograms. On the construction site, there is an expense for construction equipment rental including related personnel (e.g., rigger, operator). For instance, a crane rental including an operator and heavy cargo liability for a basic task operation in Germany costs approximately 1130 euros per day[73]. The



Figure 4.8: Example of difficult annotation. It is difficult to label every single objects in each image. [180]

operator requires additional specific driving licenses and training. Installing sensors can be difficult to (un-)mount and adjust due to the large machine size. To assure that the outdoor robots function in all situations, every possible states of weather should be collected. This introduces another difficulty of data recording as it has to be based on unpredictable nature such as rain, sun, and snow. Image annotation techniques can be manual, semi-automatic and automatic [179]. Annotation is not simple, see Fig. 4.8. Manual annotating data is tedious and can create localization error. The annotators require the knowledge to define object boundary and must follow the labelling policy e.g., occlusion constraints, object representation and boundary [180]. For the very largescale dataset, there exist crowd-sourcing platforms, such as Amazon Mechanical Turk (MTurk) [181] and Amazon SageMaker Ground Truth [182], to gather image annotation possible. The annotated data needs to be verified. Semi-annotation tool (e.g., CVAT [183], Labelbox [184], SuperAnnotate [185]) includes a linear interpolation feature for image stream which allows the annotator not to label objects in every single frame. Despite the defined annotation guideline (e.g., [186, 187]), the ground truth data is yet not clean or perfect. In particular, the size of the BBox is much bigger than the objects or the ground truth box has a poor alignment, see Fig. 4.9. Consequently, these error deviates and worsens the learning algorithm. Milan [190] left a remark in his public dataset portal that "Bounding boxes are not always perfectly aligned due to articulation, interpolation, and mistakes made by the annotator.". Zhang et al. [188] showed the localization errors of original annotation in Caltech dataset [191]. Ammar et al. [189] found annotation error and BBoxes are not fit in Stanford Drone Dataset (SDD) [48]. Nechyba and Schneiderman [192] show error due to groundtruth interpolation.

Simulation, therefore, helps to augment data while reducing localization error and time from the manual labeling. Table 4.2 shows the comparison of data generation productivity between simulation and real world. Using simulation to generate data costs around 400 times per image cheaper than generating in the real world. Soltani et al. [194] propose an automated annotation using the synthetic image that can reduce the annotating time while improving the detection accuracy. Another automated data generation is developed in [195]. The authors exploit ray-tracing techniques of UE to extract annotated BBoxes. Kolar et al. [196] propose guardrail detection based on CNN. The authors indicate promising results by using a combination of synthetic and real data. The synthetic data is generated by placing the guardrail 3D model on the real-world background of the construction site.



Figure 4.9: Ground truth localization error: (a) Ground truth error of Caltech on body part [188][©] IEEE 2016. FP is shown in red, while original annotations in blue and TP in green. The dashed blue box can be ignored in this case due to irrelevance. (b) Imprecise ground truth of SDD dataset which is much larger than the actual object [189].

	Synthetic	Real World
Dataset Size (images)	1,000,000+	1,500
Dataset Preparation Time (hours):		
Acquisition	5	10
Content	70	0
Annotation	8	110
Simulation	13	0
Total Dataset Cost	\$7,200	\$4,800
Cost per Image	\$0.0072	\$3.20

Table 4.2: Data generation productivity. [193]

Neuhausen et al. [197] demonstrate the synthetic data can be used as an additional option to augment the data. The authors compare the detection and tracking errors between actual data and synthetic scenario which is mimic from the real world.

4.3.1 Gap Between Synthetic and Real Data

Although the simulation platforms allow researchers to generate large quantities of training data, there is one important remark in the using of the synthesized images which is how similar between synthetic data and real data. The estimation of similarities or differences between two image data sources helps in defining the qualities of the synthetic data. In like manner, the machine learning literature employ this kind of benchmark as error metric to model or generate a novel style of image base on the original which is called Neural Style Transfer (NST).

To generate the synthetic data as close as in the real world, the data generation process has to be decently handled and ensure them in the good distribution. Henceforth, such the virtual data can potentially be a complement or alternative solution to improve the quality of the object detection using data-driven approaches in addition to the lower data generation cost and less time-consuming. Generating synthetic data copes with a problem of how to methods *model* and *render* scene to be comparable with the real data. This can be carried out by setting the scenario manually or using the machine learning methods e.g., Domain Randomization (DR), Generative Adversarial Network (GAN)[198, 199], noise modelling. Modelling or generating the scene content is the process of constructing the virtual environments and objects. On the other hand, scene rendering is the process of light transport in the environment or how the light transfers from the light sources to the camera. Takemoto et al. [200] refined the synthetic data with the real noise using NST technique. Maximov et al. [201] addressed the generalization in training deep learning networks for depth estimation. The authors in [202] applied DR to minimize the difference between the synthetic data and real world data in training DNN object detection. Jain et al. [203] adopted the euclidean distance as the error measurement between real and synthetic data. As a final note, a complete survey of image synthesis including the experimental comparison is beyond the scope of this thesis (see [204] for an overview).

4.4 Environment and Scenario

The chosen location for the experiment is the headquarter of the crane service provider company, Steil Kranarbeiten GmbH & Co. KG which is located in Trier, Germany, see Fig. 4.3. The operating area covered over 20,000 m^2 . The ground surface is gravel soil or asphalt, see Fig. 4.1b and 4.1c. It is one of the main hubs for crane maintenance, repair, and checkup service. Although Steil is not precisely categorized as a construction area, it is an industrial area or a typical environment where accidents or incidents could possibly occur. This is because the crane activities happen all the time over the place e.g., crane assemble and disassemble. As can be seen in Fig. 4.3, the area is occupied with various cranes or heavy vehicles, counterweights, dummy loads, containers, boom parts, etc. There are forklifts or trucks that drive around and workers carry tools or work with devices. The test crane used in real environment is a telescopic crane Liebherr LTM1130, see Fig. 4.10. The maximum load capacity is 130 tons. The maximum working radius is 72 m and the maximum hoisting height or the maximum height where the crane can lift is 91 m. The telescopic boom can extend from 12.70 m to 60.00 m. For safety reason, only certified operator is permitted to maneuver the crane.

For the experiments at Steil, the crane operator performed the blind lift, which is a common movement in crane activity, see Fig. 4.11. The operator moved the dummy load block from position A to position B. To reach position B, the operator had to pass the load over the white building. While dropping the load at position B, the operator was unable to observe the load from the cabin besides the load-view camera or getting the information from the hand signaller via the handheld transceiver. During the lifting, the operator could observe the annotated worker walking, see Fig. 4.12.

In addition to the test location Steil, a building at TU Kaiserslautern (TUKL) was used to further assess the proposed worker detection algorithms, see Fig. 4.1d. Although the experiments were conducted without the actual crane, the setup was akin. Instead of installing the camera on the crane boom, it was mounted on MiniTec profile and protruded out of the building. Likewise, the camera faced down to the ground. The ground surface is both clay/sand soil and concrete brick blocks. During the experiments, several actors, who were with and without the emergency vest, walked under the camera. Some construction tools (e.g., wheelbarrow, pipes, and barricades) were brought into the scene and worker's activities, such as squatting and working with devices, were imitated, see Fig. 4.1d.



Figure 4.10: The telescopic crane, Liebherr LTM1130, which is used in the experiments.



Figure 4.11: At the test location, Steil. On the left figure, a crane operator performs a blindlift during moving a dummy load block from position A to position B. In the right figure, CRANEBEE, a crane planning software, visualizes the planned trajectory from [73]

4.5 Crane Setup for Real Dataset Generation

The hardware setup on the crane is shown in Fig. 4.13. The camera system consists of four main components i.e., an image sensor, a wireless video transmission system, a display monitor, and a camera control unit. The last two components together with an industrial PC were located in the crane cabin, see Fig. 4.13b. The Motec crane motor zoom camera MC5200 is attached at the pendulum bracket which is mounted at the boom tip. The camera always faces down to the ground.

Regarding the transmission system, the pair of the transmitter and receiver must be calibrated or paired at the first usage. The transmitter was located nearby the zoom camera and aligned parallel to the telescopic boom and facing the sender which was attached at the first telescopic section. To send or receive the information, their antenna surfaces should be faced and aligned to each other as shown in Fig. 4.13a. The distance between them should be longer than 10 meters, otherwise, the video stream can be interfered by an overamplified signal. By the wireless video transmission system, the image frame and the zoom control command can be streamed and sent, respectively. The



Figure 4.12: These figures present a snapshot from a drone (upper row) and the crane planner software, CRANEBEE (lower row). During the experiments, the worker walked from point C to point D. The yellow spots in CRANEBEE shows the corresponding position of the annotated worker from the load-view crane camera image which is located at the top left corner. The pop-up warning message showed up due to the crane working radius violation of the worker at point D. [73]

operator can observe the video stream from the 7-inch screen monitor which was in the crane cabin. The video output is analog which contains an interlaced display.

To obtain an image from the camera to the industrial PC, we need an A/D video grabber, Logilink VG0001A. It results in a partially interlaced image after digitalization. According to VLC codec information, the image resolution is 720×576 . The image format is YUV422 which is later converted to RGB24 format. The industrial box PC, which ran the experiment under FINROC framework, is Nuvo-7160GC. The specification of the PC is Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz, 12 cores, 32GB RAM with a graphic card NVIDIA GeForce GTX 1660 Ti, 6GB.

The camera control can be executed manually by either the remote controller or sending the control command via Controller Area Network (CAN) bus protocol, namely Motec System Bus (MSB 2.0) [206]. Due to the product confidentiality, we have partial access to the camera which are 2D streaming video and basic controller function—Zoom in (O_{zi}) and Zoom out (O_{zo}) which make the objects inside the image becomes larger and smaller, respectively. The CAN pulse frequency of the zoom command can be parameterized.

4.6 From Real World to Simulation

Recently, the construction industries have started to adopt automation and robotics. The perception and control algorithms require proper validation. The test should be executed thoroughly in every possible usage condition.



Figure 4.13: Crane hardware setup. (a) Hardware setup on the telescopic crane Liebherr LTM1130. The industrial box PC including the camera controller and the 7-inch display monitor are placed inside the crane operator cabin. (b) Inside crane cabin. The figure is modified from [205].

Although experiments on the actual robot or vehicle are very important, the simulation platform is necessary, especially for construction research. The construction machines are huge. Running any experiments on a real construction site is not simple due to stringent safety regulations. One small mistake can lead to fatal accidents. To generalize top view object detection algorithms, the algorithms should be assessed in a broad range of environmental conditions. Despite the additional test location at the university, not everything can be imitated such as the crane cable or hook block.



(a) Rain

(b) Day

(c) Day-Night cycle

Figure 4.14: Sample data with different light conditions. [124]

The *simulation platform* [126] in this work is mainly developed using Unreal Engine (UE), Epic Games. UE is a photorealistic simulation tool or a game engine. It provides not only photorealism to the environment e.g., various light conditions, camera lens flare, and weather but also the dynamic movement of objects. This platform has the advantage of a more comparable environment to the actual construction including surrounding objects. The environment can be created and modified in a fast manner. Importantly, the simulation platform contains an immediate connection to the robotic framework, FINROC. By using the platform, many experiments or algorithms can be achieved concurrently without any risk and low cost.

4.6.1 Implementation Concept

In the following, the approach to transfer from reality to the simulated world will be discussed. The further detailed implementation of each component in the virtual platform, such as characters and crane, is discussed in Appendix B. The workflow of transition to the simulation can be found in Fig. 4.15.

- 1. Observation of the real world First of all, the observation in the real world had been made to create the simulated environment as close as possible. Steil site is the place where cranes ground and maintain. The color texture of the crane and environment were collected to later apply to the color of the virtual crane or other construction entities in the simulation. The texture was mapped to the surfaces of the assets in UE. Besides the observation in Steil, we additionally inspected the construction site nearby TUKL through different construction phases. The observed period of TUKL was from Jan to May 2018. The appearance and behaviors of workers-on-foot, especially those who worked under the load or around the crane, had been noted. In comparison to Steil, TUKL site is the place where the actual construction took place. However, both places were beneficial to observe because mobile cranes are generally used everywhere not limited to the construction area.
- 2. Create 3D models and map—To have such 3D models and map from the observation are arduous. Fortunately, there are many available 3D model resources that resemble the real world. For instance, instant scene properties or props can be found in UE4 Marketplace [152]. The character's asset or person can be obtained from Mixamo, Adobe Fuse CC, Renderpeople, etc. The specific map region can be selected



Figure 4.15: Workflow of simulating platform. Each number in grey circle is aligned to the implementation step number in Sec. 4.6.1, namely (1) Observation of the real world, (2) Create 3D models and maps, and (3) Skeletal animation or rigging.

and exported into a 3D map from OpenStreetMap (OSM) [207]. The virtual Steil resembles the actual site. They both are comparable not only the buildings but also the weather such as the daylight cycle and rain. To model the 3D objects, it requires additional tools in addition to UE. *Blender* is primarily a modeling and animation tool.

3. Skeletal animation or Rigging—After the 3D models are given, the models are not yet able to move as they are merely surface or mesh of the model. Thus each 3D model should have a hierarchical set of interconnected parts, see Fig. 4.17. The hierarchical set forms the skeleton, bones, or joints of the models. To animate virtual people or vehicles, the sequence of joint movements has to be assigned. Similar to the 3D model, the sequence templates are available in UE4 Marketplace and *Mixamo*. Motion capture technology is used to capture and track the real movement of human activities and turn them into character movement templates.

Once all 3D models and the map are ready, these external assets are imported to UE. All of the created assets are eventually assembled into one virtual site which resembles to the actual Steil site as shown in Fig. 4.16. The connection interface between FINROC and UE are later connected, shown in Fig. B.3. The port name list can be found in Fig. B.2. As a result, all the sensor IO and control IO can be obtained and managed through our framework.

4.6.2 Virtual Environment and Scenario

The virtual crane was placed in the related position as in the experiment at Steil, see Fig. 4.16. The landscape is filled up with buildings, vegetation, virtual workers/pedestrians,



(a) Real Steil

(b) Virtual Steil





(a) Grove GMK3060



Figure 4.17: The hierarchical set of interconnected parts shown in the white edgy cone blocks or virtual bones.

construction entities, and supplies, etc. Different atmospheric conditions are shown in Fig. 4.14. The conditions can be adjusted based on the light and rain/snow particle distribution rate such as low or high precipitation.

The day-night cycle can be applied by changing the brightness intensity in the physical light unit (lux). The position of the sun in the sky causes the various length of the shadow. The longer the shadows are, the lower the sun on the horizon and vice versa. The shadow from the sunrises and sunsets is advantageous to the assessment or data generation of the detection algorithm. The discussion of how the shadow affects the detection algorithm can be later found in one of experiments in Sec. 4.12.

Virtual characters are one of the most important simulated components because they are the targets of the top view object detection. The person's appearance is diverse e.g., the worker with a safety helmet, with/without an emergency vest, and pedestrians. The virtual workers can move likewise to the construction worker in the real world. The different postures that can be seen in the construction site such as working with the device and bending, see the second row of Fig. 4.24. The people can do talking on the phone, walking or looking around, etc. The virtual persons in the scenarios randomly move with the collision avoidance or interact with other characters. The animated movements between



Figure 4.18: UE crane all view. 1 - front view, 2 - left side view, 3 - right side view, 4 - top view, 5 - rear view, 6 - front view.

different genders can be distinguishable. After each character is properly configured, the crowd simulation can be implemented. Instead of placing characters on the map one by one, the crowd simulation generates many characters into the map at once. The character manager handles how to character move and interact with others or obstacles.

4.6.3 Virtual Hardware

Grove GMK3060 is given as a 3D telescopic crane model for the simulation. Fig. 4.18 presents different perspectives of Grove GMK3060 in UE.

Although Grove GMK3060 is smaller than Liebherr LTM1130, their functions are the same in principle e.g., hoisting, swing. In addition, this demonstrates that our simulation platform is flexible to any vehicle. This virtual crane has a basic driving functionality. It contains three crane common mechanisms as the real crane i.e., hoisting, swing, and travelling. For hoisting, the boom can extend and retract. The hook block can hoist up and down with or without attached load. Different types and sizes of load can be changed. Fig. 4.19 shows the example of an intermodal container or a shipping container as a load. The second mechanism is swing. Based on the robot coordinate system O_R , the uppercrane part can rotate around the z-axis (yaw or slew angle). The uppercrane can rotate around the y-axis or pitch angle. Finally, basic travelling can be performed such as driving forward/backward, steering and braking. The outrigger, which is an extended beam to stabilize the crane, can be applied during lifting or removed when the crane truck travels. The maneuver of the virtual crane can be manually controlled or defined as a sequence of trajectory paths. The type of hook block, cable, or load can be effortlessly changed. Any crane lifting trajectory (e.g., blind lift) therefore can be defined by Sequencer feature in UE. The example of the iterative sequence maneuvers temporally based on given

controls as shown in Fig. B.1. These features allow us to investigate the proximity between the load and worker-on-foot. In other words, it is beneficial to investigate struck-by load accidents.

For the virtual image sensor, the MC5200 motor zoom camera is implemented in the same manner as the actual camera. The virtual camera is mounted with the pendulum bracket at the tip of the boom. The physic properties in UE allow the zoom camera to have the effect of the pendulum. Therefore, the pendulum bracket can be swung by the action of gravity and acquired momentum. The zoom level can be adjusted by changing FOV. In addition, the zoom speed can be changed similarly to the Controller Area Network (CAN) pulse frequency in the actual camera. To have better observation during the testing in the simulation world, additional monocular cameras are additionally mounted on several positions e.g., operator and driver cabin. During the simulation, each view can be toggled by a keyboard input, C. Fig. 4.19 visualizes the different views from each camera. Finally, the whole virtual world setup in UE is an alternative to the real world. It can be run on any PC and connected to the proposed framework in FINROC in the same fashion.

Previously, the development of the hardware setup in the real world including the simulation platform is presented in Sec. 4.5 and 4.6, respectively. The following presents the approach on how the datasets were gathered both in the virtual world and using the actual setup, including an analysis of the synthetic data.

4.7 Data Collection From Real Environment

As mentioned in Sec. 4.4, the test took place in two locations. Steil is the industrial area in Trier where many crane activities occur, while TUKL is the building of Technische Universität Kaiserslautern where no crane involved during the data collection.

4.7.1 Data Recording

• Steil—In November 2018, the data were recorded during the afternoon between 15:00 and 17:00. As it was winter, the weather was cloudy. Three data sequences had been recorded, namely R00-S-C2, R01-S-C2, and R02-S-C2. The estimated height from the camera to the ground was 25 m. The first two sequences are relatively similar. They were recorded at the same position. A few construction workers with protective gears walked under the zoom camera. In addition, there were four to five cars parked nearby the workers. The crane hook without load was always in the camera frame. The camera was at the minimum zoom position or no zoom. On the other hand, the brightness in the last sequence is quite higher than in the previous two sequences. There were seven workers on average and all wore dark color clothes with mostly no protective gear. In this sequence, the operator performed the blind lift. At the beginning of the sequence, the rigger tied up the wooden pallet, which was used as a dummy load. After the pallet was equipped, the operator hoisted up the load and rotated the uppercrane to position B, see Fig. 4.11. While the operator was lifting, the workers walked beneath the load. The camera was zoomed in and out. The camera view partially covered the road, where a forklift or a car occasionally passed by. There were parking cars, white stones, and grey boulders lying around.



(e) Position of each camera view (a) - (c)

Figure 4.19: UE crane camera in different view

• **TUKL** —Two sequences were recorded during the summer afternoon in June 2018 and during spring noon in April 2021, namely R03-K-N1 and R04-K-N2. The camera was mounted on the constructed aluminium profiles and extended out of the building on the 5th floor and 7th floor, respectively. These sequences were recorded without a



Figure 4.20: CVAT interface. A purple BBox represents a labeled target, whose object class is *person*. On the right hand side, an object list of the current image frame shows on the side bar.



Figure 4.21: Sample public datasets which were used in this thesis. (a)Okutama[102]© IEEE 2017. (b)VCI-CITR[24]© IEEE 2019 (c)VCI-DUT[24]© IEEE 2019

crane, hence no load, hook and crane hoist cable existed. None of the workers wore protective gear. For sequence R03-K-N1, the ground surface had both clay/sand soil and concrete brick blocks. The activities of the workers included squatting, carrying construction assets, dragging an air compressor generator, etc. Moreover, the typical activities were appeared such as talking on the phone, walking. It was very sunny during the data collection. The workers stood in the position where the sun made the angle caused the shadows' length which was nearly identical to the height of the worker, see Fig. 4.26k. From another side of TUKL building, R04-K-N2 was collected during the sunny day at the parking lot. The ground surface was concrete brick blocks. In this sequence, many construction supplies were brought into the scenarios such as barricades, wheelbarrows, pallets, cable reels, and large PVC tubes. Several workers walked randomly to the area under the sun at least half an hour. Some of them wore emergency vests. A couple of them carried the pipes and pushed the wheelbarrow around.

4.7.2 Data Annotation

As discussed in Sec. 4.3, the data annotation is the important process as the data recording. To manually label the real data, is fairly tedious. Faulty BBox alignments can lead to poor learning in object detection. In this work, we followed the VOC2011 annotation protocol guideline [208] to have labeling errors as small as possible. Although the guideline is based on the frontal view object detection, most of the protocols are applicable. For example, objects in mirror should be labelled or BBox should contain all visible area of the object but not the estimated total boundary of the object. We additionally specifically defined for the labeling top view object. In particular, the worker can be labeled only when a head and a complete shoulder are visible. Consequently, the protocol ensures all annotators label the groundtruth in the same manner. Without the protocol is one of the reason that causes the annotation error. For instance, one annotator could draw a large bounding box very fit to the shape of the target.

CVAT [183] is the annotation tool used in our manual labelling. The sample of CVAT interface can be seen in Fig. 4.20. It is an open-source tool developed by Intel for annotating images and videos. The tool supports three object recognition tasks—object detection, image classification, and image segmentation. CVAT is available both locally and browserbased, however, the online portal limits the amount of the image. Furthermore, it supports many additional optional components such as TensorFlow. Many standard annotated output file formats are available e.g., TFRecord, Pascal VOC XML[209]. There is an interpolating annotation feature, which estimates the targets in-between frames. This feature is available for both video and image sequence in *Track* mode. By using this tool, the one-class annotation, which was done by an experienced person, took approximately 14-20 seconds per frame. Each frame has three objects on average.

4.7.3 Public Datasets

Two public datasets, which are referred in this work and both, were taken from UAV. All sample snapshot of public datasets can be found in Fig. 4.21. A brief description of the public datasets is as follows.

- Okutama [102] contains one object class i.e., person and is originally used for the action recognition. Although the action is pedestrian-based, it has some common activities with the construction worker e.g., carrying, pushing/pulling wheelbarrow, and lying. These datasets were captured in different heights (10-45 m) and angles (45-90 degrees) at a baseball field in Okutama, Japan. They were collected at two different times of the day, morning and noon. Ordinary actions were included e.g., drinking, walking, and carrying. The annotation is available in both detection, single and multi-label action. Only non-interpolated data was selected for training in this work. Due to original groundtruth mislocalization, the defected data is filtered out by setting the *generated* flag to false.
- Vehicle-Crowd Intraction (VCI) [24] consists of two datasets, namely VCI-CITR and VCI-DUT. These datasets contain two object categories, pedestrian and vehicle. The original authors aim to model the crowd motion under the vehicle



Figure 4.22: Overview simulation interface of collecting data using a load-view camera. \bigcirc Reprinted by permission from Springer Nature: Springer [124], Copyright (2020).-check again rgd copyright

influence. While recording the data, the drone hovered above the ground plane with no change in altitude. The video was stabilized to remove the oscillation of drone motion. The camera was always faced down to the ground. In other words, the camera depression angle is 90 degrees. VCI-CITR was collected at a parking lot near the facility of Control and Intelligent Transportation Research (CITR) Lab at The Ohio State University (OSU) in the United States. A golf cart was used as a small vehicle to drive through the group of eight people. VCI-DUT was collected at the campus of Dalian University of Technology (DUT) in China. The area includes a pedestrian intersection crosswalk and roundabout. Most of the pedestrians were students. The density of the pedestrian in the scene was higher than VCI-CITR. There were more than 20 students who just came out of classes. The original pixel groundtruth of VCI dataset provides only center point format (c_x, c_y) not BBox format (x, y, w, h). To have the BBox format for training object detection, we estimated the size of the box based on the center point and expanded the point to 30×30 pixel BBox. The object class *vehicle* was removed as it is not our goal target.

4.8 Synthetic Data Generation

To generate the synthetic data, we employed the simulated platform developed in Sec. 4.6.

4.8.1 Data Recording

After the simulation environment was set up, we started to collect the data. The data collection user interface (UI) in Fig. 4.22 allows us to observe what is the possible risks

associated with the crane and the current movement of the crane. The interface allows us to observe what is the possible risks associated with the crane and the current movement of the crane. In addition, this ensures a clean data i.e., neither purely background image nor only negative samples for training. The inteface is built by using Unreal Motion Graphics UI Designer (UMG). It is a tool that can be used to create user interface elements presenting to users such as in-game menus. Similarly, we created to the inteface to facilitate our data recording. As shown in Fig. 4.22, the overall crane at Steil is visualized. There are several control buttons and displays of other camera angles including the crane information. From the bottom left of Fig. 4.22, the small figure shows a current load-view camera. The following three buttons are the possible annotation options, namely, $start_seq_00$, record qt, and snapt qt. For start seq 00, the simulation executes the defined vehicle control sequence including the camera snap control based on the given timeline. record qtstarts after the option is selected and ends until simulation terminated. *snapt* qt allows us to take a camera shot at any desired time. Lastly, the bottom right figure shows the top view perspective of the crane. The white texts on top-right display current states of crane control. In the end, the raw annotation output consists of three items —an original RGB image, segmented image, and a comma-separated values (CSV) file, see Fig. 4.23.

At the virtual Steil, there were five datasets collected, namely U00-S-C0, U01-S-C1, ..., U04-S-C4. The number at the end of the dataset indicates how many crane telescopic sections are extended during the record of each sequence. For example, none of the boom sections in U00-S-C0 was extended while all four sections were fully extended in U04-S-C4. The crane lift angle or pitch was approximately between 45 and 50 degrees. While recording, the slew angle was kept turning around for 360 degrees. The hook and the hoisting cable were in the camera view. The weather was gradually changed from day to night or sometimes rain. The virtual characters were spawned into the virtual world. The workers and pedestrians walked around both inside and outside the barricades. Similar to the real world, workers were working with devices, pushing the wheelbarrow, driving the forklift or the truck. U06-S-N2 and U07-S-N2 were recorded in the same manner as U00-S-C0, ..., U04-S-C4, but the ground plane was fully asphalt and the lift angle is 30 degrees.

As the usage of the crane is not limited to the construction, we additionally collected the data from the Urban City environment, which is available in UE4 Marketplace. The dataset U05-C-N2 was collected without crane. A down-facing camera was fixed at one position with its depression angle of 90 degrees. The people arbitrary walked on the road and pedestrian path.

4.8.2 Data Annotation

We exploited the post-processing feature of UE into our implementation. The original image is labeled at pixel level. The target object in the annotated image, which has the same color, belongs to the same object class and identity. Both the original image and its annotation are then converted into a proper data format and used for learning to get the optimal weight. We adopted the annotation calculation in [195]. It processes the raw annotation outputs and later draws bounding boxes for each object. The final annotation data are converted into PASCAL VOC in the XML file. Regarding having clean data annotation, human verification can be quickly done by using labeling tools. After generated data is verified, we can now use this data for training object detection



Figure 4.23: Groundtruth generation outputs. (a) and (b) show RGB images with virtual actors from load-view camera while (c) and (d) show the groundtruths which correspond to the second row.[©] Reprinted by permission from Springer Nature: Springer [126], Copyright (2020).

using deep learning. This XML file can be accessible for further training object detection by many conventional deep learning frameworks such as TensorFlow [210]. In general, our system recorded and annotated the data with a speed of 0.5 seconds per frame with no limitation of object number or class.

4.9 Dataset Summary

Table 4.3 lists all the datasets which were used in this thesis. The list summarizes both synthetic and real data. There are nearly 177K target boxes in total i.e., ~69K real target boxes and ~108K synthetic target boxes. The dataset naming convention can be found in Fig. 4.25. Generally, the data source (S) identifier denotes where the data was captured and will be either U for UE or R for real world. The sequence number (##) identifies dataset index in each individual source. The data recording location (L) can be City (C), Steil (S), TUKL (K). The vehicle platform identifier (V) can be either with crane (C) or without crane (N) as mentioned in Sec. 4.7.1. Finally, the last digit (#) represents the altitude ranges between the camera and the ground. The ranges were categorized based on the number of the extended boom section. In other words, the bigger number the digit is, the longer the distance between the camera and the ground is. There are five ranges i.e., 0 (0-12m), 1 (13-19m), 2 (20-26m), 3 (27-33m), and 4 (higher than 33m).

(d)



row (d-f) is RGB images visualized the bounding boxes of each object from a generated annotation file. Activity in (a),(d) is bending. Activity in (b),(e) is carrying an object. Activity in (c),(f) is working with the device. © Reprinted by permission from Springer Nature: Springer [124], Copyright (2020).

(e)

Figure 4.24: Sample annotation output. The first row (a-c) is segmented images and the second



Figure 4.25: Dataset naming convention.

4.10 Evaluation Metrics for Object Detection Algorithms

Evaluation metric is a measurement to assess how good an object detection algorithm to satisfy the user's requirements is or how much it differs from the others in various aspects among literature such as speed or accuracy. In other words, choosing detectors depends on many factors e.g., an application usages and capability of existing hardwares. For

(f)



(d) U03-S-C3



(b) U01-S-C1



(c) U02-S-C2



(f) U05-C-N2



(g) U10-S-C4



(e) U04-S-C4

(h) U11-S-C4



(j) R02-S-C2

(k) R03-K-N1



(m) R10-S-C2

(n) R14-K-N2

(o) R16-S-CX

Figure 4.26: Sample datasets for worker detection [118]. The detail of each dataset can be found in Table 4.3.

Seq name	Frames	Resolution	$D_{cam}(\mathbf{m})$	Collected from crane	Average object instances per frame	Total object instances	Snapshot figure
1100 S C0	120	1600 × 1200	19		2	283	4 265
U01 S C1	200	1600×1200	12	•	2	265	4.20a 4.26b
U02-S-C2	303	1600×1200 1600×1200	15 26	v ./	5	1636	4.200 4.26c
U03-S-C3	501	1600×1200 1600×1200	20 33	•	9	4463	4.26d
U04-S-C4	1110	1600×1200	39		8	8448	4.26e
U05-C-N2 [211]	10231	800×600	20	x	8	81302	4.26f
U10-S-C4	1083	1600×1200	40		8	8656	4.26g
U11-S-C4	363	1600×1200	40	1	8	2974	4.26h
R00-S-C2	713	720×480	25	1	3	2139	4.26i
R01-S-C2	264	720×480	25	1	3	592	similar to $4.26i$
R02-S-C2	4073	720×480	25	1	5	22138	4.26j
R03-K-N1	9748	720×480	19	×	2	22013	4.26k
R06-K-N2	1382	720×576	25	X	4	5497	4.261
R10-S-C2	642	720×480	25	1	1	777	4.26m
R14-K-N2	500	720×480	25	X	3	1500	4.26n
R16-S-CX	3829	720×480	25	1	4	14330	4.260

Table 4.3: Dataset summary. The snapshot of dataset can be found in Fig. 4.26.

example, Detector A can provide nearly 100% accuracy, however it requires very powerful computation resource to be able to execute in real time. On the other hand, Detector B is able to execute in real time with 80% accuracy. If an end-user does not have such hardware resource, Detector B is the option.

Examples of the object detection evaluation metrics, which are used in literature or competitions [209, 212, 119] are Average Precision (AP), mean average precision (mAP), average recall (AR) and mean average recall (mAR), etc.

In this thesis, we adopted AP metric. Particularly, our detection evaluation metric is adopted from PASCAL Challenge [209] including its evaluation MATLAB toolbox from [213]. The basic concepts which are parts of the metric calculation e.g., confidence score, IoU are first described before the discussion of the metric.

- *Confidence score*—is the probability that a bounding box contains an object. It is usually assigned by a classifier.
- Intersection over Union (IoU)—is an overlap criteria to determine whether a detection is considered correct or not. The definition of IoU is identical as the Jaccard similarity index in statistics, see Eq. 4.1. This concept was introduced by Everingham et al. [214]. For the detection task, a detector returns a list of bounding boxes and their associated confidence score with respect to an image. Each detection result is assigned to a ground truth object and evaluated to be either true positive (TP) or false positive (FP) by calculating IoU or the overlapped area between the ground truth B_g and the predicted detection B_x . The IoU is a score to determine how much the detection closes to the ground truth. The ratio must be higher than a certain threshold to consider as TP. According to [214], we followed the default threshold which is 0.5 or 50%. The depiction of the overlap terminology can be found in

Fig. 4.27. In PASCAL challenge, the additional rules are defined for TP and FP. When there are multiple predictions correspond to the same ground truth, only the prediction with highest confidence score counts as TP. The rest predicted detection are considered as FP.

$$IoU = \frac{Area \ of \ Intersection}{Area \ of \ Union} = \frac{B_x \cap B_g}{B_x \cup B_g}$$
(4.1)

where B_x is predicted bounding box and B_g is ground truth bounding box.

• *Precision and Recall*—Precision in Eq. 4.2 is positive predictive value (PPV) or the ratio of correct positive detected cases to all the predicted postives cases, which is the sum of TP and FP. Recall or sensitivity in Eq. 4.3 is true positive rate (TPR) or the ratio of the correct positive detected cases to all actual positive cases, which is the sum of TP and FN.

$$precision = \frac{TP}{TP + FP} \tag{4.2}$$

$$recall = \frac{TP}{TP + FN}$$
(4.3)

By assigning the threshold for the confidence score at different levels, pairs of precision and recall can be obtained. A Precision-Recall (PR) curve can be plotted with precision on the y-axis and recall on the x-axis. The PR curve visualizes the relationship between the two metrics. It indicates the trade-off between the TPR and the PPV for a predictive model using different probability thresholds.

4.10.1 Average Precision (AP)

Average Precision (AP) is widely used in measuring the accuracy among object detectors. The AP addresses the problem of selecting the best performance detector of PR-curves when there are many curves intersect with each other. AP is a numerical metric, therefore, it is easier to compare. Basically, AP is finding area under the PR curve or AUC. For the PASCAL VOC challenge, the 11-point interpolated AP is calculated. The recall value is divided into 11 points r under the value from 0 to 1.0—0, 0.1, 0.2, ..., 1.0.

$$AP = \frac{1}{11} \sum_{r \in \{0,0.1,\dots,1\}} AP_r$$
(4.4)

When there is a multiclass detector or a detector which recognizes more than one object class (i.e., M > 1), the mean of the AP across all M classes is defined by *mean Average Precision* (mAP) in Eq. 4.5, where M is total object classes. In our work, there is only one single target class M = 1 (person); thus, mAP and AP are equivalent.

$$mAP = \frac{\sum_{i=1}^{M} AP_i}{M} \tag{4.5}$$


(e) TN

Figure 4.27: Observation type and its terminology for overlap criteria. The green bounding box is denoted as ground truth g while the red bounding box is denoted as detected hypothesis x. (a) TP - True positive or correct detection. (b) FN - False negative or miss detection. (c) FP - False positive or false alarm. (d) FP and FN. (e) TN - True negative or correct rejection, which is neither the ground truth nor predicted detection exists. In general, this value is not considered in object detection algorithms.

4.11 Analysis of Real and Synthetic Data in Frequency Domain

Frequency domain study of the synthetic data captures the unrealistic features of these images by analyzing the local rate of changes of pixels. It can highlight edges and noise as



Figure 4.28: (a) $MS(I_R)$ is a magnitude spectrum summation of real world image dataset. The second column, (b) and (c), are a magnitude spectrum summation of raw synthetic dataset, $MS(I_{U,nBL})$ and a magnitude spectrum summation of preprocessed synthetic dataset, $MS(I_{U,BL})$, respectively. In the the third column, (d) $\Delta MS(I_R, I_{U,nBL})$ is a magnitude spectrum difference between real world and raw synthetic dataset. (e) $\Delta MS(I_R, I_{U,BL})$ is a magnitude spectrum difference between real world and preprocessed synthetic dataset. The intensity of the color represents the magnitude of Fourier transform, while u and v are the frequencies along m and n, respectively.

high-frequency contents which appears on-surround region of magnitude spectrum. De and Masilamani[215] proposed a measurement method to assess image sharpness and blurriness in frequency domain.

As discussed earlier, we have applied average filter on synthetic data during preprocessing. This filter makes the image blurred by applying low pass filter which attenuates the magnitude of high-frequency components of the frequency spectrum. As an example, frequency spectrum of a grayscale image resembles Fig. 4.28(a). The high frequency components are located on outer diameters and by moving toward center the frequency reduces such that the center point captures the DC value of the image (average over all pixels).

To examine an image in frequency domain, we first apply 2D Discrete Fourier Transform $(DFT)^1$ to an individual greyscale image²:

$$F[u,v] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[m,n] e^{-i2\pi u m/M} e^{-i2\pi v n/N},$$
(4.6)

 $^{{}^{1}}F[u,v]$ denotes a 2D Discrete Fourier Transform (DFT) of f[m,n]

 $^{{}^{2}}f[m,n]$ denotes a greyscale image in the spatial domain.

where $u = 0 \dots M - 1$ and $v = 0 \dots N - 1$. *m* and *n* are spatial coordinates, while *u* and *v* are frequencies along *m* and *n* respectively. The result of Fourier transform consists of two components, magnitude and phase. For this analysis, we consider only the magnitude.

We apply Fourier transform on real world image dataset (I_R) , raw synthetic dataset $(I_{U,nBL})$, and preprocessed synthetic dataset $(I_{U,BL})$. The sample result of Magnitude Spectrum (MS) of each image sequence can be found in Fig. E.1 of Appendix E. After Fourier transform of each image dataset I_R , $I_{U,nBL}$, and $I_{U,BL}$ is individually calculated, MS of each dataset is accumulated to later determine the differences. Using I_R as a reference, we then find the absolute MS difference on two pairs i.e., $(I_R, I_{U,nBL})$ and $(I_R, I_{U,BL})$, listed in the following Eq. 4.7 and Eq. 4.8. Before determining the difference, all MSs should be normalized.

$$\Delta MS(I_R, I_{U,nBL}) = |MS(I_R) - MS(I_{U,nBL})| = \Big| \sum_{k=0}^{K-1} log(|F_R^k[u, v]|) - \sum_{l=0}^{L-1} log(|F_{U,nBL}^l[u, v]|) \Big|,$$
(4.7)

$$\Delta MS(I_R, I_{U,BL}) = |MS(I_R) - MS(I_{U,BL})| = \left| \sum_{k=0}^{K-1} log(|F_R^k[u, v]|) - \sum_{l=0}^{L-1} log(|F_{U,BL}^l[u, v]|) \right|,$$
(4.8)

where K is total number of real world dataset (I_R) and L is total number of synthetic training dataset (I_U) .

The results of the magnitude spectrum differences are depicted in Fig. 4.28. F(0,0) indicates the DC-component or zero frequency of the image which corresponds to the average brightness, and F(M-1, N-1) represents the highest frequency.

 $MS(I_R)$ in Fig. 4.28a has only strong edge along the high-range frequencies and contains mostly low-frequency magnitudes. Most dominant edges of the building, road, or cars in (I_R) correspond to the bright lines in the Fourier transform, which can be seen in the first row of Fig. 4.29 illustrating a snapshot of real image sequence R00-S-C2. The cars and crane cable in the figure appears as oblique lines in the magnitude spectrum, see Fig. 4.29e and 4.29f.

In contrast, $MS(I_{U,nBL})$ in Fig. 4.28b gets various large values of the magnitude even for higher frequencies which scatter in the further off-center region, see Fig. 4.29b. As a result, the magnitude spectrum difference between I_R and $I_{U,nBL}$ in Fig. 4.28d tends to have more high-frequency components compared to the real-preprocessed MS difference in Fig. 4.28e. Unlike the raw synthetic images, $MS(I_{U,BL})$, see Fig. 4.28c, is comparable to the real world dataset.

The frequency analysis proves that $I_{U,BL}$ shares common characteristic with I_R . To quantitatively describe the MS difference instead of the visualization, Root Mean Square Error (RMSE) of each Δ MS is calculated. The RMSE of $MS(I_R, I_{U,nBL})$ is 0.143, while the RMSE of $MS(I_R, I_{U,BL})$ is merely 0.072. The frequency characteristics of the image have a prominent influence on the detection rate. Our results clearly provides evidence to further develop frequency tests and complex frequency-domain filters for the synthetic data. In fact, these filters can be integrated in the simulation engines to obtain more realistic dataset.



Figure 4.29: Examples of Magnitude Spectrum (MS) on sample images. The first column is the snapshots of each original image dataset. The second column depicts MS of the raw image of the previous column. The last column shows MS of the filtered images.

4.12 Experiments

As mentioned in Sec. 4.3, the simulation platform supports data acquisition of the object detection using a deep learning algorithms, which requires a large amount of data. Many research domains, such as autonomous passenger vehicles, have exploited the simulated data to yield the best detection accuracy. However, there is not much evidence in the construction research area including the validation of simulated data usage.

The objective of the experiments is to provide the findings of how the simulated data impacts upon the top view object detection using deep learning algorithms in construction environment. In other words, we explored the quality of the synthetic data whether it can be further adopted to improve the quality of the data-driven top view detection. Given the synthetic datasets which were generated by our system, two experiments were conducted to accomplish the defined goal. The first experiment is to explore the pre-trained UAV network model directly on the crane load-view data. The association between two different top view images, which are aerial and crane load-view were investigated. As discussed in Sec. 2.2.2, there is no public pre-trained model including dataset from load-view crane camera. Nevertheless, the crane load-view and the aerial view are nearly identical. Regarding the data deficit situation, the second experiment analysed whether the synthetic data can completely replace the real world data in training of deep learning object detection. Overall experimental workflow can be found in Fig. 4.30. Basically, we addressed the problem twofold, network model (Experiment 1) and synthetic data (Experiment 2).

- *Experiment 1*: Can UAV *network model*, which was trained with comparable dataset, be immediately used as the alternative detector for the crane load view object detection ?
- *Experiment 2*: Can real data be completely replaced by *synthetic data* in training process of top view object detection ?

According to the requirement of visibility assistance, the operator should be warned about the nearby object in order to recognize the accident risk in (near) real-time. In these two experiments, SSD based detectors were then chosen. SSD is a one-stage detector and introduced to address the problem of multi-scales. As described in Sec. 5.1, SSD architecture model is based on VGGNet [82]. Unlike the two-stage detectors such as Faster R-CNN [88], SSD is relatively fast, but less accuracy [84]. Therefore, SSD detectors provide a good trade-off between speed and accuracy. Experiment 1 and Experiment 2 were conducted in Sec. 4.12.1 and Sec. 4.12.2 respectively. The detail of evaluation metrics for object detection algorithms in all experiments i.e., AP can be found in Sec. 4.10.

4.12.1 Exploration Aerial Image DL model With Crane Load-View Data

To the author's knowledge, there is no pre-trained model for top view object detection from the crane, which is available publicly. Unlike UAV model, they are non-ubiquitous. As mentioned in Sec. 2.2.2, the background of UAV training dataset is much plainer, in constrast to the load-view camera in construction area which is relatively complex.



Figure 4.30: Experiment workflows. (a) illustrates the workflow of the first experiment discussed in Sec. 4.12.1. (b) shows the workflow of the second experiment discussed in Sec. 4.12.2.

However, the camera perspective of UAV data is quite similar to the crane load view. In general, the data from UAV shares most in common similarity with the load-view data in relative to other applications e.g., frontal view in Autonomous Vehicle (AV).

The workflow of this experiment can be found in Fig. 4.30a as Experiment 1. First, we directly adopted the original UAV pre-trained SSD model from [102], or SSD-OkuPed for short. The authors [102] retrained the model with Okutama dataset, pedestrian action from aerial view. Unlike other UAV datasets, the dataset is relatively similar to the load-view crane camera as discussed in Sec. 4.7.3. The AP of the model, which the authors originally evaluated on the Okutama dataset is 72.3%. The authors made the remark when the altitude is higher than 30 meters, the model performed poorly because the pedestrians were too small. Finally, we passed our four load view data sequences, two synthetic sets, and two real sets to the model. The hardware used in experiments is Intel(R) Xeon(R) Gold 6126 CPU, 2.60GHz, 48 cores, 188G memory.

Seq name	AP@0.5 (%)	Average inference time (ms per frame)
U06-S-N2	49.03	217.24
U07-S-N2	39.24	217.33
R00-S-C2	75.23	206.52
R03-K-N1	12.0	208.08

Table 4.4: The detection results of Experiment 1.



(c) R00-S-C2

(d) R03-K-N1

Figure 4.31: Sample of detected results. The green bounding box is the detected target with class and confidence label. © Reprinted by permission from Springer Nature: Springer [124], Copyright (2020).

Fig. 4.31 shows selected frames from the detector. Table 4.4 reports AP of each sequence. For the synthetic dataset, the detector results in fair performance. The detector is able to achieve AP of 49.03% and 39.24% for the synthetic data, U06-S-N2 and U07-S-N2. As mentioned earlier in Sec. 4.7.3, Okutama dataset which is used in the pre-trained model has some common activities to the worker. The detector, therefore, produced a correlative result with the load view.

For further analysis of the real dataset, the detector achieved a good result in sequence R00-S-C2 with 75.23% AP. On the contratry, the detector in sequence R03-K-N1 suffered from the shadow of the object which is strongly akin to a person dimension itself, see Fig. 4.31d. It is a challenging sequence because it was very sunny. Regarding the small angle between the sun and the targets, this results in a short shadow. The detector considered the body and its shadow as one single object. Moreover, the shadow shape looked similar to the *lying* person, which is one of the activity class in the Okutama data. Consequently, this created a lot of false positives. Despite the fact that this SSD detector precisely detected a person with the bigger bounding box, this causes IoU to become extremely low and eventually lower AP. The difference of inference time between sequence U and sequence R is derived from the larger image size in sequence U whose takes longer to process.

In conclusion, there were common features between aerial view and load view image such as standing pose. On the other hand, it failed when there was short shadow or the workers bent their back or crouched, whose activities do not exist in pedestrians. The pretrained aerial image detector was able to generally sufficient to produce average results. However, the detector unfortunately could not apply to the load view image in direct manner for the construction environment. The aerial image sequences mostly occupy with plain background and not sunny. To achieve better result, it is recommended to do transfer learning.

4.12.2 Investigation of Synthetic Data Replacement in Training a NN Model

The objective of the experiment is to investigate the performance of the deep learning object detector which was trained by only the synthetic data. In particular, we wanted to examine if the virtual data can support or completely replace the real data in the training process. The experimental workflow can be found in Fig. 4.30b. Five synthetic datasets, U00-S-C0, ..., U04-S-C4 were first generated. Instead of training the deep learning detector from scratch, the transfer learning was used to accelerate the training process. Once the training ended, the actual crane load view images were fed to the detector for the evaluation.

In the experiment, we selected the SSD based detector [97]. It is introduced to handle objects in different scales and accurately localize dense objects. To create the synthetic data closely resembling the target dataset (i.e., crane load-view images), the synthetic data were preprocessed by image filtering. We noticed that the target images have more motion blur than the training samples because they tend to come from the swing movement of the camera, the vibration of the machine, or the video interlace. In order to close the domain gap as discussed in Sec. 4.3.1, the motion blur was added to the synthetic data. In practice, the averaging filter K with the kernel size of 10×10 was applied to all simulated data to blur the images. The normalized box filter is shown in Eq. 4.9, where J_{10} is an 10×10 matrix of ones. The size of box filter was chosen by trial and error. The original synthetic datasets are denoted as U00-S-C0, ..., U04-S-C4 and the blurred datasets are denoted (U00-S-C0)', ..., (U04-S-C4)'. Thus the blurry image is close enough to the actual distribution.

$$K = \frac{1}{100} \cdot J_{10} \tag{4.9}$$

The ResNet-50 model was used as a backbone network. We initialized our weights from a pre-trained checkpoint of the MS COCO dataset [212]. All synthetic data, (U00-S-C0)', ..., (U04-S-C4)' are combined and randomly shuffled into training and development sets. The train set and the development set consist of 10907 and 4675 objects respectively. The test set with 4934 objects are from R00-S-C2 and R02-S-C2. The network was trained until the optimal point with a learning rate of 1e - 7. The sizes of anchors were set to {32, 64, 128, 256, 512} and the strides to {8, 16, 32, 64, 128}. The hardware used in detection experiments is NVIDIA GeForce GTX 1060, 3GB GDDR5.

We conducted two main trials. In the first trial (BL), we trained the network with the *BLurred* images, $(U00-S-C0)', \ldots, (U04-S-C4)'$, while the second trial (nBL) trained the *non-BLurred* images, $U00-S-C0, \ldots, U04-S-C4$. In each trial, we validated the network with two test sets, R00-S-C2 and R02-S-C2.



Figure 4.32: Precision-Recall (PR) curves of the experiments. AP in Table 4.5 can be achieved by the approximation of area under PR curve. [118]



Figure 4.33: Predicted results of trial BL on the test sequence R00-S-C2 in the first row (frame 30, 219, 632) and R02-S-C2 in the second row (frame 40, 297, 348). The blue BBox is the detected target with confidence score label while the green box is groundtruth. [118].

Our detection evaluation metric is adopted from PASCAL Challenge [209] with IoU threshold of 0.5. Fig. 4.33 presents several predicted frames from both test sets. APs of the trials are listed in Table 4.5. The AP is obtained by the approximation of areas under PR curve. The PR curves of experiments are shown in Fig. 4.32.

Trial	Test seq name	AP@0.5 (%)	Average inference time (ms per frame)
BL - All	R00-S-C2,R02-S-C2	66.84	-
BL - R0	R00-S-C2	78.10	150.0
BL-R2	R02-S-C2	50.10	152.7
nBL - All	R00-S-C2,R02-S-C2	53.13	-
nBL-R0	R00-S-C2	78.20	155.6
nBL-R2	R02-S-C2	38.26	151.7

Table 4.5: Results of AP on each dataset. The performance gain between preprocessed training images (green font) and original images (red font) is approximately 10%.



Figure 4.34: Comparison of the top view perspective between load-view camera (left) and drone camera (right). The identity of each object in both images is defined by the same number tag in the scenario. Number 1 is a rock border next to the fence. Number 2 is two yellow emergency vests hanging on the fence. Number 3 is two road manholes. Number 4-5 are cars. [118]

First, we evaluated the networks, which are trained with blurred and non-blurred images on the test sequence R00-S-C2. Both of them, BL - R0 and nBL - R0, yield nearly the same results (AP \approx 78%). The workers in the sequence most often can be recognized by both networks. Despite the low-light condition, the workers were wearing the high-visibility color vest and hard helmet which can be visible to the networks.

Afterward, we assessed the second test sequence R02-S-C2 for the trial BL - R2 and nBL - R2. The detector trained with blurred images, BL - R2, shows a positive outcome. As a result, the overall AP of the network is higher when trained with the blurred datasets $(U00-S-C0)', \ldots, (U04-S-C4)')$, compared to the non-blurred ones $(U00-S-C0, \ldots, U04-S-C4)$, check the AP values for trial BL - All and nBL - All in Table 4.5. The difference in the average predicting times among trails is negligible.

In fact, R02-S-C2 is a difficult sequence. It is recorded in higher elevations and thus it is hard to recognize the worker. Fig. 4.34 shows the comparison of the same objects from two different camera angles. Apparently, the white rocks (number 1) and manholes (number 3) are almost identical to the person wearing the safety helmet. The workers' appearance forms a similar color and shapes view as of the ground. For the yellow emergency vest, we

notice that the load-view camera is unable to reproduce the same color as shown in the drone camera or being visible to the human eye. Instead, it displays as white pixels, see Fig. 4.34. This could be caused by the variant brightness, low image resolution, etc. In addition to the issue of the traditional detectors using only PPE color features mentioned in Sec. 2.1.2, color inaccuracy shown in the load-view camera can worsen these detectors because those color feature ranges are normally predefined. These negative samples can likely lure the human to misjudge as well as the detector.

Furthermore, we had prior experience in training the load-view worker detector with merely UAV data whose detail is not included in this work or likewise in the experiment of Sec. 4.12.1 where UAV detector was employed to detect worker from the load-view camera. The drone data were initially expected to be used as an alternative to augment the training dataset for load-view worker detection. The prediction results were quite unsatisfactory. Evidently, the workers in the drone camera in Fig. 4.34 can be seen fully while only the heads and shoulders of the workers in the load-view camera are visible.

In conclusion, there is a significant improvement of 11.71% in average precision between the pre-processing and non-preprocessing synthetic dataset. Using artificial data to train a DNN model is beneficial. The model acquires the image features and is able to yield good performance without seeing none of the real-world data. Nevertheless, they can not completely replace the real data, but can be used as a supplement. Synthesizing the artifical data remains challenging. Sometimes, it is easier to create the synthetic data that appears realistic to a human than creating the data that appears realistic to the learning algorithms [216].

4.13 Discussion

In this chapter, we introduced the setup of the automatic safety monitoring system as the platform to experiment with our proposed top view object detection algorithms both in real and virtual world. The system can increase situational awareness of the operator, especially blind lift. In the real world platform, we chose the typical scenario i.e., blind lift, in the industrial area *Steil* which appears to have many crane activities take place. Furthermore, the experiments were performed under expert guidance and advice. On the other hand, the simulation platform strongly supports research and development, especially in the construction domain for the following reasons.

First, it shortens the development time. Multiple researchers can concurrently work on different parts under the same platform. To conduct specific experiments such as high brightness or under the rain, the experiments have to be always prepared and standby to achieve such states. Because in the real world, the circumstances e.g., weather, light condition are beyond human control. On the other hand, this can be conveniently done by the configuration change in the UE. Any new vehicle or environment can be built up in a fast manner according to the workflow in Fig. 4.15. Second, it prevents the chance of accidents or incidents. To prepare the experiments, the sensors had to be mounted and unmounted many times on the huge machine. Driving a crane mandates a certified or trained operator. On the contrary, the complete experiments can be carried out on a single PC using the simulation platform. Lastly, it costs less expense. The development time speeds up. The real crane is unnecessary to rent.

Nonetheless, the question arises of how close the simulated environment to the real world is. Eventually, not all simulated data always look identical to the real world, even to the computer [216]. People dedicated themselves to travel over the world and capture the texture assets from the entire real environment and bring them into the simulated world [217]. In addition, there is a big activity in the research domain of bridging the gap between real and virtual world [204].

Later in Sec. 4.3, we presented how the datasets were collected from both the real world and the virtual world using the proposed platforms described in Sec. 4.5 and 4.6. The real-world data were labeled manually, while the groundtruth of the synthetic data were automatically produced via the developed simulation platform. The scenerio in the virtual world was physically based modelling or hand modelled scenes that visually follows those physic laws without mathematical formulation. In particular, we modelled the simulated platfrom from observation in the real construction area in Trier and applied real-time rendering for differnt light transportion which allowed us to collect diverse weather scenarios.

Regarding the data deficit in the construction domain, we generated diverse data sequences to later improve the quality of the deep learning object detection algorithms. Several data sequences were collected from the real world. More than five synthetic sequences were produced based on the observation in the real construction areas including the discussion with the experts in mobile crane operation. The platform provides the researchers a place where they can independently investigate and analyze their algorithms and design. Also, it reduces time of the development or system optimization or adjustment. For the dataset generation, it is sufficient to point out that the simulation platform shortens both data collection and data annotation time. As discussed in Sec. 4.7.2 and 4.8.2, the virtual platform is able to generate the annotation 280-400 times faster than the hand labelling method. Hence, it significantly speeds up the system delivery time. The last important remark is that it reduces the probability of accident occurrence as there is less contact or interaction between the actual vehicle and the human.

Finally, two experiments were conducted to assess if the simulated data can be exploited to improve the top-view detector. In the first experiment in Sec. 4.12.1, the UAV pretrained SSD model indicated the common features to the targets from the load-view crane camera. This leads to the starting point to use this pre-trained network as the initial weight for the fine-tuning process of our top view detection. While the second experiment in Sec. 4.12.2 purely used the synthetic data to train SSD based detector and test with the real load-view data afterward. There are two networks trained for evaluation. The first network was trained with preprocessed images and the second was trained with the primitive images. The synthetic data were preprocessed to make them comparable to the real data. In the end, the detector ran on the two test sequences that were taken from the real crane. Consequently, blurred virtual data appeared to make data more realistic to the learning algorithm with the improvement of more than 10% accuracy relative to non-preprocessed data.

In the following chapter, the choice of network including suitable configuration for the top view object detection from load-view crane camera will be diagnosed. Additionally, the generated datasets will be employed in the training process of the chosen network model for optimization.

5. Top View Object Detection Using Deep Learning

Detecting workers from the load-view is challenging. One of the important requirements of having effective deep learning detectors is a great amount of high-quality data. In particular, it does not only require a high number of data, but the data should also contain correct ground truth and be relevant to the goal application. In the previous chapter, the load-view dataset generation in a fast manner and the validation of the synthetic data with the deep learning algorithms were presented. Consequently, synthetic data is able to be employed for data augmentation to improve the detectors. Besides the data augmentation, there are many other approaches that can enhance the quality of the detection using deep learning algorithms such as parameter configuration.

The goal of this chapter is first to investigate the trade-off between two standard types of deep learning object detectors for worker detection from the load-view crane camera and determine the network model which is competent for the target application. The second is to find suitable parameters which are applicable for the chosen network and the hardware setup.

The structure of the chapter is as follows. Sec. 5.1 presents the detail of the specific network architecture models which are widely adopted and their development. The analysis of the collected dataset was examined in Sec. 5.2. Later, the candidate list of network models in Sec. 5.3 is discussed based on the architecture and the data statistic from the earlier section. The network configuration and selection are later presented in Sec. 5.4.

5.1 Deep Learning-Based Detection Techniques

As mentioned in Sec. 2.2, object detection using deep learning approach has became notorious. Several network architecture models which were commonly used in object detection are listed. In this section, we will further discuss the networks in detail including previous studies, especially in top view object detection in construction domain or similar. The well-known DL-based object detector algorithms can be categorized into two classes, namely two-stage detector and one-stage detector (see, Fig. 5.1).

Two-stage detectors:

- **R-CNN** [86] used the *Selective Search* as an object proposal method. Nearly 2000 Region of Interests (RoIs) with the size of 224×224 are fed into CNN based feature extraction. At the end, SVM predicts and assigns the class and the probability for each RoI. Although R-CNN is first notably of using the CNN in object detection, a speed bottleneck is the CNN feature extraction of ≈2k region proposals.
- Fast R-CNN [87] was able to reduce the overall training time while improving the accuracy compared to R-CNN. RoI pooling layer, which is a type of max-pooling, is added. Given different sizes of input images, this network pooling layer allows creating feature maps in varying scales. Instead of SVM classifier, BBox regressor is employed to predict localization boxes. As can be seen, Fast R-CNN network contains two network branches, a classification which indicates an object class and regression which gives object coordinate.
- Faster R-CNN [88] replaced Selective Search to a small ConvNets, called Region Proposal Network (RPN) as depicted in Fig. 5.1a. Using Selective Search for candidate generation is cumbersome because it is the slowest part of Fast R-CNN. Faster R-CNN used RPN to perform candidate search instead. Anchor boxes, which is a set of BBoxes, is introduced as an input to train the RPN. The boxes are variant in aspect ratios and size. RPN only distinguishes the object from the background but not the object class. The aim of the RPN is therefore to indicate the BBoxes which are close to the position of the ground truth object as much as possible. Finally, the BBoxes are passed to the pooling layer like Fast R-CNN. With comparable accuracy in the VOC-2007 dataset, Faster R-CNN is 10 times faster than Fast R-CNN.
- Mask R-CNN [89] is an object detector based on instance segmentation. The architecture is built upon Faster R-CNN. Likewise, it has two stages, candidate generation, and classification. The main difference is an additional masking network branch which pixel-to-pixel masks the object based on the first stage proposal. As the result, the pixel-level masking refines the BBox and provides better alignment of the object.

One-stage detectors:

- YOLO [90] passed an input image to CNN only once at runtime, hence the name, You Only Look Once (YOLO). The method contains only a single NN which directly predicts BBoxes location, a confidence score and a likelihood of object class. The network breaks down the input image into square grids as shown in Fig. 5.1b. Each grid predicts a limited number of BBoxes. The grid cell handles only objects whose center lies on that cell. Despite speed outperformance, one major drawback of this approach is detecting very small objects. Moreover, each grid can only predict a single object class. In other words, if there is more than one object in the same grid, the method outputs only one of the object classes.
- **SSD** [96] is a VGG-16 based model to extract feature maps and uses a convolution filter to detect objects. This approach provides a good trade-off between speed and



Figure 5.1: Examples of two common architectures of Deep Learning-based object detector. (a) Two-stage detector - FasterR-CNN. [88]© IEEE 2015. One-stage detectors - (b) YOLO © Reprinted by permission from Springer Nature: Springer [98], Copyright (2020). (c) SSD © Reprinted by permission from Springer Nature: Springer [96], Copyright (2016)., and (d) RetinaNet [97]© IEEE 2017.

accuracy. To handle multi-scale objects, it added several convolutional feature layers of decreasing sizes on top of VGG-16. As the result, such pyramid representation allows the network to capture objects of various sizes. SSD generates *default boxes* which is equivalent to anchor boxes of Faster R-CNN, for each location of activation map. Unlike the anchor boxes, the size and position of the default boxes are fixed relative to their associated cell. In contrast to YOLO, SSD does not divide an image into random size grid cell but simply predicts the offset of the default boxes. • RetinaNet is a one-stage object detector that performs well for images that contain dense and small size objects. The backbone of this architecture shown in Fig. 5.1d is ResNet model [83]. RetinaNet is constructed based on the combination of *featurized image pyramids* and *focal loss*. The image pyramid in traditional image processing resolved the scale-variant of the object detection. However it is quite unpleasant due to very expensive computation and memory. On the other hand, featurized image pyramids, which is similar to image pyramid in SSD, resolves object detection at different scales in a faster manner and less computation because CNN utilized pyramid structure.

In addition, the method addresses the imbalance problem between foreground and background classes. It introduced the new loss function, called *focal loss* which solves the foreground-background class imbalance problem on top of FPN. Focal Loss (FL) described in Eq. 5.2 is a modified Cross-Entropy Loss (CE) described in Eq. 5.1. By adding a modulating factor $-(1 - p_t)^{\gamma}$, it appends more weight to hard examples while down-weight on easy examples.

$$CE(p_t) = -log(p_t) \tag{5.1}$$

$$FL(p_t) = -(1-p_t)^{\gamma} log(p_t) ; \gamma \ge 0$$
(5.2)

where γ is a focusing parameter of the focal loss with the default value of 2.

5.2 Data Inspection

The procedure workflow can be found from Fig. 5.2, which is the network selection process based on the real target dataset. The data inspection must be initially carried out. Three networks were carefully picked from the literature regarding our data observation result.

In training neural networks, inspecting data is a critical, inevitable, and very important step because this has effects upon the training procedure. Any irrelevant data or localization errors can worsen the performance of the detectors. Given the collected dataset from the previous chapter, we spent long hours scanning through thousands of generated examples to understand their patterns, distributions, correct ground truth localization error or class name, remove corrupted images, etc.

For the process of splitting data, we adopted the hold-out method for network model selection [218]. The method splits the entire dataset into three different sets, namely training, development (dev), and test. The dev-set is referred to in many terms in the



Figure 5.2: Workflow for object detection network selection procedure.



Figure 5.3: Data statistic of (a) R03-K-N1, (b) R00-S-C2, R01-S-C2, R02-S-C2, and (c) combination of (a) and (b). The different BBox sizes on the x-axis can be categorized as small (BBox< 32×32 px), medium (32×32 px < BBox < 96×96 px) and large(BBox > 96×96 px).

literature such as hold-out, cross-validation, or evaluation set. As a rule of thumb, we chose the train-dev-test split ratio of 70-15-15 for around 47000 examples. Although the trend of the deep learning split training set ratio should be extremely much high than the dev-set and the test set e.g., 90-5-5, 98-1-1, 99.5-0.25-0.25, this ratio setup is suggested for the big data which contains one or more than millions examples. In this case, the dev-set and test set become much smaller percentage because the goal of having these sets is to simply see how the network models behave towards the data, unlike learning in the training process which requires a lot of data to provide sufficient knowledge to guide the learning algorithm for reasonable recognition.

The data statistic for network selection procedure can be found from Fig. 5.3. In spite of different data sources in Chapter 4, the data from real source (i.e., R00-S-C2, R01-S-C2, R02-S-C2, and R03-K-N1) were solely used. Under this circumstance, this ensures not to mismatch the data distribution with simulated data during the network model selection process.

The dataset is categorized based on the size of the BBox according to the standard of [212] and types of the dataset. Fig. 5.3c shows overall statistics. Around 70% of the overall data are subjected to small object size. Almost half of the small objects are occupied by 20×20 px BBox. Two quarters of the small objects are 24×24 px and 28×28 px BBox. Fewer than 4% is 16×16 px BBox. On the other hand, a third of overall data are medium BBox size. Just over half of the medium proportion were 96×96 px BBox and the rest

are 32×32 px BBox. Lastly, less than one percent of the overall data is the object larger than 96 px size.

In brief, the small object size constitutes the highest proportion of the selected dataset. All datasets has only a single object class, which is *person*. As shown in Fig. 5.3a, the statistic of dataset R03-K-N1 contains the large object size more than the other because several zoom-in events occurred during the data record which caused the larger object size. Unlike R03-K-N1, the zoom level of the dataset R00-S-C2, R01-S-C2, R02-S-C2 usually remain in zoom out most position which results in the objects were visibly smaller, see Fig. 5.3b.

5.3 Candidate List of Network Models

Similar to most object detection studies [84], we chose two criteria in picking models for real-world application which are *speed* and *accuracy*. Nevertheless, there are other factors that we do not consider in this work, but also can affect the performance of the detector as well such as boundary box encoding, deep learning software platform that used, the number of predictions or proposals, and feature extractor.

Most of the safety assistances in industry or autonomous vehicles use safety proximity or distance sensor. Using the proximity sensor provides no semantic information. For instance, crane collision avoidance system employs Ultra-wide Band (UWB), Radio Frequency Identification (RFID) tag, laser scanner, or IMU to obtain the pose of the crane and perceive the surroundings [219, 173]. These proximity sensors present the existence of objects but not the accurate position where the object is. Given the information or alarm, the operator must immediately stop the ongoing construction task despite the unknown exact position of the nearby worker. The requirement of the applications using proximity sensors prioritizes the time aspect before the accuracy. In other words, the Safety Response Time (SRT) of the proximity sensors is quite low regardless of given non-semantic information such as no appearance of the target. For instance, once the sonar sensor detects any incoming worker to the forklift, it suddenly gives a warning to the forklift driver without identifying which part of the worker's body violates the working area of the vehicle. As can be seen, the response time or speed is more important than the accuracy in a safety application.

Likewise, our priority goal is speed and then accuracy while providing the semantic information for the operator via the load-view crane camera. Regarding the requirement of visibility assistance, the operator should be alarmed about any surrounding workers-on-foot or to be aware of the hazards in (near) real-time. In this work, we therefore selected two typical deep learning network baselines i.e., Faster R-CNN, two-stage detectors and RetinaNet, one-stage detector, see Fig. 5.5. The backbone architecture or the deep learning classification model which is chosen for all detectors is ResNet [83]. ResNet is one of the most powerful deep learning classification model, which yields ImageNet Large Scale Visual Recognition Challenge or ILSVRC in 2015 [220]. It shone a light after the great success of AlexNet in 2012. ResNet has many variants —ResNet-18, ResNet-34, ResNet-164, ResNet-1202, etc. The digit suffix represents a number of neural network layers of ResNet architecture. The more layer the network has, the better network accuracy [83].

According to the speed and accuracy measurement shown in Fig. 5.4a, the ideal detector should have high mAP and low GPU time, which means the detector should be at the



Figure 5.4: (a) mAP vs GPU elapse real time by meta-architecture and feature extractor. [84]© IEEE 2017, p.8. (b) mAP for size 300 input by meta-architecture and feature extractor. The GPU time report based on a batch size of one. The machine used in this benchmark is Intel Xeon E5-1650 v2 processor, 32GB RAM and Nvidia GeForce GTX Titan X GPU card. [84]© IEEE 2017, p.9.

top left corner of the plot as much as possible. FasterR-CNN models are suitable for high accuracy and low throughput application requirement. On the contrary, SSD models are recommended when processing time is the most important factor.

In addition, Fig. 5.4b reported the accuracy of the detectors based on different object



Figure 5.5: (a) FasterR-CNN detector architecture. (b) RetinaNet detector architecture. The figures were modified from [88][©] IEEE 2015 and [97][©] IEEE 2017 respectively.

sizes—small, medium or large¹ [212]. Obviously, all detectors perform relatively well on large object sizes. Although SSD based models trail in the accuracy of small objects, they are competitive on large objects with FasterR-CNN. SSD models are faster and their feature extractor is more lightweight than the two-stage detectors. One-stage detectors like SSD suffer inequality of class numbers among all training samples.

Unlike the two-stage detectors, the number of the generated candidate from the one-stage detector is fairly huge, 10k to 100k candidates. Among the candidates, it may contain much more easy samples than hard samples. The difficulty level (easy and hard) describes how difficult for the detector to categorize an object. The easier the object class, the more confident the detector. In other words, the loss becomes less when the detector is very certain which class of the object it sees. The loss of easy examples (e.g., background) dominates the loss of hard example e.g., class *person*. Consequently, it reduces the overall accuracy of the detector. This type of example causes the problem for the one-stage detector. Besides the class imbalance problem, one-stage detectors also have difficulty in handling small size objects. The feature mapping of the detector is very low resolution. As a result, the feature of the small objects becomes too less detectable.

In terms of load-view application, the network baseline models which are mostly adopted are Faster R-CNN and RetinaNet instead of SSD detector. The RetinaNet architecture base is widely adopted as a lightweight and accurate aerial image detector in many literature. It addresses the problem of dense and small size objects as discussed in Sec. 2.2. It solves the class imbalance problem by focusing on the loss of the hard example.

¹small (BBox< 32 × 32 px), medium (32 × 32 px < BBox < 96 × 96 px) and large (BBox > 96 × 96 px). BBox stands for bounding box.

Detector Network Model	Backbone architecture	Abbreviation
FasterR-CNN	ResNet50	F50
FasterR-CNN	ResNet101	F101
RetinaNet	ResNet50	R50

Table 5.1: Candidate list of network models.

According to the data observation in Sec. 5.2, most of the data are small. The data content was mostly filled with the plain road which is considered as the high number of easy negative examples or background. Therefore, picking a one-stage detector like SSD is not preferable. As previously mentioned, such one-stage detectors have a class imbalance problem. The loss value will be influenced by the high amount of background examples. Another problem of a one-stage detector is handling small size objects as already stated. RetinaNet copes with this issue by adopting Feature Pyramid Network (FPN) which allows detecting objects in large variation in object scale. As reported in the aerial image object detector challenges [221, 222], RetinaNet base model holds the best AP of the VisDrone detection challenge in 2018. Furthermore, 13% of the VisDrone 2019 challenge participants used RetinaNet architecture base. For FasterR-CNN, approximately 15% and one-fourth of submitted detectors employed this method in 2018 and 2019 respectively.

Based on these assumptions, the following detectors are chosen in the network model selection process which will be discussed in Sec. 5.4. The summary of candidate list can be found in Table 5.1 including their abbreviation which will be referred for the rest of the chapter. FasterR-CNN models with two different backbones (i.e., ResNet50 and ResNet101) are used as the baseline accuracy in comparison, while RetinaNet-ResNet50 is our detector which we aim to employ for the load-view object detection.

5.4 Network Configuration and Selection

In this work, the framework which we used to train and optimize the deep learning algorithm is TensorFlow Object Detection API. This API is an open-source deep learning framework that is built on top of TensorFlow (TF), which is a software library developed by Google Brain for machine learning and aritficial intelligence. The API facilitates building the neural network models starting from data preparation, training, until the deployment phase of the models. The version of TF which is used in this thesis is 1.14.

The hardware resources² for training Deep Neural Network (DNN) is Google Colaboratory (Colab) more commonly referred to as Google Colab or just simply Colab [223]. Colab is a Jupyter notebook environment, which runs in the cloud platform and stores data on Google Drive. It is an interactive programming terminal, hence it is easy to learn and debug. This tool has got several features such as it requires neither setup nor most dependency library installation. Moreover, it offers free access to Google computing resources e.g., GPUs, Tensor Processing Units (TPUs) under a certain time period. In general, the maximum lifetime which Google allows the notebooks connecting to the virtual machine is up to 12 hours.

Before presenting the network selection procedure, we first go through the standard object detection network configuration in TensorFlow Object Detection API in the following

²Intel(R) Xeon(R) CPU @ 2.20GHz, 12.72G memory, Nvidia Tesla T4, 16GB

section. Afterward, the configurations of each network model candidate which were chosen in Sec. 5.3 are presented. Based on the training outcomes, the final model which is suitable for our application is determined in Sec. 5.4.2.

5.4.1 Network Configuration

In TensorFlow Object Detection API, there are three main basic configuration components —model, configuration, and input, as seen in List. 1. All component configuration files are in protobuf format, which can be set in a single file pipeline.config of the TensorFlow Object Detection API. Each training and evaluation process contains both configuration and input components.

```
1
          model {
\mathbf{2}
          (... Add model config here...)
3
          }
4
\mathbf{5}
          train_config: {
6
          (... Add train_config here...)
7
          }
8
9
          train_input_reader: {
          (... Add train_input configuration here...)
10
          7
11
12
          eval config: {
13
          (... Add eval_config here...)
14
          }
15
16
17
          eval_input_reader: {
          (... Add eval_input configuration here...)
18
19
          }
```

Listing 1: Skeleton configuration file of TensorFlow Object Detection API [224].

- Model configuration —Defines which object detection model will be trained. Some examples of the parameters of the network model are the total of object classes, loss, feature extractor, anchor generator, etc.
- Evaluation configuration —Defines which metrics_set will be used during detection assessment. Our detection evaluation is set to pascal_voc_detection_metrics. The TensorFlow Object Detection API supports three main evaluation protocols which were originated from the well-known competitions namely, PASCAL VOC 2010 detection metric, MS COCO detection metric, and Open Images V2 detection. In general, these benchmarks commonly use mAP metric for evaluating the quality of the object detectors. The conventions are slightly different. For instance, the Open Images protocol has hierarchy label system. The MS COCO benchmark has additional statistc of reporting IoU and object size. The detailed comparison among the evaluation metrics can be found in [225]. The num_examples is the number of the test images.



Figure 5.6: This figure visualizes the example of the feature development through layers, starting from the left is the initial layer then goes to the deeper layer as we can see more feature pattern developed[®] Reprinted by permission from Springer Nature: Springer [227], Copyright (2014).

• Input configuration —The train_input_config and eval_input_config define the desired training dataset and evaluation dataset respectively. A tfrecord file contains a sequence of binary sample data records including their annotation. Besides the tfrecord data, a label file, labelmap.pbtxt, is a reference lookup table file of the object class name. The file is referred to in both training and evaluation input configuration. The file contains two fields, namely *id* and associated *object class name*. The label id starts at 1 as the id 0 is reserved for an object class *background*. An example of the label map file can be found in Appendix C.5.

5.4.2 Network Selection

The procedure of selecting the network is following. First, the network configurations of the three networks are discussed. After the training process, the result and the choice of the network are presented.

To yield a low training loss, it is better to start from a pretrained weight of the selected models. A pretrained model is a model trained by others to address a similar problem. This technique generally refers to Transfer Learning (TL) or knowledge transfer, which is another complex research area that addresses what kind of knowledge to transfer, how to transfer, and when to transfer. For a more extensive survey of TL, see [226]. To avoid negative transfer, the researchers try to explain these arising questions which depend on domain and task of the source and target (or destination). It occurs when the knowledge transfer worsen the target learning process. Although the training data and the target data should theoretically be in the same feature space and distribution, this premise may not apply in many real-world scenarios [226]. Therefore, training network models from scratch or building up own network is strongly not recommended at the initial stage of implementation. Although a pretrained model may not first work perfectly in our application, using the pretrained knowledge saves huge efforts instead of teaching the network from zero. As illustrated in Fig. 5.6, such the knowledge can initially ease the feature extraction process of the network as it employs the feature presentation which learned by the previous network that typically trained on a very large-scale dataset.

According to the candidate list discussed in Sec. 5.3, the following network models in Table 5.2 including their pretrained weights were taken from TF detection model zoo [228], which is a marketplace for TF standard model architectures. The networks were trained on the MS COCO dataset which is a large-scale image dataset. It contains approximately 32000 images with the context of everyday humans and objects. To the author's knowledge,

there is no pretrained network available which is trained by crane load-view dataset. The initial weights of each network are in the checkpoint (ckpt) file format which is set as fine_tune_check_point of training configuration file.

 Table 5.2:
 Network model name listed in model zoo.

Network model	Model zoo network model name
F50	faster_rcnn_resnet50_coco_2018_01_28
F101	faster_rcnn_resnet101_coco_2018_01_28
R50	<pre>ssd_resnet50_v1_fpn_shared_box_predictor_640x640_coco14_sync_2018_07_03</pre>

5.4.2.1 Configuring the Trainer

Next, only important parts of the configuration file of the three networks are discussed. The rest of the configuration mostly leave as default and is included in the complete configuration, which are available in Appendix C. Table 5.3 lists the model configuration for the three network candidates.

- Number of classes (num_classes) The aim of this work is to detect worker-on-foot, hence the number of object class is one for all networks.
- Image resizer (image_resizer) —FasterR-CNN family are the model trained to detect object in real aspect ratio thus keep_aspect_ratio_resizer option is chosen. The default [min, max] image dimension is [600,1024]. However the image resolution of our real training data set is 720 × 480 therefore the new image dimension range is [600,900] according to the function of shape aspect ratio calculation in Alg. 5.1. For RetinaNet, the input image is stretched into the default specific width and height i.e., 640 × 640 by using fixed_shape_resizer option.
- Data augmentation (data_augmentation_options) —It is carried out not only to increase the amount of relevant data in our dataset but also to reduce overfitting. In other words, our target application may exist in a variety of conditions such as different orientations, scales, and locations. In addition to the simulated dataset discussed in Chapter 4, another data augmentation technique can be done by synthetically modifying the existing data. Basic augmentation techniques are flip, rotation, scale, crop, translation, and Gaussian noise. On the other hand, advanced techiques are Conditional Generative Adversarial Networks (cGANs), neural style transfer, etc. Despite many augmentation options, we need to consider whether the

Algorithm 5.1: Calculate shape keeping aspect ratio
Input: H, W, min_dimension, max_dimension
Output: min_dimension, max_dimension
1 $ratio_min = \frac{min_dimension}{Min(H, W)};$
2 $ratio_max = \frac{max_dimension}{Max(H, W)};$
3 ratio = Min(ratio_min, ratio_max);
4 $min_dimension = Round(H \times ratio);$
5 $max_dimension = Round(W \times ratio);$

Configuration	F50, F101	R50
Number of classes	1 (person)	1 (person)
Image resizer	keep_aspect_ratio_resizer, [600,900]	fixed_shape_resizer, 640×640
Data augmentation	 random horitzontal flip random vertical flip random jitter boxes with ratio of 2e-2 	random horitzontal fliprandom vertical fliprandom crop image
Number of training steps $(\times 10^3)$	700	50
Batch size	1	16
Optimizer	Momentum	Momentum
Learning rate (η)	step decay	cosine decay
Anchor scales	[0.0625, 0.125, 0.25, 0.5, 1.0, 2.0]	

 Table 5.3: Model configuration for each network candidate. Both FasterR-CNNs used the same configuration.

option is relevant to our target application or not. For instance, augmenting data for the upward arrow classifier with the vertical flipped images, which becomes a downward arrow, will lead the model in a completely wrong direction. According to our top-view dataset, an image could be possibly view rotated by 180 degrees, which is equivalent to a vertical flip or horizontal flip. The two flips applied to both FasterR-CNNs. Moreover, random jitter or Gaussian jitter was applied with a ratio of 0.02. For RetinaNet, horizontal and vertical image flipping and random crop image were kept as the default option.

- Number of training steps (num_steps) —Both F50 and F101 were set to train for 700k steps. On the other hand, RetinaNet required much less than the FasterR-CNN because of the smaller network therefore we set to 50k steps.
- Batch size (batch_size) —Similar to the training steps, the smaller the network size, the higher the batch size can be. Increasing batch size results in increasing the required GPU memory. On the other hand, taking a bigger batch size allows the network to directly reach a local minimum faster. The default batch size of both FasterR-CNNs is 1 which we found out that it is also suitable for our hardware to have smooth training without GPU memory error while training. The batch size of RetinaNet can be set upto 16 not to be over the memory limitation.
- Optimizer (optimizer) —All three models used momentum [229] as the default gradient descent optimization algorithm. The typical momentum value γ is set to 0.9. Momentum term in practice means a strength gained by motion or a series of actions. The more actions are built, the more motion is generated. As a result, momentum is analogous to great speed improvement in learning. The momentum is a method to accelerate Stochastic Gradient Descent (SGD) in the related direction including absorb of loss oscillation. A further detailed comparison of gradient-based optimization algorithms can be found in [230].
- Learning rate (learning_rate) —It is one of the most important optimization parameters. As the name implies, it controls how fast or slow a Neural Network (NN) learns the training samples. In other words, how often the model should update the weight in response to the estimated loss. Too large a learning rate will result Gradient Descent (GD) overshoot to the minimum. It may fail to either converge or diverge. On the contrary, finding the minimum can be slow if the learning rate is too

small. For FasterR-CNN, we changed the default initial learning rate to be smaller instead of 3e - 4. To anneal the learning rate over time, a step decay method was adopted with a reducing factor of 0.1. The learning rate was scheduled based on the step number or after a set of training epochs as in Eq. 5.3.

$$\eta_{FasterRCNN} = \begin{cases} 1e - 4 & ; \ 0 < step < 500k \\ 1e - 5 & ; \ 500k \leqslant step < 700k \\ 1e - 6 & ; \ \text{otherwise} \end{cases}$$
(5.3)

Instead of reducing the learning rate stepwise, RetinaNet used the cosine learning rate decay technique. The annealing schedule relies on the cosine function. Unlike one single downward trip for searching minimum, the model undergoes several learning rates annealing cosine cycles or moving ridges.

• Anchor scales (grid_anchor_generator) —In constrast to MS COCO, the overall object size discussed in Sec. 5.2 is much smaller. Therefore, we appended two additional smaller anchor scales i.e., 0.0625 and 0.125 to the default [0.25, 0.5, 1.0, 2.0]. For RetinaNet, this parameter does not exist hence it is not applicable.

5.4.3 Result of Network Selection

The training results of the three network candidates are shown in the second to the fourth columns of Table 5.4. The test results are shown in the last three columns of the identical table. The hardware that was used during the test procedure in FINROC was Intel(R) Xeon(R) Gold 6126 CPU @ 2.20GHz, 12 cores, 12.72G memory, and NVIDIA GeForce GTX 1660 Ti, 6GB.

Among the three network models, the network size of R50 is relatively small. Training process therefore took much less than both Faster R-CNN i.e., seven times less. With the same reason, its speed performance is 3-3.5 times faster. The accuracy of R50 was comparable to the large network, Faster R-CNN. The snapshot of evaluation can be found in Fig. 5.7. First, the objects in the scenario in R00-S-C2 dataset (1st column of Fig. 5.7) appeared to be stood out of the background. All the workers wore the emergency vests and the ground of the image sequence was very dark. As a result, all detectors were able to properly distinguish and recognize the workers. For the second scenario as R02-S-C2 dataset (2nd column of of Fig. 5.7), the workers were very small. The appearance of the worker was mostly in black without wearing a high-visibility jacket or safety helmet. Moreover, there were many circular objects (e.g., manhole or rocks) that closely resembled the head of the worker from the top view. This is very difficult not only for the detectors but also for a human to differentiate the objects out of those similar object shapes. Despite

		Train		Test		
Dataatian		Number		Average		
Notwork Architecture	Batch size	of	Training time	detection time (ms)	FPS	\mathbf{AP}
Network Architecture		epochs $(\times 10^3)$		per frame		
R50	16	50	48 hours	238	3.986	0.72
F50	1	700	2 weeks	714	1.329	0.69
F101	1	700	2 weeks	833	1.139	0.70

Table 5.4: Training results of the three network models.

no false positive of the similar objects, F50 and R50 missed several workers, see Fig. 5.7b and 5.7e. On the other hand, F101 was able to detect them all as shown in Fig. 5.7h. Lastly, the 3rd column of Fig. 5.7 represented the R03-K-N1 dataset. All detectors were generally able to detect workers without false positive with such shadow. Unlike the result in Fig. 4.31d, SSD-OkuPed, which were trained by UAV dataset, included both worker and their shadow as a single object. Regarding network selection criteria discussed in Sec. 5.3, we opted RetinaNet-Resnet50 or R50 as our final network model as it was able to satisfy the requirement both in terms of speed and accuracy based on our load-view object detection application. As the higher prior criteria of speed, R50 was able to gain the best time performance not only runtime, but also took less time than other networks to train. In term of accuracy, focal loss of R50 solved the shortcoming of the class imbalances problems which resulted the network to distinguish the small size object from the background.



Figure 5.7: The snapshot of the detection result on three datasets (i.e., 1st column: R00-S-C2, 2nd column: R02-S-C2, and 3rd column: R03-K-N1). The ground truth is shown in blue BBox and the predicted result is shown in green BBox. The first row (a-c) are results of R50. The second row (d-f) are the result of F50. The last row (g-i) are the result of F101.

5.5 Discussion

In this chapter, top view object detection using deep learning approaches had been investigated. The network selection procedure is mainly conducted in this chapter to seek for the appropriate network for our load-view worker detection. The extensive data inspection first showed that most of the object size among the real load-view crane camera datasets were small $(20 \times 20 \text{ px})$. Later, three network model candidates namely, FasterRCNN-Resnet50 (F50), FasterRCNN-Resnet101 (F101), and RetinaNet-Resnet50 (R50) which are mostly adopted in terms of load-view object detection application or similar, were selected. They were top primarily network models selected in object detection research in many application domains, but not limited to, safety in construction area, AV, surveillance, etc. The network selection criteria was speed over accuracy due to the requirement of visibility assistance, which the operator should be warned regarding any nearby hazards such as workers-on-foot in (near) real-time. The network configuration of each model had been properly adjusted according to the data inspection. Finally, R50 or RetinaNet detector was chosen out of the three detector choices. R50 detector achieved the good result, specially in term of speed and small object size. Moreover, its training duration time was seven times less than the R-CNN series with the comparable accuracy.

The deep learning network struggles to detect the objects from the top view since the information embedded in each BBox is not sufficient for robust detection performance. Hence, the active zoom control is introduced to improve the performance of the detector instead of parameter tuning and retraining the deep learning network. By controlling the camera zoom we can adapt the size of the BBoxes in the image and boost the number of features in each BBox. The output of the detector is used as sensory information in a feedback loop to control the camera zoom. Since the zoom control performance is sensitive to the sensory data it is crucial to achieving a detector output that has (a) low delay and (b) stable detection rate. The selected detector is suitable for zoom control since it provides a suitable compromise on both of the above conditions.

6. Analysis of Motorized Zoom Camera

In the previous chapter, a suitable network model was determined for the worker detection from the load view camera. Three renowned object detection network models i.e., FasterRCNN-Resnet50 (F50), FasterRCNN-Resnet101 (F101), and RetinaNet-Resnet50 (R50) were chosen as network candidate for the application. According to the speed and accuracy criteria, R50 detector outperformed the rest of the detectors. Nevertheless, the result remains imperfect. Top-view images inherently contain a poor feature set for worker detection, explaining the low performance of the state-of-the-art detectors. Furthermore, the size of the Bounding Boxs (BBoxs) are relatively small in the crane applications which exacerbates the feature deficiency in the image.

The crane operator actively uses a zoom camera to increase the size of the workers, which further decreases the efficiency of the detection algorithm due to dynamic and abrupt changes in the size of the targets. Although R50 detector is robust in handling variations in BBox scale, it is insufficient to cope with rapid scale change resulting from the zoom.

In this chapter, we proposed an adaptive zoom control scheme that not only assists the operator during load lifting but also improves the accuracy of the detection network. The proposed adaptive zoom control incorporates the requirements of the detector to achieve a smooth zoom operation and track the optimum object size.

6.1 Adaptive Zoom Camera and Its Applications

A zoom camera offers the flexibility to change focal lengths in a single lens. The zoom level corresponds to the value of focal length. Zoom camera is used in a wide range of applications. In Human-robot Interaction (HRI), Atienza and Zelinsky [233] implemented an active stereo zoom camera for optimal gaze tracking. The face is first detected using a skin color feature and later the gaze. The quality of the gaze is based on the image resolution of the face. To keep face image resolution high, the zoom is automatically adjusted based on the distance between the head and the camera. For zoom camera calibration, the authors indicated that it was more challenging relative to a static camera



Figure 6.1: Differt camera lens types. (a) Fixed lens camera. [231]. (b) (Motorized) Varifocal Lenses and (c) PTZ camera [232]© IEEE 2016.

because there is a large number of possibilities of lens settings. Similarly, Zheng et al. [234] employed a camera with zoom and focus control in guiding robot motion. The authors demonstrated that changing zoom and focus improves the accuracy of positioning the robot end effector to grab an object. Moreover, they also suggested using a zoom camera instead of a classic stereo camera or multiple viewpoints from several cameras.

In the construction domain, Azar [235] improved construction equipment detection for an automated monitoring system for productivity. The aim of the zoom function in this study is to both obtain a suitable resolution and have a wide Angle of View (AOV) to monitor the overall construction area. An construction equipment is detected via visual marker i.e., AprilTag [236]. The author proposed an automated zoom control algorithm to have reliable detection. The active zoom control maintains the minimum tag pixel resolution [237]. The zoom function of the crane load-view camera is necessary for the crane operator. Depending on the load size, the operator zooms in or out to observe the distance between load and workers including obstruction not to hit them. Vierling et al. [238] proposed an automatic zoom load-view camera based on the working zone and load occlusion. The authors trained the CNN with the load-view images and several zoom levels, which then results in the optimal zoom level for the operator. Li et al. [130] increased the precision in tracking tower crane hook during hoisting to avoid blind lifting. To capture the hook movement, the author used the PTZ surveillance camera to detect the hoist cable instead. The cable provides better and stable features than the hook since linear cable shape can be consistently detected as it always points downward and is vertically aligned. The adaptive zoom is used to maintain the hook size on the monitor. The hook is recognized by the color histogram of the warning stripes on the hook e.g., yellow, black and red. The zoom is adapted based on the defined pixel distance of the hook which is recognized and maintained to appear between 0.125 and 0.1 of the full image width in pixel.

Zoom control is a critical subject. One major problem with all the studies reviewed is that they fail to take zoom control performance into account. To zoom to a certain condition e.g., the defined specific pixel size requires stable zoom control to maintain the state. The closed-loop zoom control is highly sensitive to the performance of its vision-based perception system. Imperfections in the sensor system result in oscillations in the control loop. For example, jittery target detection introduces a nonlinear effect on the control loop which induces abrupt zoom or undesirable oscillations.



Figure 6.2: Accessible zoom signals. The yellow arrow denotes the sensor signal, while the red arrow denotes the control signal. O_{zi}, O_{zo} are zoom in and zoom out control commands.

6.2 Implementation of Zoom Control

In this section, the control interface of zoom camera MC5200 is introduced. This camera is developed by Motec and is deployed on the crane for experimentation. The camera monitor system consists of two main components i.e., the camera and the control unit. The communication between the camera and the control unit is via CAN bus protocol [206]. All inputs and outputs are connected to the control unit. The control can be executed manually by either the remote controller or sending the control command via CAN bus protocol. The accessible signal are listed in Fig. 6.2. Due to the product confidentiality, we have partial access to the camera which are 2D streaming video, basic controller function i.e., *Zoom in* (O_{zi}) and *Zoom out* (O_{zo}) which make the objects inside the image becomes larger and smaller, respectively. Therefore, the desired zoom level can not be precisely adjusted, and we can only control the direction of the zoom by issuing O_{zi} or O_{zo} commands to zoom in or zoom out respectively. When the camera receives the CAN message, a step-motor drives the lens forward or backward. Unfortunately, the amount of change in focal length is nondeterministic, and hence zoom level is not available. However, the zoom speed can be adjusted by the time interval T between two CAN messages.

To test the zoom speed control, a fiducial marker is positioned in the distance of 20 m from the camera. The workflow of the test can be found in Fig. 6.3. First, the camera is initialized to maximum zoom out. We counted the number of pulses that are sent to the camera by a parameter called zoom step count or zoom step (Z_i) . At the start $Z_i = 0$ which means no zoom. The zoom pulse interval T is initialized at 50 ms. In other words, the zoom control command is sent to the control unit every 50 ms. Our Hardware Abstraction Layer (HAL) of the camera in FINROC started to generate the zoom in pulse command until the camera reaches the maximum zoom in. Afterwards, we calculated downward turn by zooming out. Likewise, the zoom step is decreasing while the camera is zoom level and zoom step count Z_i for a variety of intervals. The zoom speed is the slope of the lines in Fig. 6.4. For $T \ll Z_i$ for a variety of the zoom step is almost constant, indeed zoom commands are sent faster than the response time of the zoom motor.



Figure 6.3: Workflow for zoom pulse control command test.



Figure 6.4: Zoom step count Z_i vs time. Each legend represents period T or zoom interval. The zoom level is measured by fiducial marker in placed at 20 m from the camera.

level is not controllable and the camera reaches rapidly to the digital zoom zone i.e., zoom level higher than 500. This indicates that the consistent zoom interval in the range of 300 - 1000 ms yields a controllable zoom speed. The highest zoom speed can be reached

at $T = 300 \,\mathrm{ms}$ and by increasing T the zoom speed linearly decreases.

Despite the lack of access to absolute zoom level, in this section, we devised a method to control the zoom speed of the camera by using the zoom interval parameter T. Additionally, the zoom count parameter Z_i also provides an approximate zoom level. These two parameters will be used as camera inputs in our proposed method for autonomous zoom control.

6.3 Worker Localization in Camera Coordinate

The zoom level of the camera can be used to localize the target with respect to the camera. This section provides the mathematical calculations to exploit the localization advantage of the zoom camera. Knowing the size of the target, it will be shown that it is possible to estimate the position of the person with respect to camera and load. This information is crucial to assess if the detected worker is inside the safety zone.

The location of the worker P_c in the camera coordinate O_c (3D) is estimated, given the location of the worker in pixel coordinate O_p (2D). This can be used can a piece of base information for further applications which can be developed as future works. For instance, the speed of the workers around the crane moving from one point to another can be further calculated from their position and time difference. Consequently, we can assess the accident risk of the workers. If the workers travel toward the crane very fast or come nearby the load falling zone, there is a high possibility that the worker can get an accident. The safety assistance system can warn the operator in advance when the worker who is coming fast, is still far away.

To determine the object size in the world units or the camera's location in the scene, it requires the parameters of a lens and image sensor of the camera itself. Camera calibration is the process to estimate the parameters such as intrinsic parameter K, scale factor s, focal length f, and extrinsic parameter. The pinhole camera model [239] depicted in Fig. 6.5 is employed in this work. The model maps an object in the 3D world scene onto the image plane. The mathematical model which describes how a point in a 3D scene gets projected into 2D pixel coordinates is called forward projection. In contrast, backward projection model maps a point in pixel coordiates back to the 3D world coordinates as depicted in the diagram in Fig. 6.5 and Table 6.1. The pinhole camera matrix comprises two matrices i.e., the extrinsic and intrinsic matrix. The extrinsic matrix represents the location of the camera in the 3D world and which angle it is pointing. It has two components, namely a rotation matrix ${}^{c}R_{w}$ and a translation vector ${}^{c}T_{w}$. On the other hand, the intrinsic matrix in Eq. 6.1 represents the focal length and the principal point of the camera. The following value in matrix K belongs to the camera MC5200 calibrated at maximum zoom.

$$K = \begin{bmatrix} f_x & \epsilon & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 820.30 & -1.9201 & 352.79 \\ 0 & 890.40 & 306.64 \\ 0 & 0 & 1 \end{bmatrix}$$
(6.1)

where ϵ is skew coefficient. (c_x, c_y) is the principal point or optical center in pixels. $K_{3\times 3}$ is the camera intrinsic matrix. (f_x, f_y) is the focal length in pixels. It is the distance from the center of the lens to the principal foci (or focal points) of the lens.



Figure 6.5: Pinhole camera model. P_c is the location of a worker in the world coordinate.

Table 6.1: Camera projection. The right arrow denotes the forward projection, while the left arrow denotes the backward projection. A is rigid transformation matrix. $^{\triangle}P_{\boxplus}$ defines as a point P in coordinate \boxplus, O_{\boxplus} with respect to coordinate \triangle, O_{\triangle} .

Backward projection		$^{w}A_{c}$	$^{c}A_{i}$	$^{i}A_{n}$
	World	∽ Camera	rightarrow Image (Film)	$\stackrel{r-p}{\leftrightarrows}$ Pixel
Forward projection		$^{c}A_{w}$	$^{i}A_{c}$	${}^{p}A_{i}$
Dimension	3D	3D	2D	2D
Coordinates (Origin)	O_W	O_C	O_I	O_P
Point	${}^{w}P_{w} = \begin{bmatrix} x_{w} \\ y_{w} \\ y_{w} \end{bmatrix}$	${}^{c}P_{c} = \begin{bmatrix} x_{c} \\ y_{c} \\ z_{c} \end{bmatrix}$	$^{c}P_{i} = \begin{bmatrix} x_{i} \\ y_{i} \end{bmatrix}$	${}^{c}P_{p} = \begin{bmatrix} x_{p} \\ y_{p} \end{bmatrix}$

6.3.1 Camera Projection

As there is no genuine zoom step returned from the original image sensor, the focal length f could not be achieved accurately. Nevertheless, we attempted to examine under the condition of available interface which is in the case of zoom out max or no zoom (i.e., no focal length changed). First, the camera matrix was obtained by using the standard Open source computer vision (OpenCV)¹ library. The checkerboard pattern was used as a reference during the calibration process. The nomenclature list for MC5200 camera calibration paramether can be found in Table 6.2. Given a point in a pixel coordinate ${}^{p}P_{w}$, the location of a worker P_{c} in the world coordinate or ${}^{w}P_{w}$ can be derived through the following steps. A position of a worker in pixel coordinate ${}^{p}P_{w}$ can be obtained from any top view worker detectors, assuming all workers stand on the same flat ground plane.

 $^{^{1}}OpenCV - v3.4.5$

Symbol	Definition	Value		
		820.30	-1.9201	352.79
K	intrinsic matrix	0	890.40	306.64
		0	0	1
(f_x, f_y)	focal length (pixels)	(820.30,8	890.40)	-
(c_x, c_y)	principal point (pixels)	(352.79, 3	306.64)	
ϵ	skew coefficient	-1.9201		
$^{\triangle}P_{\oplus}$	a point P in coordinate \boxplus, O_{\boxplus} with respect to coordinate \triangle, O_{\triangle}	-		
$^{c}R_{w}^{-}$	Rotation matrix from O_w to O_c	$I_{3 \times 3}$		
$^{c}T_{w}$	Translation vector from O_w to O_c	$0_{4 \times 1}$		

Table 6.2: Nomenclature for MC5200 camera calibration.



Figure 6.6: Camera projection from top view. d_w is a person's shoulder width whose default value is an avarage men's shoulder width (i.e., 40 cm). d_p is a diameter of the detected person BBox resulted from top view worker detector in pixels. f is focal length. z_c is the distance between the camera and the head of the person.

In particular, the point (x_p, y_p) is located at a center point of each detected BBox. For simplicity, we started with the forward projection equation.

$${}^{p}P_{w} = K({}^{c}R_{w}{}^{w}P_{w} + {}^{c}T_{w})$$

$$K^{-1p}P_{w} = {}^{c}R_{w}{}^{w}P_{w} + {}^{c}T_{w}$$

$$K^{-1p}P_{w} - {}^{c}T_{w} = {}^{c}R_{w}{}^{w}P_{w}$$

$${}^{c}R_{w}^{-1}(K^{-1p}P_{w} - {}^{c}T_{w}) = {}^{w}P_{w}$$

$${}^{w}P_{w} = {}^{c}R_{w}^{T}(K^{-1p}P_{w} - {}^{c}T_{w}); {}^{c}R_{w}^{-1} = {}^{c}R_{w}^{T}, \left|{}^{c}R_{w}\right| = 1$$

$$= {}^{c}R_{w}^{T}(K^{-1}\left(s \begin{bmatrix} x_{p} \\ y_{p} \\ 1 \end{bmatrix}\right) - {}^{c}T_{w})$$
(6.2)

Assume the world coordinates is the same as camera coordinates i.e., $O_w = O_c$. Therefore, there is no cR_w and cT_w (i.e. ${}^cR_w = I_{3\times 3}$ and ${}^cT_w = 0_{4\times 1}$), we then get

$${}^{w}P_{w} = K^{-1}(s \cdot {}^{p}P_{w})$$
 (6.3)

Regarding to Eq. 6.3, it is impossible to find ${}^{w}P_{w}$, given ${}^{p}P_{w}$ and K because s is unknown. s depends on the object size in the real world. For instance, two identical objects are located in front of a fixed position camera at a different distance. The s value of the closer object to the camera is smaller than the s value of the farther object. To bypass finding s, there is another approach. By incorporating the average men's shoulder width referred as the object size in the real world d_{w} , we can estimate the scale factor s as long as the focal length does not change. Hence, the scale factor can be derived via similar triangles rule, see Fig. 6.6.

$$\frac{d_p}{f} = \frac{d_w}{z_c} \quad ; z_c = s \tag{6.4}$$

The test was executed both in the simulation platform and the real world. The result of the backward projection is illustrated in Fig. 6.7. The load falling zone can be parameterized by setting the radius shown in the red circle which means there are workers inside the fall zone. The fall zone is typically projected out further the suspended load. The calculation of the fall zone is beyond the scope of this thesis. In general, it is estimated based on the swing radius and the direction where the load move.

6.4 Proposed Approach

In this section, we proposed an adaptive zoom control method to eliminate the retraining and data augmentation process in object detection using DL algorithm, and meanwhile increase the situational awareness of a crane operator.

The data-driven method mandates a large amount of training data to reach high accuracy. Detecting objects from a load-view crane camera is challenging especially in the construction area. The object appears in wide-ranging size and appearance. During lifting, the distance between the load-view camera and the ground is dynamically changing because the boom arm can be lowered or extended. Hence, the detected objects appear differently—small, medium or large² [212]. Additionally, the background is full of features that can be easily misclassified as a worker. On the other hand, recording data from the crane for training data-driven detector is expensive and effort demanding such as crane rental, (un)mounting sensor on the huge machine, and data annotation.

The image sensor in this work is a Motec MC5200 crane motor zoom camera mounted at the boom tip of the mobile crane with a pendulum bracket, see Fig. 3.1. The original video output is analog which contains an interlaced display. It results in a partially interlaced image after digitalization. As mentioned in Sec. 4.5, the camera can do basic controller function by sending the output control command, *Zoom in* (O_{zi}) and *Zoom out* (O_{zo}) which make the objects inside the image becomes larger and smaller, respectively.


Figure 6.7: Worker localization without zoom using 3D backward projection. The estimated worker's shoulder width = 40 cm. The load falling zone can be parameterized shown in red circle with 2.5 m radius.



Figure 6.8: The sketches illustrate two scenarios of the proposed FSM, target size preservation (left) and target area preservation (right). R_+ is the outer region filled with a dot pattern. R_- is the inner region filled with upward diagonals. R_c is the center region. R_t is the overall detected region which is shown in translucent. On the left figure, the green translucent region R_t identifies the satisfied constraints of size including the borders which show in black. On the other hand, the right figure depicts the violated case of both size and region which are identified by the red translucent rectangle and red border. d_i is a bounding box diagonal of target x_i in pixel.



Figure 6.9: Zoom controller manager architechture in FINROC.

6.5 System Architecture

In our system modules, there are two main components, namely perception, and control, see Fig. 6.9. First, the perception part is an object detector. In a real-world application, a

 $^{^2 {\}rm small}$ (BBox< 32 \times 32 px), medium (32 \times 32 px < BBox < 96 \times 96 px) and large (BBox > 96 \times 96 px). BBox stands for bounding box.



Figure 6.10: Dataflow of the adaptive zoom control architecture.

worker detection using deep learning approach, whereas AprilTag detector, which is adopted from [236] used to evaluate our proposed zoom control method. In this chapter, only the adaptive zoom control method including its verification will be discussed. The application of the adaptive zoom control to the worker detection will be later presented in Sec. 7.2 as a comprehensive experiment. Second, the control regulates the zoom level to satisfy the defined constraints. We applied a finite state machine (FSM) for control strategies to generate O_{zi} and O_{zo} pulse command as the output from our ZoomController by incrementally increase or decrease, respectively. The detail of the control logic will be further discussed in Sec. 6.6. The system data flow is shown in Fig. 6.10. The load-view crane camera feeds image frames to the detector. The detector processes the image and subsequently passes the recognized bounding boxes (BBoxes) to ZoomController. Finally, the controller generates the zoom command based on the observation back to the camera to adjust the zoom level.

AprilTag Detector: The incoming sensor data, like the worker detector, can be inconsistent. It can cause difficulty to assess the zoom control logic. To decouple the control part from the perception, we then verified our control using AprilTag. The AprilTag detector is used as a reference of sensor data. This fiducial marker detector provides relatively more reliable and consistent sensor data. In other words, using the visual fiducial marker creates more measurable and controllable experiments[236]. Additionally, the output from both detectors, which is BBoxes, is comparable.

6.6 Zoom Controller

The proposed solution exploits the zoom function of the standard crane camera to keep the quality of the detector instead of parameter tuning and retraining the DL network. The principle idea of the method is to maintain the BBox size of all target instances while keeping them in the image frame as long as possible. In particular, the zoom method preserves the targets in the image frame not to let them out of the camera FOV. The regions, R_{\pm} , R_c are additionally defined to restrict targets by two frame offsets $\Delta_{R\pm}$, shown in Fig. 6.8. Each offset is equally positioned in both x- and y-axis. The outer region R_+ is a prohibited zone, where any target should not be inside. R_{-} is an inner region while R_{c} is a center region.

The zoom control logic in *ZoomController* is mathematically modeled in the Mealy FSM, see Fig. 6.11. The nomenclature of the zoom control is described in Table 6.3. The FSM is defined using a 6-tuple $(S, S_0, \Sigma, \Lambda, T, G)$ as the following.

- A finite set of all states $S = \{S_E, S_{TA}, S_{TD}\}$
- An initial or reset state $S_0 = S_E$
- A set of inputs $\Sigma = \{\overline{N}, \overline{D}, \overline{A}\}$
- A set of outputs $\Lambda = \{O_{zi}, O_{zo}\}$
- Transition function $T: S \times \Sigma \to S$
- Output function $G: S \times \Sigma \to \Lambda$
- A set of parameters $\Pi = \{D_D \pm \Delta_D, \alpha, R_{\pm}, R_c\}.$

Symbol	Definition	Value
Input		
\overline{A}	Moving average R_t	-
d_i	Bounding box diagonal of target x_i (pixel), see Fig. 6.8	-
D_{SD}	Standard deviation of D_t (pixel)	-
D_t	Instant average diagonal of all targets (pixel)	-
\overline{D}	Moving average of D_t (pixel)	-
Ι	Image frame	-
N	Instant target number	-
\overline{N}	Moving average of N	-
P_X, P_Y	A set of instant target BBox coordinates in x- and y-axis	-
R_t	Instant overall detected region, see Fig. 6.8	-
X_t	A set of detected targets at time t	-
Output		
O_{zi}, O_{zo}	Input zoom control command in and out	-
Z_t	Instant zoom level	$[0, Z_{ci,max}]$
Parameters		
D_D	Desired BBox diagonal (pixel)	60
R_{\pm}	Outer and inner region, see Fig. 6.8	-
R_c	Center region, see Fig. 6.8	-
α	Moving average window size	5
Δ_D	Range of the desired BBox diagonal D_D (pixel)	10
$\Delta_{R+,T}, \Delta_{R-,T}$	Image frame offset of R_+ and R for tag detector	(5,35)
$\Delta_{R+,W}, \Delta_{R-,W}$	Image frame offset of R_+ and R for worker detector	(45,75)

 Table 6.3:
 Nomenclature for zoom controller.

The inputs of FSM are obtained by preprocessing the raw data X_t . The component of a detected target BBox consists of top left box x_{tl}, y_{tl}, w , and h in an image coordinate. In addition to the set of inputs Σ , we have a set of parameters Π . The values of Π are chosen



Figure 6.11: Mealy finite state machine diagram of the *ZoomController* logic. \Re is a reset signal.

by trial and error experiments. The moving average (MA) is applied to the input data as a noise filter.

The set of states $S = \{S_E, S_{TA}, S_{TD}\}$ is designed to associate to three following scenarios, namely target loss, target area preservation and target size preservation, respectively. Fig. 6.8 depicts the last two scenarios. The pseudocode of the method is described in Alg. 6.1. We manipulate the zoom control by zoom level and perception. Zoom level Z ideally represents how much the camera lens has move based on the zoom pulse command as there is no original zoom control access. The following presents the definition of state machine in Fig. 6.11 including the transition T and output G function.

- State Explore S_E This state corresponds to target loss case (T, G:SearchTarget). When there is no target or the detector is unable to recognize the target, the camera should explore or search for the target(s) by zooming in or out. The scenarios is depicted in Fig. 6.8 on the left. The state machine is initiated or reset to this state. The zoom level Z of the camera must be set first at the zoom out max $(Z_0 = 0)$. Then the camera starts to search for targets until the target appears in the image frame or the detector is able to recognize it, which implies $\overline{N} > 0$. The searching procedure is carried out by zooming in $(O_{zi}: Z_{t+1} = Z_t + 1)$ until the camera reaches maximum zoom in $(Z_t = Z_{ci,max})$ then it starts to zoom out $(O_{zo}: Z_{t+1} = Z_t - 1)$. The procedure repeats until a reliable target is found. In this case, the next state goes to S_{TA} to further observe the overall target area.
- State TrackArea S_{TA} This state corresponds to target area preservation case which the overall target area is assured in the center area (T, G: AdjustRegion). Any target steps into the region R_+ , the camera should adjust the zoom level to keep the target inside at least in the inner region R_- or the center region R_c as long as it is not beyond the camera FOV limit, see Fig. 6.8 on the right. In other words, if \overline{A} intersects R_+ , the camera zooms out until \overline{A} intersects R_- or \overline{A} does not anymore overlap with R_+ . When the area criterion is satisfied, the next state goes to S_{TD} .
- State *TrackDiagonal* S_{TD} This state corresponds to target size preservation case (T, G: AdjustDiag). The camera should adapt the zoom level to keep the average

diagonal value of overall detected objects \overline{D} to the ideal diagonal D_D which is suitable to the selected DL detector. In particular, this condition $(D_D - \Delta_D) \leq \overline{D} \leq (D_D + \Delta_D)$ should be satisfied. When the \overline{D} is lower than the desired diagonal range, the camera zooms in to observe the targets closer, and vice versa. Unless the overall detected area \overline{A} complies, the next state goes back to S_{TA} because the area criterion has higher priority than the diagonal one.

6.7 Zoom Controller Verification



Distance from the camera to the end of corridor = 20.3 m

Figure 6.12: Zoom verification setup on the hallway.

This section presents the verification of the zoom controller using AprilTag as a reference target to evaluate the controller function. The AprilTag family is 36h11 with the size of 16×16 cm. The test was set up in a hallway. Both camera and the tag were placed on the same ground plane. The maximum distance from the camera to the corridor end was 20.3 meters as depicted in Fig. 6.12. During the experiment, only the tag was moved farther away or nearer to the camera. The D_D is primarily set to 60 with its offset Δ_D of 10 pixels. The graph in Fig. 6.13 shows how the zoom level Z adapted to the target. In each image frame, the dashed line locates in between the region R_+ and the region R_- , while the dotted line divides between the region R_- and the region R_c . The further detail of the substate of FSM can be found in Appendix D.

At t_1 , both \overline{D} and Z_t are zero because no tag was found. Therefore, the FSM started to search for the target by O_{zi} . Despite the tag was found at t_2 , the FSM continued to zoom in because \overline{D} remained lower than the floor of D_D . At t_3 , Z_t started to be steady as it met the diagonal criterion. At t_4 , D_D was later manually increased, thus the zoom control started to zoom in and \overline{D} was then back again in range at t_5 .

From t_6 to t_7 , Z_t slightly depreciated because ZoomController tried to maintain the size by O_{zo} as the tag was moved toward the camera which caused \overline{D} became larger and accordingly exceeded $D_D + \Delta_D$.



Figure 6.13: ZoomController verification using AprilTag. The dashed line locates in between the region R_+ and the region R_- , while the dotted line divides between the region R_- and the region R_c . The translucency on the tag identifies the size violation. The green tag at $t_{3,5,7,11}$ means \overline{D} are in target range, $(D_D - \Delta_D) < \overline{D} < (D_D + \Delta_D)$. The yellow tag at $t_{2,4,10}$ means it is below the range, $\overline{D} < (D_D - \Delta_D)$. The red tag at $t_{6,9}$ means it is over the range, $\overline{D} > (D_D + \Delta_D)$. The red border at t_9 identifies the area violation i.e., \overline{A} overlaps with R_+ . The complete state of tag detection and camera state can be further found in Fig. D.1 and D.2.

Between t_8 and t_9 , the tag was removed out of the camera FOV. For this reason, ZoomController went to the explored state S_E . At t_9 , \overline{D} suddenly soared up during the searching because the tag immediately appeared with violated \overline{D} and \overline{A} where the tag was colored in translucent red and the borders visualized in red, respectively.

At t_{10} , The zoom level Z_t gradually rose because the tag is moved away from the camera. \overline{D} became lower than the desired range where the tag was colored in translucent yellow. ZoomController simultaneously tried to handle until the tag is back in the D_D range at t_{11} .

Algorithm 6.1: Adaptive zoom control algorithm **Input:** Σ, I **Output:** $\Lambda = \{O_{zi}, O_{zo}\}$ **Parameters** : $\Pi = \{D_D \pm \Delta_D, \alpha, R_{\pm}, R_c\}$ 1 Initialization; **2** $Z_0 := 0;$ **3** $S := S_0;$ 4 while $true \ do$ $X_t := \text{DetectTarget}(I);$ $\mathbf{5}$ $N := n\{X_t\};$ 6 $\overline{N}_0 := (\overline{N} == 0);$ 7 $R_t := BBox(min \ P_X, min \ P_Y, w, h);$ 8 $\begin{aligned} h_t &:= D D O t(min \ T \chi, min \ T \gamma, w, n), \\ d_i &:= \sqrt{w_i^2 + h_i^2}; \\ D_t &:= \frac{\sum_{i=1}^N d_i}{N}; \\ \overline{A} &:= \frac{1}{\alpha - 1} \sum_{i=1}^\alpha R_i := \frac{R_1 + R_2 + \dots + R_{\alpha - i}}{\alpha}; \\ \overline{D} &:= \frac{1}{\alpha - 1} \sum_{i=1}^\alpha d_i := \frac{d_1 + d_2 + \dots + d_{\alpha - i}}{\alpha}; \\ \mathbf{if} \ \overline{N}_0 \ \mathbf{then} \\ & \downarrow S := S : \end{aligned}$ 9 10 11 $\mathbf{12}$ 13 $S := S_E;$ $\mathbf{14}$ SearchTarget(S, \overline{N}); 1516 else if \overline{A} is violated then $\mathbf{17}$ $S := S_{TA};$ $\mathbf{18}$ AdjustRegion($S, R_{\pm}, R_c, R_t, \overline{N}$); 19 else $\mathbf{20}$ if \overline{D} is violated then $\mathbf{21}$ $\mathbf{22}$ $S := S_{TD};$ AdjustDiag($S, D_D \pm \Delta_D, \overline{D}, \overline{N}$); $\mathbf{23}$ $\mathbf{24}$ \mathbf{end} end $\mathbf{25}$ 26 end 27 end

7. Comprehensive Experiment

Top view object detection using Deep Learning (DL) algorithms is able to solve complex problems or find the features which are complicated and hard even for a human to recognize. To derive the optimal outcome, the number of training datasets is of great importance. To recap, Chapter 4 presents the crane hardware setup and how to transform the real-world scenario into the virtual world to collect and augment data. The simulation platform does not only support concurrent research development but also remedies the data deficit in load-view object detection using deep learning methods. In addition to the dataset generation and simulation platform, the experimental results confirm that using preprocessed synthetic to train the data-driven detection method give clearly better Average Precision (AP) than using the raw one.

As can be seen in Chapter 5, the primary problem with the top view detection is scale variation. Despite less occlusion compared to the frontal view, the viewpoint of the object is changed and the target size becomes very small which make it more difficult for detectors to recognize, also for the human because of the small size and less information—only a head and a shoulder of a person can be seen from the top view. Furthermore, the size can be changed in different altitudes by either crane boom arm or zoom camera level. Although there are many deep learning object detection studies, the research in detecting objects from top view or aerial images remains limited, particularly in the construction domain. The applications using top view images include surveillance, traffic, inspection, and construction. The image sensor can be installed on Unmanned Aerial Vehicle (UAV), buildings, or large machines such as a mobile crane. To address the problem, there is great effort to augment training small-object size datasets for training to yield better accuracy of the data-driven methods [240]. However, there is room for improvement. Although using synthetic data helps the detectors, the detector remains intolerant to rapid change in object size due to altitude variation. In the mentioned chapter, the final network, RetinaNet-Resnet50 (R50), is selected based on speed over accuracy criteria, after the extensive data inspection.

In Chapter 6, we investigated the zoom crane camera and proposed the zoom controller method to tackle the scale variant issue in object detection using *adaptive zoom*. The method is able to support object detection by smoothly adjusting the zoom level to address



Figure 7.1: This figure mainly focuses on the content of Chapter 7. This chapter presents the integration of all proposed component from the previous chapters as a comprehensive experiment to improve the top view deep learning using the adaptive zoom camera. The contributions lie into three main parts i.e., dataset generation, load-view worker detection, and adaptive zoom control, also listed in Fig. 3.3. They are referred as blue column, green column, and red column, respectively. The upper horizontal band part, namely Chapter 4, Chapter 5, and Chapter 6, recaps individual component contributed. On the other hand, the lower horizontal part is the integration of all proposed component at the upper part and presented in this chapter.

the problem of inconsistent detected object size. Zoom camera is widely used in many applications such as construction sites or surveillance. The zoom feature is used to retain the image quality in a wide field of view or provide a close-up view for better recognition. For example, the zoom camera can enhance the load view for the crane operator to observe the safety proximity surrounding the load during the lift operation. The surveillance zoom camera can track the movement of suspects and zoom in to their faces for accurate facial recognition [241]. Nevertheless, it is crucial to control the zoom level automatically because it requires stable zoom control to hold each zoom constraint. In general, it is challenging to adjust the zoom level smoothly, given the noisy sensor data. Moreover, zoom control has rarely been studied directly. The research is mainly focused on adaptive zoom conditions rather than how to implement an effective zoom control. In other words, the previous studies [130, 238] do not provide evidence on how to conduct the zoom control to reach the desired zoom level, but only zoom constraints. For instance, the authors in [130] merely mentioned that the camera is zoomed to hold a defined pixel range. However, there are many factors that should be considered such as inconsistent detection, zoom speed, which can cause zoom oscillation.

The overview of Chapter 7 is illustrated at the bottom horizontal band in Fig. 7.1. In this



Figure 7.2: Workflow for the final network model test is shown in red frame. It extended from the workflow in Fig. 5.2. After the network was chosen in Sec. 5.4.2, the improvement of the network with larger data training was carried out. Once it was done, the final detector was then deployed on FINROC robotic framework under the mobile platform.

chapter, we eventually integrate all proposed components from the previous chapters as a comprehensive experiment in order to improve the top view deep learning object detection using the adaptive zoom camera.

In Chapter 5, we have already selected the network model RetinaNet-Resnet50 which will be further evaluated in Sec. 7.1 with much larger amount of dataset. Then, the Bounding Box (BBox) output from the RetinaNet-Resnet50 instead of AprilTag detector is forwarded to the zoom controller. The analysis of the zoom camera including the proof of the adaptive zoom control is already presented and verified as demonstrated in Sec. 6.7. To our knowledge, no prior studies have examined zoom function on the mobile crane to improve worker detection for safety monitoring. The application of the zoom controller logic for the top view worker detection is presented in Sec. 7.2.

7.1 Final Network Model Test

According to the network selection procedure in Sec. 5.4.2, the RetinaNet model was chosen for the final test. The selected R50 network model henceforth will be called *R50-final*.

Using the full dataset to train three networks would elongate the selection process which would hinder the speed of development of the project. Hence only the available limited-data was used during the network selection process presented in Chapter 5. The comprehensive dataset that will be used in this chapter is the result of data collection and annotation that has been gradually performed in the duration of four-year study.

The workflow of this section is presented in Fig. 7.2. The comprehensive training data contains both real-world dataset (i.e., $I_R = \{R00-S-C2, \ldots, R03-K-N1\}$) and synthetic dataset (i.e., $I_U = \{U00-S-C0, \ldots, U11-S-C4\}$) which were documented in Chapter 4.

In regard to using synthetic and real data for training, the study in [242] has shown that the combination between the rendered images and the real images outperformed training either real or synthetic data alone. The authors [242] assessed the quality of the rendered images if it is beneficial to the car viewpoint classification task. In addition, there are several factors involved such as rendering quality and size of the training set. Our synthetic datasets is blurred (pre-processed) in the same manner that had been done in Sec. 4.12.2. The averaging filter made our synthetic data look comparable to the real crane dataset. The investigation of synthetic data in Sec. 4.12.2 proved that the preprocessed synthetic data yields better training results than the original image. As can be seen, applying the basic image filtering method on the synthetic data could enhance the accuracy. The supporting evidence is further discussed in Sec. 4.11. In addition to the real crane data, the drone VCI public datasets were included. As described in Sec. 4.7.3, VCI-CITR and VCI-DUT shared the similar view as the load-view crane camera. The camera of these VCI datasets was always looking downward. The summary of the data, which is used to train and test, is shown in Fig. 7.3. Around 85% of all data came from the real world, while the rest were generated from Unreal Engine (UE).

The network configuration of R50-final were identical as in the selection procedure (see Table 5.3). Regarding the training results, we further brought up part of training sequences to discuss. The training results including the frame rate of each sequence are listed in Table 7.1. R50-final also performed well in the sequences where the workers were distinct from the background namely, R00-S-C2 and R01-S-C2. Despite having many training samples of R02-S-C2, the sequence is challenging for the detector, since the distance between workers and the camera is very high. Consequently, workers became very small and are nearly blended into the surface background because of the similar colors. For synthetic data, the R50-final detector obtained a moderate accuracy result.

For testing, there were three standard scenarios —a sequence from the university building (R06-K-N2) and two sequences from Steil (R10-S-C2, R16-S-CX).

In addition to the test result of R50-final, we reported the evaluation of two UAV Single Shot MultiBox Detector (SSD)-based detectors i.e., SSD-OkuPed [102] and SSD-IntelUav [243] as baseline. To the best of the author's knowledge, it is very hard to find a comparable trained model for top view object detection. Most studies merely presented the result of





Figure 7.3: Data statistic of R50-final. The data consists of real and synthetic data which are collected from the camera mounted either on crane (orange) or on the building (blue).

Figure 7.4: Training data set distribution of SSD-IntelUav. The training data contains 1312 color images. The seven object class consists of person(yellow), train(black), tree(green), truck(blue), boat(light blue), bus(orange), and car(grey). [243].

	#sample	AP	FPS
R00-S-C2	2139	76.68	5.12
R01-S-C2	592	90.91	5.57
R02-S-C2	22138	48.44	6.91
U02-S-C2	1636	43.38	4.50

Table 7.1: Result of R50-final training sequences.

the standard UAV dataset without publishing their trained models. Fortunately, we found these two SSD detectors which were relatively close to our application.

List of all detectors applied in testing can be found in Table 7.2. As discussed in Sec. 2.2.2, SSD-OkuPed is the person detector from UAV which was trained by the public dataset, Okutama. It was trained for 20k iterations with a learning rate of 1e - 4. On the other hand, SSD-IntelUav was the object detector on drone video from Intel[®]. The authors [243] described the chosen dataset for training consisting of 30 real-time aerial videos in seven classes, shown in Fig. 7.4. More than 50% of the dataset are object class *car*. The second-largest occupied object class is *person*. The amount of object class *tree* and *truck* are relatively same. The rest, which is *bus, boat, train*, take up small numbers. The training dataset contains 1312 color images. Nevertheless, the paper made no attempt to publish the dataset. The network base architecture of both SSD is VGG16 with the pre-trained data Microsoft Common Objects in COntext (MS COCO). The only difference between the two SSDs is input image size. The input image of SSD-OkuPed and SSD-IntelUav is 512×512 and 300×300 respectively. SSD512 provides better resolution and is more accurate than SSD300 [96].

Table 7.2: List of detectors for the test set comparison.

Detector	Base network	Pretrained data	Train data (FineTune)	Test data
R50-final	ResNet50	MS COCO	 VCI-CITR, VCI-DUT R00-S-C2,,R03-K-N1 U00-S-C0,,U05-C-N2, U10-S-C4, U11-S-C4 	• R06-K-N2 • R10-S-C2
SSD-OkuPed [102]	SSD512VGG16	MS COCO	Okutama	• R16-S-CX
SSD-IntelUav [243]	SSD300VGG16	MS COCO	30 real-time drone videos	

7.1.1 Results

AP in Table 7.3 can be achieved by the approximation of area under Precision-Recall (PR) curve. PR curves of each test sequences including the overall AP can be found in Fig. 7.5. The overall AP measures the accuracy of the detectors on all test sequences. The results of the three detectors will be reported in dataset order. In general, our R50-final was the one that obtained the most robust results with AP of 41.18. Despite the inferior AP, the result was directly in line with the previous studies of top view evaluation benchmark mentioned in Sec. 3.1 in which the best AP of object detection challenges on aerial images is approximately 30 percent. The Precision-Recall (PR) curve in Fig. 7.5e presented

Detector	R06-K-N2		R10-S-C2		R16-S-CX		Overall A D	
Detector	AP	FPS	AP	FPS	AP	FPS		
R50-final	70.48	6.58	68.53	6.50	17.91	6.35	41.18	
SSD-OkuPed	54.01	6.29	29.48	6.28	23.03	6.30	33.35	
SSD-IntelUav	24.88	6.15	9.09	6.17	≈ 0	6.19	9.09	

Table 7.3: Result of AP on test sequences.

overall outcome of the four different scenarios for all detectors. R50-final gained the best recall. In other words, the detector returned most of the relevant results (whether or not irrelevant ones were also returned). At the same time, SSD-OkuPed had the same PR curve pattern, but with less recall, that is to say, it produced less false positive than R50-final. SSD-IntelUav outputed many oversized BBoxes and tended to fail with small object. Alternatively, it could presumably mean that the object size in train data of the unpublished video sequences were bigger than the average object size in our application.

Challenging Dataset: Besides the three test dataset, we assessed the detector with the challenging sequence, R14-K-N2, which imitates typical construction scenario. The sequence includes construction tools of different sizes separated everywhere in the parking lot. The dataset is recorded on the bright sunny day. As the result, it causes shadow both on objects and persons. Moreover, construction activity such as throwing objects from one person to another is performed.

The snapshot of evaluation of all detectors can be found in Fig. 7.8. This test sequence appears as one of the tough challenging datasets, although it was recorded both without crane and not at Steil. R50-final results AP of 1.43, while the AP SSD-OkuPed and SSD-IntelUav approximate to zero. Throughout the sequence, it was occupied with construction equipment. The objects and workers have either short or long shadows under the harsh sunlight. In this scenario, all detectors basically misperformed. Instead, they recognized all construction resources and their shadows such as a cable reel, barrier, wheelbarrow, wooden pallet, etc. R50-final arduously localized the objects in spite of the highest AP. For instance, one of the workers, who stood nearly in the center of the image, threw an object at another. It caused the shape like another person in opening arm posture as illustrated in Fig. 7.8a. Unlike R50-final, SSD-OkuPed and SSD-IntelUav had less false positive errors.



Figure 7.5: Precision-Recall curves of the experiments. AP in Table 7.3 can be achieved by the approximation of area under PR curve. The red, green and blue legends are corresponded to R50-final, SSD-OkuPed, and SSD-IntelUav detector respectively.

- R06-K-N2—The snapshot of evaluation of all detectors can be found in Fig. 7.6. The three columns in the figure represent different zoom levels during the data recording. R50-final was able to detect workers with occasionally false alarms especially on a stopping sign on the ground, see Fig. 7.6a, 7.6b. SSD-OkuPed accurately detected objects in closer distance, see Fig. 7.6e. When the zoom level became less, the false negative error increased because the size of workers became smaller. SSD-IntelUav obtained not only false negative error, but also false positive. In summary, R50-final achieved the best Average Precision (AP) out of the three detectors. Fig. 7.5a indicates that R50-final got the best sensitivity rate and AP.
- R10-S-C2—The snapshot of evaluation of all detectors can be found in Fig. 7.7. As shown in Fig. 7.5b, R50-final performs well, giving good results by gaining the highest Area Under Curve (AUC). In contrast to SSD-OkuPed, its training data did not contain any crane hooks in the background, but the baseball field. As the result, the detector falsely outputed the crane hook as the worker, see Fig. 7.7d. SSD-IntelUav failed to detect the workers and consistently missed the workers of this sequence.
- R16-S-CX—The snapshot of evaluation of all detectors can be found in Fig. 7.9. Most of them failed to recognize the workers. In this sequence, the construction activities were imitated. For instance, two workers were rigging the load to the hook in the second column of Fig. 7.9, while there was a worker lying on his stomach in the third column. Nonetheless, none of the detectors detected workers during such activities. Moreover, the sunlight was very strong. The distance between the camera and the ground was high which is a result of the very small object size. SSD-OkuPed performed the best for this sequence; however, it failed to detect a person lying on the ground, which was part of its training set, see Fig. 7.9f. In comparison to SSD-OkuPed, R50-final obtained the AP in fractional difference. Similar to other sequences, SSD-IntelUav generally failed.



Figure 7.6: Detection result on the dataset R06-K-N2. The ground truth is shown in black BBox and the predicted result is shown in non-black BBox. The first row (a-c) are results of R50-final. The second row (d-f) are the result of SSD-OkuPed. The last row (g-i) are the result of of SSD-IntelUav. Each column (frame 36, 670, 1368) represents different zoom level.



Figure 7.7: Detection result on the dataset R10-S-C2. The ground truth is shown in black BBox and the predicted result is shown in non-black BBox. The first row (a-c) are results of R50-final. The second row (d-f) are the result of SSD-OkuPed. The last row (g-i) are the result of SSD-IntelUav. Three columns represent frame 358, 528, and 590 of the sequence respectively.



Figure 7.8: Detection result on the dataset R14-K-N2. The ground truth is shown in black BBox and the predicted result is shown in non-black BBox. The first row (a-c) are results of R50-final. The second row (d-f) are the result of SSD-OkuPed. The last row (g-i) are the result of SSD-IntelUav. Three columns represent frame 120, 259, and 460 of the sequence respectively.



Figure 7.9: Detection result on the dataset R16-S-CX. The ground truth is shown in black BBox and the predicted result is shown in non-black BBox. The first row (a-c) are results of R50-final. The second row (d-f) are the result of SSD-OkuPed. The last row (g-i) are the result of of SSD-IntelUav. Three columns represent frame 1318, 1384, and 2201 of the sequence respectively.

Regarding the speed performance, all three detectors were comparable. By referring our results to the similar study in operator's Situational Awareness (SA) [244], we hope to verify our time performance that it is capable to notify the operator in the real application. Fang et al. [244] provided the measurement of operator's SA of the crane assistance system for the blind-lifting. The system provided 3D environment surroundings, crane motion using crane camera, Inertial Measurement Unit (IMU), and laser scanner. Given the lifting task, the crane operator gave an acknowledge when he saw the obstacles and assessed the proximity during the measurement. The minimum and maximum response time (or time to acknowledge) were 1.0 and 9.5 seconds respectively, while the average response time was 4.5 seconds. As can be seen, the study case of this SA measurement is equivalent to our application as the operator should recognize nearby worker-on-foot during the lift operator. Consequently, the average response time in our application aligned in the range of operator response time. In particular, our estimated throughput of detection pipeline for the operator was 15.71 milliseconds which was around 6 to 60 times faster than the operator's response time in [244].

7.2 Top View Worker Detection Using Adaptive Zoom

Finally, we demonstrate the practical experiment of the top view worker detection using adaptive perception. The camera position on the crane is simulated by mounting the camera from the rooftop of a building. The camera looks down to the parking lot which has an even surface. The approximate distance from the camera to the ground is 22 meters which is equivalent to the height of a 6-story building. The $D_D \pm \Delta_D$ is initially set to 60 ± 10 pixels. During the experiment, four workers walk into the camera field of view. In spite of worker safety requirements, the workers do not wear any protective gear or Personal Protective Equipment (PPE). Most of the traditional detector methods exploited the PPE appearance to ease the detector which allowed the detector to see the target better [118]. However, there are many incidents of non-compliant workers violating the rules [245]. Thus, we have chosen a generic construction environment where there is no restriction on worker cloth.

7.2.1 Result

The graph in Fig. 7.10 shows how the zoom level Z adapted to the detected workers. The figure consists of two main parts, which are snapshot image frames and the zoom control result. In the image section, the 1st and 3rd row of images show the detection result while the 2nd and 4th row display the same image frame with the result for zoom control logic.

At t_1 , Z_t swiftly increased because the average overall diagonal size \overline{D} was below the floor of D_D . At t_2 , Z_t is stable because the size preservation was complied.

At t_3 , the border violation occurred. Two of the workers walked near toward the prohibited region R_+ . One worker was toward the top right corner and the other was toward the bottom of the image frame. Consequently, Z_t decreased in between t_3 and t_4 until the workers stayed inside the inner region R_- . After the area adjustment, the zoom level went up because of the size violation before t_4 where ZoomController reached the stable state S_E . At t_5 including the nearby period, the zoom level was gently changed and kept \overline{D} in the desired diagonal range because of the size change. Likewise, the border violation



Figure 7.10: ZoomController experiment with the worker detection from load-view crane camera. The upper image row of each time point shows the raw data from the detector while the lower image shows the visualized result from the ZoomController. The red border identified the area violation which means there is an intersection area between \overline{A} and R_+ . The complete state of box detection and camera state can be further found in Fig. D.3 and D.4.

again happened as the border showed in red at t_6 . One worker walked toward the top of the frame. As the result, the camera zoomed out and later Z_t became consistent at t_7 . At t_8 , the camera again plunged because the parameter D_D was configured to a smaller value. The further detail of the substate of finite state machine (FSM) can be found in Appendix D.

In summary, our results demonstrate that adaptive zoom control can improve the quality of data-driven worker detection. To evaluate the zoom control logic, we replaced the worker detector with AprilTag detector which provides a reference target. The verification performs well, giving the correct result as is defined in the FSM. The zoom level was adjusted without jitters. The camera first could not detect the target because of the too-small size object. With adaptive zoom control, the target was able to be recognized from distance. When comparing AprilTag results to the worker detector, it must be pointed out that there is a lot of noise from the sensor data. In other words, the RetinaNet could not constantly detect all four workers despite the MA filter. However, the *ZoomController* functions without any zoom oscillations. The controller is able to handle the side effect of the detector on less powerful hardware including the video transmission introduced a delay of approximately two seconds in our experiment. Our proposed method can be adjusted to this lagging via the parameters Π .

A major source of limitation is the lack of camera sensor that measures the zoom level. We use an approximate estimation of zoom level Z by incrementing the Z counter up and down when zooming in/out is executed. Additionally, the crane control information of the operator e.g., hook length measurement and boom angle would be definitely beneficial to refine the detector. For instance, the zoom controller can be performed based on hook cable length in addition to the proposed criteria.

7.3 Discussion

As the final network test, R50-final was intensively trained by real-world datasets and preprocessed synthetic datasets which were generated from our simulation platform. The network was evaluated through the four test scenarios, which consisted of many challenging conditions (e.g., strong sunlight, short and long shadows, construction resources looked alike worker from the top view.) and actions (e.g., hoisting the load, person lying on stomach, workers throwing objects to the another). Despite no published pretrained model for load-view crane object detection in previous studies, we additionally provided the evaluation results of two aerial SSD detectors (i.e., SSD-OkuPed, SSD-IntelUav) using the same four scenarios as the reference to our R50.

The final result of the network model test found clear support that R50-final was able to yield the most robust results with AP of 41.18. Unlike the evaluation of the frontal view object detection, the best AP of the aerial view evaluation challenges is 30 percent on average as mentioned in Sec. 3.1. Importantly, it is able to promptly output nearby workers-on-foot within the range of the crane operator SA response time. From this standpoint, this result can be considered as the promising one.

Later, we applied the adaptive zoom control of the load-view crane camera for worker detection. This is an important finding in the understanding of how to handle the zoom control to reach the zoom criteria. We exploited the zoom mechanism which exists in typical mobile crane cameras. The proposed method adopts the Mealy FSM to observe and determine the zoom command which suitable for the given situations, namely target loss, target area preservation, and target size preservation. The state definition is characterized by the three scenarios. The evaluation is first verified in Sec. 6.7 by using the reliable and controllable detector, AprilTag. Our proposed zoom control method is able to smoothly adapt to the problem of DL object detection, which is inconsistent detection and detecting small size objects.

Further studies should investigate sensor fusion with more access to crane information for zoom control improvement. An additional monocular camera can be installed nearby the zoom camera. The monocular camera provides overview information to the zoom camera. Although the zoom camera status is in maximum zoom in, the overview camera can notify *ZoomController* of the zoom camera if there is a new incoming target then the zoom camera can zoom out. Moreover, the position of workers including the velocity in world coordinate can be estimated by camera projection and object tracking. Hence, the risk of each worker can be assessed for the operator safety assistance system. For instance, if the worker walks away from the crane, the risk of the worker getting hit by the crane is low.

8. Conclusion and Outlook

This thesis aimed to develop top view DNN-based object detection using an active perception with specialization in construction environment as a vision-based operator safety assistance. Unreal Engine (UE), the simulation tool, was employed to support not only the training data shortage, but also as an experimentation platform. The active zoom feature of the off-the-shelf image sensor was utilized to enhance the quality of recognition in addition to the data-driven detection method.

Based on the literature surveys in Chapter 2, it can be concluded that detecting objects from the aerial images is challenging and the research activities in this area, particularly construction remains far behind, unlike passenger autonomous vehicles. Hence, this thesis chose to study the crane load-view object detection in construction domain as a use case, which contains most of the common challenges in other applications such as complex environment and rapid changes in object scale.

First of all, testing every implementation of algorithms on heavy-duty vehicles is not trivial. Working with heavy machinery is very dangerous because any small mistakes can lead to high fatality. It requires a safe, efficient working environment including an experienced operator.

It is therefore necessary to have the simulation platform for the experimentation in order to avoid accidents. Furthermore, it enables concurrent development, which allows multiple researchers simultaneously work on the same or different components independently. Consequently, the manufacturing costs is lowered while the productivity becomes higher.

By developing the simulation platform in UE, this thesis has shown how flexibility and photorealism in building up the environment in Sec. 4.6. The adaptability of new surrounding, vehicles, or various weather conditions can be done swiftly.

Conventional object detection methods, which cling to the object appearances or color, could fail because of Personal Protective Equipment (PPE) noncompliance or dynamic background. Hence, this work adopted the top view object detection using deep learning methods to learn hidden patterns in which human may not be able to solve from the data by themselves. To put it another way, it is difficult for human to determine the key information of the target image for tackling the top view object detection task.

To yield the optimal results of the deep learning object detection, it is required to have massive amounts of data. However, there is a great absence of training data for the deep learning object detection in previous studies, especially from top view. Collecting data is laborious process and take a lot effort such as mount and dismount sensors from the huge industrial machines or finding the large area for evaluation. The data should contain all environmental conditions and importantly they must be correctly annotated in order not to mislead the learning algorithms.

As another goal of the proposed simulation platform, this thesis demonstrated how to exploit the platform to swiftly produce numerous of training data and simultaneously their groundtruth without any localization error, see Sec. 4.8. Our virtual platform is able to generate the groundtruth much faster than manual annotation.

By investigating the synthetic data from our proposed simulation platform in Sec. 4.12, we tested the two following hypotheses. First, the preliminary examination of the pretrained UAV object detection model with the crane load-view image data shows that the UAV images and load-view images share basic image features. The result in Sec. 4.12.1 clearly demonstrated that the UAV detector was able to localize the target worker despite some difficulties e.g., short shadow, which is a lying down person look-alike. The present findings confirm that the synthetic data can be employed for training deep learning object detection model.

Second, this thesis has shown that the synthetic data potentially accommodates the top view worker detection. To make the synthetic data be comparable to the real world data, our experimental results in Sec. 4.12.2 indicated that the data should be preprocessed using average filter before model training. Overall, our results demonstrate preprocessed synthetic data improve the accuracy gain more than 10%.

Choosing the network model that is suitable to our application is crucial. Based on the two criteria —speed and accuracy discussed in Sec. 5.3, RetinaNet-Resnet50 (R50) is the preferable network model choice as it is able to handle small object size and class imbalance issue. The investigation of the collected crane data in this work indicates that most of the BBox size are small (i.e., $\leq 20 \times 20$ px). Additionally, in our application, speed is considered as a top priority due to two reasons i.e., safety and the activeness of the image sensor.

After intensive training R50-final using both synthetic data and real world data, the speed evaluation result shown in Sec. 7.1.1 is well confined to the Situational Awareness (SA) measurement of the crane assistance system in the literature [244]. In general, the Safety Response Time (SRT) of the safety assistance system in the industry or autonomous vehicle should be very low. In other words, the faster the response time, the lower the accident risk.

Detection speed is crucial for automatic zoom control of crane camera. In this thesis, we proposed the closed-loop worker detection system which required prompt response between the sensing (i.e., detector) and control (i.e., zoom camera control) to maintain the quality of the top view detection. The detector should instantly output the sensor data to the zoom control module to process, while the zoom control command should be fed back quick enough to adjust the load-view image sensor.

Unlike the image sensor on UAV, the altitude of the crane load-view camera is frequently changed by crane boom arm or camera zoom level itself. Instead of the top view traditional

detection, we employed the deep learning method to mainly solve feature selection problem. Nevertheless, small-scale Bounding Box (BBox) samples which were collected from the crane camera, does not present adequate information. In other words, it is difficult for the deep learning approach to extract features from the small and low resolution BBox to define target representation of the worker. Despite surpassing other state of the art UAV which is shown in the evaluation report of Sec. 7.1, the quality of the deep learning detection from top view is comparatively much lower than frontal view.

Finally, this thesis has shown the recent adaptive zoom control method using Mealy finite state machine (FSM). The adaptive controller method is correctly verified by using AprilTag as a reference target. According to the analysis of adaptive zoom in Chapter 7, the zoom control is not only smoothly regulated, but also preserve the BBox size and allows the deep learning detector to perceive more information.

8.1 Outlooks

For the simulation platform, the physical properties such as crane load capacity can be included in order to make the virtual environment naturalistic. Object tracking and activity recognition could help the detector however they could decelerate the system throughput.

Due to unpublished and inaccessible camera control information e.g., zoom level, we use discrete control input, where the zoom level switch abruptly between two states i.e., zoom-in and zoom-out. With the camera control access, future studies exert control input continuously which enables the application of variety of controllers such as PID. The use of P-controller can improve the controller response based on the error size, in order to control the zoom speed e.g., the higher the error is, the faster the zoom speed becomes (i.e., $z_{t+1} = z_t + e \times \Delta z$).

In this work, the distance between the ground and camera fixates or the crane neither extended nor retracted during adaptive zoom control experiment. The further study can be explored on the different distance between the zoom camera to the ground to improve the deep learning model in this work or possibly adjust to additional models based on each distance range.

It will be important that future research investigate additional camera as an overview camera and collaborate between these two cameras. Albeit the improvement of the detection using the adaptive zoom camera, the overall information is lost when the camera zooms in. With the zoom control access including the camera detail, future research could continue to explore the target observation in 3D-world coordination.

Lastly, further research is needed to conduct the standard commercial experiments and concept for certification of automated construction safety assistance system. Also, more construction industry experts should take part in the evaluation process.

A. Acronym

ANN	Artificial Neural Network	17
AOV	Angle of View	92
\mathbf{AP}	Average Precision	107
AUC	Area Under Curve	114
AV	Autonomous Vehicle	39
ALVIN	NN Autonomous Land Vehicle In A Neural Network	4
BIM	Building Information Modeling	1
BLS	Bureau of Labor Statistics	19
BBox	Bounding Box	109
CAN	Controller Area Network	131
CARL	A Car Learning to Act	38
CNN	Convolutional Neural Network	19
MS CO	OCO Microsoft Common Objects in COntext	111
cGAN	Conditional Generative Adversarial Network	86
\mathbf{DFT}	Discrete Fourier Transform	64
\mathbf{DL}	Deep Learning	107
DNN	Deep Neural Network	83
DPM	Deformable Part Model	19
\mathbf{DR}	Domain Randomization	44
FINRO	OC Framework for Intelligent RObot Control	29
FOV	Field of View	36
FPN	Feature Pyramid Network	83
FPS	frames per second	18

\mathbf{FSM}	finite state machine	120
F50	FasterRCNN-Resnet50	90
F101	FasterRCNN-Resnet101	90
Colab	Google Colaboratory	83
GAN	Generative Adversarial Network	44
\mathbf{GD}	Gradient Descent	87
HOG	Histogram of oriented gradients	15
HAL	Hardware Abstraction Layer	93
IoU	Intersection over Union	61
IMU	Inertial Measurement Unit	119
HRI	Human-robot Interaction	91
LBP	Local Binary Pattern	16
LiDAF	t Light Detection and Ranging	9
\mathbf{ML}	Machine Learning	1
\mathbf{MS}	Magnitude Spectrum	. 155
\mathbf{NN}	Neural Network	17
NMS	Non-Maximum Suppression	13
NST	Neural Style Transfer	43
MoG	Mixture of Gaussians	13
OCM	Orientation Code Matching	18
OSHA	Occupational Safety and Health Administration	1
OSM	OpenStreetMap	49
OpenC	\mathbf{CV} Open source computer vision	96
PPE	Personal Protective Equipment	. 119
\mathbf{PR}	Precision-Recall	. 111
\mathbf{PTZ}	Pan-tilt-zoom camera	9
PANet	Path Aggregation Network	22
R-CNI	N Region Based Convolutional Neural Networks	20
RPN	Region Proposal Network	76
RFID	Radio Frequency Identification	80
RoI	Region of Interest	76
ResNe	t Residual Neural Network	20
$\mathbf{R50}$	RetinaNet-Resnet50	. 107
RMSE	Root Mean Square Error	65
RRLat	b Robotics Research Lab	. 133

\mathbf{SA}	Situational Awareness
\mathbf{SAR}	Search and Rescue
SIFT	Scale-Invariant Feature Transform
Social	GAN Socially Acceptable Trajectories with Generative Adversarial Networks 21
SORT	Simple Online and Realtime Tracking21
SRT	Safety Response Time
SGD	Stochastic Gradient Descent
\mathbf{SSD}	Single Shot MultiBox Detector
\mathbf{SVM}	Support Vector Machine
SPP	Spatial Pyramid Pooling22
\mathbf{SDD}	Stanford Drone Dataset
\mathbf{TF}	TensorFlow
\mathbf{TL}	Transfer Learning
TPU	Tensor Processing Unit
TUKL	TU Kaiserslautern
UAV	Unmanned Aerial Vehicle107
UE	Unreal Engine
UWB	Ultra-wide Band
V-REF	• Virtual Robot Experimentation Platform
\mathbf{VR}	Virtual Reality
YOLO	You Only Look Once

B. Building Simulation Environment

This appendix describes the technical implementation detail of turning real world environment into the simulation. The features of main constructed virtual elements at Steil site such as a mobile crane and workers, are described as follows.

B.1 Crane and Sensor

In this work, Grove GMK3060[246] is given as a 3D telescopic crane model for the simulation. Although Grove GMK3060 is smaller than Liebherr LTM1130[247] which was the actual crane that we experimented with, their functions are the same in principle e.g., hoisting, swing. This virtual crane can drive in basic movement. It contains three crane common mechanisms as the real crane, namely hoisting, swing, and travelling. For hoisting, the boom can extend and retract. The hook block can hoist up and down with or without attached load. Different types and size of load can be changed. The second mechanism is swing. Based on the robot coordinate system O_R (see Fig. 4.18), the uppercrane part can rotate around z-axis or yaw angle. The uppercrane can rotate around y-axis or pitch angle. Finally, the basis travelling can be performed such as driving forward/backward, steering and braking. The outrigger, which is an extended beam to stabilize the crane, can be applied during lifting or removed when the crane truck travels. The crane can be controlled manually via either joystick or control input sequence. Similar to a PC game control, the crane control shortcut keys can be assigned which is listed in Table B.1. The crane can be controlled manually via joystick or input sequence. The example of sequence maneuvers temporally based on given controls as shown in Fig. B.1. As a result, this allows the researchers generate the dataset or study the closed loop scenario repetitively such as a routine lift plan.

For the virtual image sensor, the MC5200 motor zoom camera is implemented in the same manner as the actual camera. The virtual camera is mounted with the pendulum bracket at the tip of the boom. The physics properties in Unreal Engine (UE) allows the zoom camera to have the effect of pendulum. Therefore, the pendulum bracket can be swung by the action of gravity and acquired momentum. The zoom level can be adjusted by changing FOV. In addition, the zoom speed can be changed similarly as the Controller

Key	Control
W, S	Uppercrane pitch (up, down)
A, D	Uppercrane yaw (left, right)
E, R	Extend, Retract
Arrow key	Vehicle control
В	Brake
С	camera toggle view (vehicle view, operator view, driver view, load view). For the vehicle view, move the mouse to look up-down / left-right.
Num 2, Num 8	Hook drop (down, up)
P, Shift+P	Spawn a load, Drop the load
O, Shift+O	Spawn a stage3-outrigger, Remove the outrigger
Num 3, Num 9	down (toward the crane), up (outward the crane) over the pitch of 2nd zoom camera. Make to enable "enable ctrl zoomcam rot"
Num 1, Num 7	Possess the pawn, Unpossess the pawn (BP_pawn_gmk3060_v01 for "steil_1" map only)
Num 0	Capture groundtruth manually (Snapshot).
Num.	Start record groundtruth (long sequence).

Table B.1: Key assignment to control the virtual mobile crane in UE.



Figure B.1: Test control sequence for a crane lift trajectory. x-axis is a step count and y-axis is time.

Area Network (CAN) pulse speed in the actual camera presented in Sec. 6.2. To have better observation during the testing in simulation world, additional monocular cameras are additionally mounted on several positions e.g., operator and driver cabin, see Fig. 4.19.

B.2 Connection to Finroc

The connection of data ports between UE and FINROC¹ are created. The port name list can be found in Fig. B.2. The FINROC control panel can be seen in Fig. B.3. The port name list can be found in Fig. B.2. A port in FINROC is a type of data structure for data exchange. They are basic sensor and control ports that suffice the experiments. Sensor and control port names can be categorized into two main groups —crane and camera. First, the crane group consists of three mechanisms which are hoist, swing, and travel. The hoist mechanism is related to parts that are used for lifting via hook blocks or sets of cables. The swing mechanism is a upperworks or superstructure which can freely rotate 360° , while the travel mechanism is associated with the main truck. The truck carries the upperwork part from one place to another. The second group is the camera. The image frame and camera FOV as sensor data are sent from UE to FINROC framework while the control value of crane and camera zoom level in UE can be configured and input via FINROC. As a result, these basic controls allowed us to experiment arbitrarily.



Figure B.2: UE - FINROC portname interface. C - Camera mechanism, H - Hoist mechanism, S - Swing mechanism, T - Travel mechanism.

 $^{^{1}}$ A real-time robotic framework based on a systematic design. It has been developed at the Robotics Research Lab (RRLab) of TU Kaiserslautern, Germany since 2008[127]



Figure B.3: Connection between FINROC and UE. On the left, the figure shows the FINROC control panel from the crane including the zoom camera feed from UE while on the right, it shows the UE viewport panel. 1 - zoom camera control, 2 - image stream from UE, 3 - crane control panel, 4 - Grove GMK3060 in UE.



Figure B.4: *Mixamo* interface. The user first must select the character appearance by choosing from the skin or skeletal mesh options provided in (c). Once the look is chosen, an individual animation sequence of the character can be rigged from the animation options in (a). In (b), it displays a final look of the character in the character panel.

B.3 Human Characters

Virtual characters² are one of the most important simulated components because they are the targets for worker detection. In this virtual platform, the virtual characters are obtained from *Renderpeople*, *Adobe Fuse CC*, and *Mixamo*, see Fig. B.6. Fig. B.4 depicts *Mixamo* feature which allows user to pick character and its individual animation sequence.

The basic properties of the imported character are following.

• Appearance —Besides the clothes and their colors which can be randomly changed, the virtual character is able to hold any objects while moving e.g., a man in Fig. B.6b is holding an object while walking, or a worker in Fig. B.6g is pushing a wheelbarrow.

 $^{^{2}}$ A virtual character in game engine is a person or any other entity acting in a game.
- Animation The movement of female and male characters were made separately. The animation can be reused between characters which have the same skeleton or their animation is retargeted. The retargeting process maps the original skeleton to the target skeleton. The movement speed of the character can be adjusted. For the unit measurement in UE, it uses Unreal Units (uu). One Unreal unit is equivalent to 1 cm by default. The animation of the character is associated to the speed. For instance, the speed of idle movement is 0 uu/s. The speed of walking and running are 375 and 600 uu/s respectively. To have a smooth movement of the character, two or more animations are blended via Blend Space asset.
- Sensory —To provide the sensory to the characters, there are several methods in UE such as Pawn Sensing and LineTraceByChannel. This allows the characters to perform collsion avoidance or interact with other characters or objects.

After each character is properly configured, the crowd simulation can be implemented, shown in Fig. B.5. Instead of placing character to the map one by one, the crowd simulation generates many characters into the map at once. The character manager handles how character move and interact with the others or obstacles.

To spawn characters on the map, the characters are initialized or created in the 3D boundary volume is shown the green area in Fig. B.5a. Such the area can be defined via NavMeshBoundsVolume. It is used to define where your AI players and controller will be able to go. In other words, the volume can restrict the area where the AI characters can move. It also can create characters from the area or destroy them if they travel into that defined area.

B.4 Environment

The overall environment shown in Fig. 3.4. Given the target area in Steil, we started to select the area in OSM, which provide 3D models of the buildings, see Fig. B.7a. Later, the 3D model map is imported into *Blender*, where we assign and color the surfaces. In UE, we employed UE Plugin called OpenStreetMap importer to bring the *Blender* model into the game engine. Once the map is built, we started to import construction assests (shown in Fig. B.8) and some decorations such as trees and vegetation. These decorations can be obtained from the UE Plugins or UE4 Marketplace. The landscape is filled up with the virtual workers, construction entities and supplies, , etc. Fig. B.7b depicted the complete decoration of Steil from top in UE. The virtual persons in the scenarios randomly move with the collision avoidance feature. In addition, the person appearance is diverse including the worker with safety helmet.



(c) Spline walk.

(d) Walk between points.

Figure B.5: Crowd simulation and Spline walk in UE. The figure depicts result of character generation features in this thesis to support synthetic data augementation. There are four main features i.e., (a) one-way or bidirectional walk, (b) random walk, (c) spline walk and (d) walking between defined points. (a) The green rectangles are NavMeshBoundsVolume, which are block areas used in UE. They are the area where the character are spawned. After the characters are generated, they can walk either toward the top of the figure or the bottom of the figure. (b) The characters are spawned on any random position of the floor. (c) After creating spline on the floor which defines the valid path for the character, the characters are generated in different appearances and walk along the spline. (d) In stead of spline path, the waypoints are defined. Character can then walk randomly among the waypoints.





Figure B.6: Virtual characters in UE. (a) Standing construction worker wearing PPE. (b) Working in carrying an object post. (c) Standing worker without emergency vest. (d) and (e) illustrate different imported characters from *Renderpeople* [248], *Adobe Fuse CC* [249] and *Mixamo* [250]. (f) and (g) depicted the virtual characters perform construction activities such as working with devices or pushing a wheelbarrow.



Figure B.7: Steil top view presented in different tools which is (a) OSM, (b) UE, and (c) Google Maps [251] *Map data*@2021 *Google, Trier.* The number label on each location is corresponding to the same place while a symbol \boxtimes is the location where the crane is, see Fig. 4.16.



Figure B.8: Examples of UE assets for construction props are imported to decorate our simulated construction area such as wheel barrows, jack hammer, cement mixer, wooden beam, etc [252, 253].

C. Network Configuration

This appendix further elaborates the network configuration which are used in the network selection process in Sec. 5.4.

C.1 FasterRCNN-Resnet50

FasterRCNN-Resnet50 is a two-stage detector. The model first proposes the region of interestes by using RPN instead of *selective search*. It is the third generation of the R-CNN with a ResNet-50-FPN backbone.

C.1.1 Model configuration

```
model {
1
       faster_rcnn {
2
         num_classes: 1
3
         image_resizer {
4
           keep_aspect_ratio_resizer {
\mathbf{5}
6
              min_dimension: 600
\overline{7}
              max_dimension: 900
            }
8
9
10
         }
         feature_extractor {
11
            type: 'faster_rcnn_resnet50'
12
            first_stage_features_stride: 16
13
14
         first_stage_anchor_generator {
15
            grid_anchor_generator {
16
              scales: [0.0625, 0.125, 0.25, 0.5, 1.0, 2.0]
17
              aspect_ratios: [0.5, 1.0, 2.0]
18
              height_stride: 16
19
              width_stride: 16
20
            }
21
         }
22
          first_stage_box_predictor_conv_hyperparams {
23
```

```
op: CONV
24
           regularizer {
25
             12_regularizer {
26
               weight: 0.0
27
             }
28
            }
29
           initializer {
30
             truncated_normal_initializer {
31
32
               stddev: 0.01
33
             }
           }
^{34}
         }
35
36
         first_stage_nms_score_threshold: 0.0
37
         first_stage_nms_iou_threshold: 0.7
         first_stage_max_proposals: 300
38
         first_stage_localization_loss_weight: 2.0
39
         first_stage_objectness_loss_weight: 1.0
40
         initial_crop_size: 14
41
         maxpool_kernel_size: 2
42
43
         maxpool_stride: 2
         second_stage_box_predictor {
44
           mask_rcnn_box_predictor {
45
             use_dropout: false
46
             dropout_keep_probability: 1.0
47
             fc_hyperparams {
48
49
               op: FC
50
               regularizer {
51
                  12_regularizer {
52
                    weight: 0.0
53
                  }
               }
54
55
               initializer {
56
                  variance_scaling_initializer {
57
                    factor: 1.0
                    uniform: true
58
                    mode: FAN_AVG
59
                  }
60
61
               }
62
             }
           }
63
         }
64
         second_stage_post_processing {
65
           batch_non_max_suppression {
66
             score_threshold: 0.0
67
             iou_threshold: 0.6
68
69
             max_detections_per_class: 100
             max_total_detections: 300
70
71
           }
72
            score_converter: SOFTMAX
         }
73
74
         second_stage_localization_loss_weight: 2.0
75
         second_stage_classification_loss_weight: 1.0
76
       }
     }
77
```

C.1.2 Training Configuration

```
train_config: {
 1
 \mathbf{2}
       batch_size: 1
3
       optimizer {
 4
         momentum_optimizer: {
\mathbf{5}
           learning_rate: {
             manual_step_learning_rate {
 6
 7
                initial_learning_rate: 0.0001
                schedule {
8
                  step: 500000
9
                  learning_rate: .00001
10
                }
11
                schedule {
12
                  step: 700000
13
                  learning_rate: .000001
14
                }
15
             }
16
           }
17
18
            momentum_optimizer_value: 0.9
19
         }
20
          use_moving_average: false
^{21}
       }
^{22}
       gradient_clipping_by_norm: 10.0
       fine_tune_checkpoint: "/content/gdrive/My Drive/train/faster_rcnn_resnet50_coco_2018_01_28/model.ckpt"
^{23}
       from_detection_checkpoint: true
^{24}
       num_steps: 800000
25
26
       data_augmentation_options {
27
         random_horizontal_flip {
28
         }
29
         random_vertical_flip {
         }
30
         random_jitter_boxes{
31
              ratio: 0.2
32
          }
33
       }
34
     }
35
```

C.1.3 Evaluation Configuration

```
eval_config: {
1
       metrics_set: "coco_detection_metrics"
2
       use_moving_averages: false
3
       num_examples: 1463
4
       num_visualizations : 1463
\mathbf{5}
       visualization_export_dir : "EXPORT_DIR"
6
7
       visualize_groundtruth_boxes : true
8
       groundtruth_box_visualization_color: 'blue'
9
       min_score_threshold : 0.1
     }
10
```

C.2 FasterRCNN-Resnet101

FasterRCNN-Resnet101 is a two-stage detector. The model first proposes the region of interestes by using RPN instead of *selective search*. It is the third generation of the R-CNN with a ResNet-101-FPN backbone.

C.2.1 Model configuration

```
model {
1
       faster_rcnn {
2
         num_classes: 1
3
         image_resizer {
4
           keep_aspect_ratio_resizer {
\mathbf{5}
             min_dimension: 600
6
              max_dimension: 900
7
8
           }
9
         }
10
         feature_extractor {
            type: 'faster_rcnn_resnet101'
11
            first_stage_features_stride: 16
12
         3
13
         first_stage_anchor_generator {
14
           grid_anchor_generator {
15
              scales: [0.0625, 0.125, 0.25, 0.5, 1.0, 2.0]
16
              aspect_ratios: [0.5, 1.0, 2.0]
17
18
             height_stride: 16
19
              width_stride: 16
           }
20
         }
21
         first_stage_box_predictor_conv_hyperparams {
^{22}
^{23}
            op: CONV
            regularizer {
^{24}
              12_regularizer {
25
                weight: 0.0
26
              }
27
            }
28
            initializer {
29
              truncated_normal_initializer {
30
                stddev: 0.01
31
32
              }
33
           }
         }
34
35
         first_stage_nms_score_threshold: 0.0
36
         first_stage_nms_iou_threshold: 0.7
37
         first_stage_max_proposals: 300
38
         first_stage_localization_loss_weight: 2.0
39
         first_stage_objectness_loss_weight: 1.0
40
         initial_crop_size: 14
^{41}
         maxpool_kernel_size: 2
42
         maxpool_stride: 2
43
         second_stage_box_predictor {
44
           mask_rcnn_box_predictor {
              use_dropout: false
45
              dropout_keep_probability: 1.0
46
47
              fc_hyperparams {
                op: FC
48
```

```
regularizer {
49
50
                  12_regularizer {
51
                    weight: 0.0
52
                  }
               }
53
54
               initializer {
55
                  variance_scaling_initializer {
56
                    factor: 1.0
                    uniform: true
57
                    mode: FAN_AVG
58
                 }
59
               }
60
             }
61
           }
62
         }
63
         second_stage_post_processing {
64
           batch_non_max_suppression {
65
             score_threshold: 0.0
66
             iou_threshold: 0.6
67
             max_detections_per_class: 100
68
             max_total_detections: 300
69
           }
70
71
            score_converter: SOFTMAX
         }
72
73
         second_stage_localization_loss_weight: 2.0
74
         second_stage_classification_loss_weight: 1.0
       }
75
     }
76
```

C.2.2 Training Configuration

```
train_config: {
 1
       batch_size: 1
 2
3
       optimizer {
         momentum_optimizer: {
 4
           learning_rate: {
5
              manual_step_learning_rate {
6
                initial_learning_rate: 0.0001
 7
                schedule {
 8
                  step: 500000
9
10
                  learning_rate: .00001
11
                }
^{12}
                schedule {
13
                  step: 700000
14
                  learning_rate: .000001
15
                }
16
              }
^{17}
           }
18
           momentum_optimizer_value: 0.9
19
         }
20
          use_moving_average: false
^{21}
       }
22
       gradient_clipping_by_norm: 10.0
23
       fine_tune_checkpoint: "/content/gdrive/My Drive/train/faster_rcnn_resnet101_coco_2018_01_28/model.ckpt"
       num_steps: 800000
^{24}
```

```
from_detection_checkpoint: true
25
       data_augmentation_options {
26
         random_horizontal_flip {
27
         }
28
29
         random_vertical_flip {
         }
30
         random_jitter_boxes{
31
32
             ratio: 0.02
33
         }
^{34}
       }
     }
35
```

C.2.3 Evaluation Configuration

```
1
     eval_config: {
2
       metrics_set: "coco_detection_metrics"
3
       use_moving_averages: false
       num_examples: 2355
4
\mathbf{5}
       num_visualizations : 2355
       visualization_export_dir : "EXPORT_DIR"
6
       visualize_groundtruth_boxes : true
7
       groundtruth_box_visualization_color: 'blue'
8
       min_score_threshold : 0.1
9
     }
10
```

C.3 RetinaNet-Resnet50

RetinaNet-Resnet50 outperforms Faster R-CNN by using focal loss and featurized image pyramid. Focal loss is designed to penalize the background class or easy-examples which is easy for the network to learn by downing the weight. Featurized image pyramid is much faster than the traditional image pyramid. Similar to image pyramid in SSD, featurized image pyramids resolves object detection at different scales in faster manner and less computation because CNN utilized pyramid structure.

C.3.1 Model configuration

```
model {
1
         ssd {
2
3
             inplace_batchnorm_update: true
             freeze_batchnorm: false
4
             num_classes: 1
5
             box_coder {
6
             faster_rcnn_box_coder {
7
                  y_scale: 10.0
8
                  x_scale: 10.0
9
                  height_scale: 5.0
10
                  width_scale: 5.0
11
             }
12
             }
13
```

```
matcher {
14
15
              argmax_matcher {
16
                  matched_threshold: 0.5 \,
17
                  unmatched_threshold: 0.5
18
                  ignore_thresholds: false
19
                  negatives_lower_than_unmatched: true
20
                  force_match_for_each_row: true
^{21}
                  use_matmul_gather: true
              }
22
              }
^{23}
              similarity_calculator {
^{24}
              iou_similarity {
25
              }
26
              }
27
28
              encode_background_as_zeros: true
              anchor_generator {
29
              multiscale_anchor_generator {
30
                  min_level: 3
31
                  max_level: 7
32
                  anchor_scale: 4.0
33
                  aspect_ratios: [1.0, 2.0, 0.5]
34
35
                  scales_per_octave: 2
              }
36
              }
37
              image_resizer {
38
              fixed_shape_resizer {
39
                  height: 640
40
                  width: 640
41
              }
42
              }
43
              box_predictor {
44
              weight_shared_convolutional_box_predictor {
45
                  depth: 256
46
                  class_prediction_bias_init: -4.6
47
                  conv_hyperparams {
48
                  activation: RELU_6,
49
                  regularizer {
50
51
                      12_regularizer {
52
                      weight: 0.0004
53
                      }
                  }
54
55
                  initializer {
56
                      random_normal_initializer {
                      stddev: 0.01
57
                      mean: 0.0
58
                      }
59
                  }
60
                  batch_norm {
61
                      scale: true,
62
                      decay: 0.997,
63
                      epsilon: 0.001,
64
                  }
65
                  }
66
                  num_layers_before_predictor: 4
67
                  kernel_size: 3
68
              }
69
              }
70
              feature_extractor {
71
```

```
type: 'ssd_resnet50_v1_fpn'
72
              fpn {
73
                   min_level: 3
74
                   max_level: 7
75
              }
76
              min_depth: 16
77
              depth_multiplier: 1.0
78
79
              conv_hyperparams {
80
                   activation: RELU_6,
81
                   regularizer {
82
                   12_regularizer {
83
                       weight: 0.0004
84
                   }
                   }
85
                   initializer {
86
                   truncated_normal_initializer {
87
                       stddev: 0.03
88
                       mean: 0.0
89
                   }
90
                   }
91
                   batch_norm {
92
                   scale: true,
93
                   decay: 0.997,
94
                   epsilon: 0.001,
95
                   }
96
              }
97
98
               override_base_feature_extractor_hyperparams: true
              }
99
100
              loss {
              classification_loss {
101
102
                   weighted_sigmoid_focal {
103
                   alpha: 0.25
104
                   gamma: 2.0
105
                   }
              }
106
              localization_loss {
107
                   weighted_smooth_l1 {
108
                   }
109
              }
110
              classification_weight: 1.0
111
              localization_weight: 1.0
112
              }
113
              normalize_loss_by_num_matches: true
114
              normalize_loc_loss_by_codesize: true
115
116
              post_processing {
117
              batch_non_max_suppression {
118
                   score_threshold: 1e-8
119
                   iou_threshold: 0.6
120
                   max_detections_per_class: 100
121
                   max_total_detections: 100
              }
122
123
              score_converter: SIGMOID
              }
124
          }
125
          }
126
```

C.3.2 Training Configuration

```
train_config: {
 1
 \mathbf{2}
       fine_tune_checkpoint: "PATH_TO_BE_CONFIGURED/model.ckpt"
3
       batch_size: 8
 4
       sync_replicas: true
       startup_delay_steps: 0
\mathbf{5}
       replicas_to_aggregate: 8
 6
       num_steps: 100000
 7
       data_augmentation_options {
8
         random_horizontal_flip {
9
         }
10
         random_vertical_flip {
11
         }
12
       }
13
       data_augmentation_options {
14
        random_crop_image {
15
           min_object_covered: 0.0
16
17
           min_aspect_ratio: 0.75
18
           max_aspect_ratio: 3.0
19
           min_area: 0.75
20
           max_area: 1.0
^{21}
           overlap_thresh: 0.0
^{22}
         }
       }
^{23}
^{24}
       optimizer {
        momentum_optimizer: {
25
26
           learning_rate: {
27
             cosine_decay_learning_rate {
28
               learning_rate_base: .04
                total_steps: 50000
29
                warmup_learning_rate: .013333
30
31
                warmup_steps: 2000
             }
32
           }
33
           momentum_optimizer_value: 0.9
34
         }
35
36
         use_moving_average: false
       }
37
       max_number_of_boxes: 100
38
39
       unpad_groundtruth_tensors: false
     }
40
```

C.3.3 Evaluation Configuration

```
1 eval_config: {
2 metrics_set: "pascal_voc_detection_metrics"
3 use_moving_averages: false
4 num_examples: 8000
5 }
```

C.4 Input Configuration

The two following snippets show the training and evaluation configuration. The multiple training sets can be done by appending to the input path.

```
1 train_input_reader: {
2 tf_record_input_reader {
3 input_path: "PATH_TO_BE_CONFIGURED/train.tfrecord"
4 }
5 label_map_path: "PATH_TO_BE_CONFIGURED/label_map.pbtxt"
6 }
```

```
eval_input_reader: {
1
\mathbf{2}
       tf_record_input_reader {
3
          input_path: "PATH_TO_BE_CONFIGURED/evaluation.tfrecord"
4
       }
\mathbf{5}
       label_map_path: "PATH_TO_BE_CONFIGURED/label_map.pbtxt"
6
       shuffle: false
\overline{7}
       num_readers: 1
8
     }
```

C.5 Label Map

The aim of this work is to detect worker-on-foot, hence the number of object class is one for all networks.

```
1 item {
2 id: 1
3 name: 'person'
4 }
```

D. Zoom Controller

This chapter elaborates the experimental results regarding zoom controller FSM by showing internal substates of the state machines. The first two figures presents the detailed result of zoom controller FSM using AprilTag detector as an input sensor which corresponds to the zoom control result in Sec. 6.7. On the other hand, the input sensor of the latter two figures comes from the top view worker detection which corresponds to the result of Sec. 7.2. The two experiments will be referred as TAG for AprilTag and as BOX for top view detection BBox. The (internal) sensor states (e.g., \overline{D} , \overline{N}) are depicted in Fig. D.1 and D.3, while the camera control states are illustrated in Fig. D.2 and D.4.

There are three main FSM states, $S = \{S_E, S_{TA}, S_{TD}\}$, which is mentioned in Chapter 6. However, S_E and S_{TA} break down as follows.

- State Explore $S_E = \{S_{E,in}, S_{E,out}\}$ The Explore state searchs for the target(s) by zooming in or out. As long as there is no target and the zoom level Z_t does not reach to the boundary i.e., $[0, Z_{ci,max}]$. $S_{E,in}$ basically means the camera zooms *in* until a target is detected or the camera is maximum zooomed out i.e., $Z_t = 0$, while the camera zooms *out* at state $S_{E,out}$ after Z_t reaches $Z_{ci,max}$.
- State TrackArea $S_{TA} = \{S_{TA}, S_{TA,adjust}\}$ $S_{TA}, S_{TA,adjust}$ The task of this state is to preserve overall target area. Once the target is found after exploring, the current state stays at S_{TA} to check the overall area of the target. The state will transit to $S_{TA,adjust}$ when the overall target area \overline{A} goes beyond or violates the outer region R_+ . $S_{TA,adjust}$ will try to adjust \overline{A} by either zooming in or out to bring \overline{A} back inside the inner region R_- .

defined boundary R_+ . (j) illustrates the raw total target number N, the moving average of target number on each frame N, and its standard (i) presents the in-range status flag of A. (f) presents the in-bound status flag of overall detected region R_t if the perimeter remains inside the in-range status of \overline{D} can be found in (<u>d</u>). (e) shows the flags identifying if \overline{D} is too large (red), in range (green), or too less (yellow). Likewise, average D, and its standard deviation band which is upper/lower D_{SD} . The desired BBox diagonal range is displayed in black dashed line. The is used to calculate the instant zoom level Z_t which indicates in Fig. D.2e. (c) is the raw average BBox diagonal of target D_t , the moving through the experiment. (b) is zoom command input which can be zooming in O_{zi} , zoooming out O_{zo} , or no zoom. The zoom command input Figure D.1: (Internal) sensor state for AprilTag detector, TAG which corresponds to the result in Sec. 6.7. (a) depicts the state transition deviation band which is upper/lower N_{SD} .





Figure D.2: Camera control state for AprilTag detector, CAM - TAG which corresponds to the result in Sec. 6.7. (a) indicates the zoom status flag whether Z_t reaches maximum zoom in $Z_{ci,max}$, maximum zoom out $Z_{co,max}$, or is resetting Z_{\Re} . At t=140, $Z_{co,max}$ flag is triggered. see (g) and is finally calcuclated as the zoom level Z_t depicted in (e). As there is a couple of zoom control command source possibilities e.g., manual remote control or FSMs, see Fig. 6.9, (f) is *SelectZoomCommandSrc* signal for the *ZoomCtrlSrcArbiter*. (c) is a manual zoom input flag for zoom control in FINROC. If the flag is false, the command is simply ignored. (i) is reset status flag. If the reset signal \Re of the FSM is Consequently, the camera starts to zoom out as the zoom level decreases, see (f). The zoom command in (b) is encoded into zoom in/out pulse, command. Despite giving zooming out command, the signal is suppressed as the arbiter does not choose this manual command. (h) is an enable activated, the flag will hold and any zoom command from other sources will be suppressed until the reset is done i.e., the camera zoom level reaches to $Z_{\alpha,max}$. In this case, there is no reset signal activated.



remains inside the defined boundary R_+ . (j) illustrates the raw total target number N, the moving average of target number on each frame \overline{N} (yellow). Likewise, (i) presents the in-range status flag of \overline{A} . (f) presents the in-bound status flag of overall detected region R_t if the perimeter command input is used to calculate the instant zoom level Z_t which indicates in Fig. D.4e. (c) is the raw average BBox diagonal of target D_t Figure D.3: (Internal) sensor state for top view worker detector, BOX which corresponds to the result of Sec. 7.2. (a) depicts the state and its standard deviation band which is upper/lower N_{SD} . dashed line. The in-range status of \overline{D} can be found in (d). the moving average D, and its standard deviation band which is upper/lower D_{SD} . The desired BBox diagonal range is displayed in black transition through the experiment. (b) is zoom command input which can be zooming in O_{zi} , zoooming out O_{zo} , or no zoom. The zoom (e) shows the flags identifying if \overline{D} is too large (red), in range (green), or too less



Figure D.4: Camera control state for top view worker detector, CAM - BOX which corresponds to the result of Sec. 7.2.(a) indicates the zoom status flag whether Z_t reaches maximum zoom in $Z_{ci,max}$, maximum zoom out $Z_{co,max}$, or is resetting Z_{\Re} . The zoom command in (b) is encoded into zoom in/out pulse, see (g) and is finally calcuclated as the zoom level Z_t depicted in (e). As there is a couple of zoom control command source possibilities e.g., manual remote control or FSMs, see Fig. 6.9, (f) is SelectZoomCommandSrc signal for the ZoomCtrlSrcArbiter. command. (h) is an enable flag for zoom control in FINROC. If the flag is false, the command is simply ignored. (i) is reset status flag. If the c) is a manual zoom input command. Despite giving zooming out command, the signal is suppressed as the arbiter does not choose this manual reset signal \Re of the FSM is activated, the flag will hold and any zoom command from other sources will be suppressed until the reset is done i.e., the camera zoom level reaches to $Z_{ci,max}$. In this case, there is no reset signal activated.

E. Magnitude Spectrum

Frequency domain study of the synthetic data captures the unrealistic features of these images by analyzing the local rate of changes of pixels. It can highlight edges and noise as high-frequency contents which appears on-surround region of magnitude spectrum. This appendix presents the additional Magnitude Spectrum (MS) result for each dataset.



Figure E.1: Magnitude Spectrum (MS) of an individual frame of each image sequence. The first column displays original images. The second column presents a magnitude spectrum of the first column. The third column shows preprocessed images using the image filtering method. The fourth column is a magnitude spectrum of the preprocessed images which locate in the immediate previous column.





Figure E.1: Magnitude Spectrum (MS) of an individual frame of each image sequence. The first column displays original images. The second column presents a magnitude spectrum of the first column. The third column shows preprocessed images using the image filtering method. The fourth column is a magnitude spectrum of the preprocessed images which locate in the immediate previous column.



(r) U09-S-C4

Figure E.1: Magnitude Spectrum (MS) of an individual frame of each image sequence. The first column displays original images. The second column presents a magnitude spectrum of the first column. The third column shows preprocessed images using the image filtering method. The fourth column is a magnitude spectrum of the preprocessed images which locate in the immediate previous column.



(t) U11-S-C4

Figure E.1: Magnitude Spectrum (MS) of an individual frame of each image sequence. The first column displays original images. The second column presents a magnitude spectrum of the first column. The third column shows preprocessed images using the image filtering method. The fourth column is a magnitude spectrum of the preprocessed images which locate in the immediate previous column.

Bibliography

- T. D. Akinosho, L. O. Oyedele, M. Bilal, A. O. Ajayi, M. D. Delgado, O. O. Akinade, and A. A. Ahmed, *Deep learning in the construction industry: A review of present* status and future innovations. Elsevier, 2020.
- [2] European Commission, "Construction | Internal Market, Industry, Entrepreneurship and SMEs." https://bit.ly/3TzXrUB, 2020.
- [3] K. S. Saidi, T. Bock, and C. Georgoulas, "Robotics in construction," in Springer handbook of robotics, pp. 1493–1520, Springer, 2016.
- [4] European Commission, "Accidents at work statistics." https://bit.ly/3sbrq9Q, June 2018.
- [5] A. Bouman, M. F. Ginting, N. Alatur, M. Palieri, D. D. Fan, T. Touma, T. Pailevanian, S.-K. Kim, K. Otsu, J. Burdick, et al., "Autonomous spot: Long-range autonomous exploration of extreme environments with legged locomotion," in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2518–2525, IEEE, 2020.
- [6] Robotics Research Lab, TU Kaiserslautern, "Bomag BW154 and BW174." https: //agrosy.informatik.uni-kl.de/roboter/weitere-roboter/tandem-roller, 2021.
- [7] K. Kaneko, H. Kaminaga, T. Sakaguchi, S. Kajita, M. Morisawa, I. Kumagai, and F. Kanehiro, "Humanoid robot HRP-5P: An electrically actuated humanoid robot with high-power and wide-range joints," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1431–1438, 2019, IEEE.
- [8] T. Bock and T. Linner, Construction Robots: Volume 3: Elementary Technologies and Single-task Construction Robots. Cambridge University Press, 2016.
- [9] T. Bock, "Construction robotics," Autonomous Robots, vol. 22, no. 3, pp. 201–209, 2007, Springer.
- [10] B. C. Paulson Jr, "Automation and robotics for construction," Journal of construction engineering and management, vol. 111, no. 3, pp. 190–207, 1985, American Society of Civil Engineers.
- [11] H. Son, C. Kim, H. Kim, S. H. Han, and M. K. Kim, "Trend analysis of research and development on automation and robotics technology in the construction industry," *KSCE Journal of Civil Engineering*, vol. 14, no. 2, pp. 131–139, 2010, Springer.

- [12] J. E. Beavers, J. Moore, R. Rinehart, and W. Schriver, "Crane-related fatalities in the construction industry," *Journal of Construction Engineering and Management*, vol. 132, no. 9, pp. 901–910, 2006, American Society of Civil Engineers.
- [13] M. McCann, "Understanding crane accident failures: A report on causes of deaths in crane-related accidents," tech. rep., The Center for Construction Research and Training (CPWR), May 2010.
- [14] J. Hinze, X. Huang, and L. Terry, "The nature of struck-by accidents," Journal of construction engineering and management, vol. 131, no. 2, pp. 262–268, 2005, American Society of Civil Engineers.
- [15] WSHCouncil, "Crane safety analysis and recommendation report," tech. rep., 2009.
- [16] R. L. Neitzel, N. S. Seixas, and K. K. Ren, "A review of crane safety in the construction industry," *Applied occupational and environmental hygiene*, vol. 16, no. 12, pp. 1106–1117, 2001, Taylor & Francis Group.
- [17] R. H. Flin, P. O'Connor, and M. Crichton, Safety at the sharp end: a guide to non-technical skills. Ashgate Publishing, Ltd., 2008.
- [18] Occupational Safety and Health Administration (OSHA), "Robotics standards." https://www.osha.gov/SLTC/robotics/standards.html, 2020.
- [19] C. Kyrkou, G. Plastiras, T. Theocharides, S. I. Venieris, and C.-S. Bouganis, "Dronet: Efficient convolutional neural network detector for real-time uav applications," in 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 967–972, IEEE, 2018.
- [20] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," tech. rep., Carnegie-Mellon University Pittsburgh PA Artificial Intelligence and Psychology, 1989.
- [21] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Handwritten digit recognition with a back-propagation network," *Advances in neural* information processing systems, vol. 2, 1989.
- [22] V. Govindan, "Observations On Tesla's AI Day." https://cleantechnica.com/ 2021/08/30/observations-on-teslas-ai-day/, 2021.
- [23] I. Demir, K. Koperski, D. Lindenbaum, G. Pang, J. Huang, S. Basu, F. Hughes, D. Tuia, and R. Raskar, "Deepglobe 2018: A challenge to parse the earth through satellite images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 172–181, 2018.
- [24] D. Yang, L. Li, K. Redmill, and Ü. Özgüner, "Top-view trajectories: A pedestrian dataset of vehicle-crowd interaction from controlled experiments and crowded campus," in 2019 IEEE Intelligent Vehicles Symposium (IV), pp. 899–904, 2019.

- [25] P. Zhu, D. Du, L. Wen, X. Bian, H. Ling, Q. Hu, T. Peng, J. Zheng, X. Wang, Y. Zhang, et al., "Visdrone-vid2019: The vision meets drone object detection in video challenge results," in *Proceedings of the IEEE/CVF International Conference* on Computer Vision Workshops, 2019.
- [26] NU, "Mobile crane checklist." https://www.northwestern.edu/risk/documents/ ehs-documents/facility-docs/mobile-crane-checklist.pdf, 2020.
- [27] WSHC, "Sample checklist for mobile crane." https://www.wshc.sg/files/wshc/ upload/cms/file/3_Mobile_Crawler_Crane_checklist.pdf, 2020.
- [28] B. Browatzki, J. Fischer, B. Graf, H. H. Bülthoff, and C. Wallraven, "Going into depth: Evaluating 2d and 3d cues for object classification on a new, large-scale object dataset," in 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), pp. 1189–1195, IEEE, 2011.
- [29] S. J. Ray and J. Teizer, "Real-time construction worker posture analysis for ergonomics training," Advanced Engineering Informatics, vol. 26, no. 2, pp. 439–455, 2012, Elsevier.
- [30] S. Han, S. Lee, and F. Peña-Mora, "Vision-based detection of unsafe actions of a construction worker: Case study of ladder climbing," *Journal of Computing in Civil Engineering*, vol. 27, no. 6, pp. 635–644, 2013, American Society of Civil Engineers.
- [31] S. Chi and C. H. Caldas, "Image-based safety assessment: automated spatial safety risk identification of earthmoving and surface mining activities," *Journal of Construction Engineering and Management*, vol. 138, no. 3, pp. 341–351, 2011, American Society of Civil Engineers.
- [32] J. Seo, S. Han, S. Lee, and H. Kim, "Computer vision techniques for construction safety and health monitoring," *Advanced Engineering Informatics*, vol. 29, no. 2, pp. 239–251, 2015, Elsevier.
- [33] X. Yan, H. Zhang, and H. Li, "Estimating worker-centric 3d spatial crowdedness for construction safety management using a single 2d camera," *Journal of Computing in Civil Engineering*, vol. 33, no. 5, p. 04019030, 2019, American Society of Civil Engineers.
- [34] 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, 2015, American Society of Civil Engineers.
- [35] A. H. M. Rubaiyat, T. T. Toma, M. Kalantari-Khandani, S. A. Rahman, L. Chen, Y. Ye, and C. S. Pan, "Automatic detection of helmet uses for construction safety," in 2016 IEEE/WIC/ACM International Conference on Web Intelligence Workshops (WIW), pp. 135–142, Oct 2016.
- [36] R. Mosberger, H. Andreasson, and A. J. Lilienthal, "Multi-human tracking using highvisibility clothing for industrial safety," in 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 638–644, IEEE, 2013.

- [37] J. Yang, O. Arif, P. A. Vela, J. Teizer, and Z. Shi, "Tracking multiple workers on construction sites using video cameras," *Advanced Engineering Informatics*, vol. 24, no. 4, pp. 428–434, 2010, Elsevier.
- [38] R. Mosberger and H. Andreasson, "An inexpensive monocular vision system for tracking humans in industrial environments," in 2013 IEEE International Conference on Robotics and Automation, pp. 5850–5857, May 2013.
- [39] K. M. Rashid and A. H. Behzadan, "Risk behavior-based trajectory prediction for construction site safety monitoring," *Journal of construction engineering and management*, vol. 144, no. 2, p. 04017106, 2018, American Society of Civil Engineers.
- [40] D. Kim, M. Liu, S. Lee, and V. R. Kamat, "Trajectory prediction of mobile construction resources toward pro-active struck-by hazard detection," in *ISARC. Proceedings* of the International Symposium on Automation and Robotics in Construction, vol. 36, pp. 982–988, IAARC Publications, 2019.
- [41] T. Sutjaritvorakul, S. Piao, and K. Berns, "Single camera based multiple pedestrian tracking for resource-limited hardware systems," in *Proceedings DGR Days 2016*, *June, 29-30, Leipzig*, p. 27, 2016.
- [42] J. T. Albers and C. F. Estill, "Simple solutions ergonomics for construction workers," Tech. Rep. 2007-122, The National Institute for Occupational Safety and Health (NIOSH), Aug. 2007.
- [43] S. Kamat, N. M. Zula, N. Rayme, S. Shamsuddin, and K. Husain, "The ergonomics body posture on repetitive and heavy lifting activities of workers in aerospace manufacturing warehouse," in *IOP Conference Series: Materials Science and Engineering*, vol. 210, p. 012079, IOP Publishing, 2017.
- [44] J. Yang, Z. Shi, and Z. Wu, "Automatic recognition of construction worker activities using dense trajectories," in *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction*, vol. 32, p. 1, IAARC Publications, 2015.
- [45] J. Kim, S. Chi, and J. Seo, "Interaction analysis for vision-based activity identification of earthmoving excavators and dump trucks," *Automation in Construction*, vol. 87, pp. 297–308, 2018, Elsevier.
- [46] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," Acm computing surveys (CSUR), vol. 38, no. 4, pp. 13–es, 2006, Acm New York, NY, USA.
- [47] S. Schneider and P. Susi, "Ergonomics and construction: a review of potential hazards in new construction," *American Industrial Hygiene Association Journal*, vol. 55, no. 7, pp. 635–649, 1994, Taylor & Francis.
- [48] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese, "Learning social etiquette: Human trajectory understanding in crowded scenes," in *European conference on computer vision*, pp. 549–565, Springer, 2016. Stanford Drone Dataset.

- [49] M. Teutsch and W. Kruger, "Robust and fast detection of moving vehicles in aerial videos using sliding windows," in *Proceedings of the IEEE conference on computer* vision and pattern recognition workshops, pp. 26–34, 2015.
- [50] C. H. Lampert, M. B. Blaschko, and T. Hofmann, "Beyond sliding windows: Object localization by efficient subwindow search," in 2008 IEEE conference on computer vision and pattern recognition, pp. 1–8, IEEE, 2008.
- [51] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010, IEEE.
- [52] R. A. Lingam and K. S. Kumar, "Statistically tuned gaussian background subtraction technique for uav videos," *Sadhana*, vol. 39, no. 4, pp. 785–808, 2014, Springer.
- [53] G. Morris and P. Angelov, "Real-time novelty detection in video using background subtraction techniques: state of the art a practical review," in 2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 537–543, IEEE, 2014.
- [54] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International journal of computer vision*, vol. 59, no. 2, pp. 167–181, 2004, Springer.
- [55] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013, Springer.
- [56] T. Tewari, K. V. Sakhare, and V. Vyas, "Vehicle detection in aerial images using selective search with a simple deep learning based combination classifier," in *Pro*ceedings of the Third International Conference on Microelectronics, Computing and Communication Systems, pp. 221–233, Springer, 2019.
- [57] D. E. Maggio and D. A. Cavallaro, Video Tracking: Theory and Practice. Wiley Publishing, 1st ed., 2011.
- [58] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01, vol. 00 of CVPR '05, (Washington, DC, USA), pp. 886–893, IEEE Computer Society, 2005.
- [59] A. M. Garcia, M. A. Rufino, L. C. Sangalang, J. A. Teodoro, and J. Ilao, "Application of histogram of oriented gradient in person detection from aerial images," 2014.
- [60] S. Razakarivony and F. Jurie, "Vehicle detection in aerial imagery: A small target detection benchmark," *Journal of Visual Communication and Image Representation*, vol. 34, pp. 187–203, 2016, Elsevier.
- [61] S. Liao, X. Zhu, Z. Lei, L. Zhang, and S. Z. Li, "Learning multi-scale block local binary patterns for face recognition," in *International Conference on Biometrics*, pp. 828–837, Springer, 2007.

- [62] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," Int. J. Comput. Vision, vol. 60, pp. 91–110, Nov. 2004, Kluwer Academic Publishers.
- [63] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition*, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, vol. 1, pp. I–I, IEEE, 2001.
- [64] T. Moranduzzo, M. L. Mekhalfi, and F. Melgani, "Lbp-based multiclass classification method for uav imagery," in 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 2362–2365, IEEE, 2015.
- [65] F. Jabar, S. Farokhi, and U. Sheikh, "Object tracking using sift and klt tracker for uav-based applications," in 2015 ieee international symposium on robotics and intelligent sensors (iris), pp. 65–68, IEEE, 2015.
- [66] C. Cortes and V. Vapnik, "Support-Vector Networks," Mach. Learn., vol. 20, pp. 273–297, Sept. 1995, Kluwer Academic Publishers.
- [67] M. Park, E. Palinginis, and I. Brilakis, "Detection of construction workers in video frames for automatic initialization of vision trackers," in *Construction Research Congress 2012: Construction Challenges in a Flat World*, pp. 940–949, 2012.
- [68] J. Gleason, A. V. Nefian, X. Bouyssounousse, T. Fong, and G. Bebis, "Vehicle detection from aerial imagery," in 2011 IEEE International Conference on Robotics and Automation, pp. 2065–2070, IEEE, 2011.
- [69] P. Thanh Noi and M. Kappas, "Comparison of random forest, k-nearest neighbor, and support vector machine classifiers for land cover classification using sentinel-2 imagery," *Sensors*, vol. 18, no. 1, p. 18, 2018, Multidisciplinary Digital Publishing Institute.
- [70] Y. Fang, J. Chen, Y. K. Cho, K. Kim, S. Zhang, and E. Perez, "Vision-based load sway monitoring to improve crane safety in blind lifts," *Journal of Structural Integrity and Maintenance*, vol. 3, no. 4, pp. 233–242, 2018, Taylor & Francis.
- [71] M. Neuhausen, J. Teizer, and M. König, "Construction worker detection and tracking in bird's-eye view camera images," in *Proceedings of the 35th ISARC, Berlin, Germany*, 2018.
- [72] L. Bourdev and J. Brandt, "Robust object detection via soft cascade," in 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 2, pp. 236–243, IEEE, 2005.
- [73] BIBA, "safeguARd, Augmented Reality-based assistance system for commercial vehicles to raise the safety level," Nov. 2015.
- [74] S. Takahashi and S. Kaneko, "Motion tracking of crane hook based on optical flow and orientation code matching," in 2008 10th IEEE International Workshop on Advanced Motion Control, pp. 149–152, IEEE, 2008.

- [75] H. Takauji, S. Kaneko, and T. Tanaka, "Robust tagging in strange circumstance," *Electrical Engineering in Japan*, vol. 156, no. 4, pp. 22–32, 2006, Wiley Online Library.
- [76] M. Andriluka, P. Schnitzspan, J. Meyer, S. Kohlbrecher, K. Petersen, O. Von Stryk, S. Roth, and B. Schiele, "Vision based victim detection from unmanned aerial vehicles," in 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1740–1747, IEEE, 2010.
- [77] M. Andriluka, S. Roth, and B. Schiele, "Pictorial structures revisited: People detection and articulated pose estimation," in 2009 IEEE conference on computer vision and pattern recognition, pp. 1014–1021, IEEE, 2009.
- [78] L. Bourdev and J. Malik, "Poselets: Body part detectors trained using 3d human pose annotations," in 2009 IEEE 12th International Conference on Computer Vision, pp. 1365–1372, IEEE, 2009.
- [79] BLS, "Accidents involving head injuries," techreport 605, Bureau of Labor Statistics (BLS), U.S. Department of Labor, Washington, D.C., Government Printing Office, July 1980. p. 1.
- [80] Z. Yang, Y. Yuan, M. Zhang, X. Zhao, Y. Zhang, and B. Tian, "Safety distance identification for crane drivers based on mask r-cnn," *Sensors*, vol. 19, no. 12, p. 2789, 2019, Multidisciplinary Digital Publishing Institute.
- [81] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.
- [82] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.
- [83] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- [84] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy, "Speed/accuracy trade-offs for modern convolutional object detectors," *CoRR*, vol. abs/1611.10012, 2016.
- [85] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A survey of deep learning-based object detection," *IEEE Access*, vol. 7, pp. 128837–128868, 2019, IEEE.
- [86] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE* conference on computer vision and pattern recognition, pp. 580–587, 2014.
- [87] R. Girshick, "Fast R-CNN," in 2015 IEEE International Conference on Computer Vision (ICCV), pp. 1440–1448, Dec 2015.

- [88] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems 28* (C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, eds.), pp. 91–99, Curran Associates, Inc., 2015.
- [89] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in Proceedings of the IEEE international conference on computer vision, pp. 2961–2969, 2017.
- [90] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2015.
- [91] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7263–7271, 2017.
- [92] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," CoRR, vol. abs/1804.02767, 2018.
- [93] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," arXiv preprint arXiv:2004.10934, 2020.
- [94] G. Jocher, K. Nishimura, T. Mineeva, and R. Vilariño, "YOLOv5." https://github. com/ultralytics/yolov5, 2020.
- [95] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie, et al., "YOLOv6: A single-stage object detection framework for industrial applications," arXiv preprint arXiv:2209.02976, 2022.
- [96] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*, pp. 21–37, Springer, 2016.
- [97] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.
- [98] I. Ahmed, M. Ahmad, A. Ahmad, and G. Jeon, "Top view multiple people tracking by detection using deep sort and yolov3 with transfer learning: within 5g infrastructure," *International Journal of Machine Learning and Cybernetics*, pp. 1–15, 2020, Springer.
- [99] 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), pp. 3645–3649, IEEE, 2017.
- [100] D. Kim, M. Liu, S. Lee, and V. R. Kamat, "Remote proximity monitoring between mobile construction resources using camera-mounted UAVs," Automation in Construction, vol. 99, pp. 168–182, 2019, Elsevier.
- [101] H. Zhu, Y. Qi, H. Shi, N. Li, and H. Zhou, "Human detection under uav: an improved faster r-cnn approach," in 2018 5th International Conference on Systems and Informatics (ICSAI), pp. 367–372, IEEE, 2018.

- [102] M. Barekatain, M. Martí, H.-F. Shih, S. Murray, K. Nakayama, Y. Matsuo, and H. Prendinger, "Okutama-action: An aerial view video dataset for concurrent human action detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 28–35, 2017.
- [103] J. Liu, R. Jia, W. Li, F. Ma, H. M. Abdullah, H. Ma, and M. A. Mohamed, "High precision detection algorithm based on improved retinanet for defect recognition of transmission lines," *Energy Reports*, vol. 6, pp. 2430–2440, 2020, Elsevier.
- [104] S. Kapania, D. Saini, S. Goyal, N. Thakur, R. Jain, and P. Nagrath, "Multi object tracking with uavs using deep sort and yolov3 retinanet detection framework," in *Proceedings of the 1st ACM Workshop on Autonomous and Intelligent Mobile Systems*, pp. 1–6, 2020.
- [105] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer, "Densenet: Implementing efficient convnet descriptor pyramids," arXiv preprint arXiv:1404.1869, 2014.
- [106] G. Golcarenarenji, I. Martinez-Alpiste, Q. Wang, and J. M. Alcaraz-Calero, "Machinelearning-based top-view safety monitoring of ground workforce on complex industrial sites," *Neural Computing and Applications*, pp. 1–14, 2021, Springer.
- [107] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8759–8768, 2018.
- [108] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015, IEEE.
- [109] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang, "The apolloscape dataset for autonomous driving," *CoRR*, vol. abs/1803.06184, 2018.
- [110] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov, "Scalability in perception for autonomous driving: An open dataset benchmark," 2019.
- [111] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), 2016.
- [112] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.
- [113] C. N. Laflamme, F. Pomerleau, and P. Giguère, "Driving datasets literature review," arXiv preprint arXiv:1910.11968, 2019.

- [114] G. Neuhold, T. Ollmann, S. Rota Bulo, and P. Kontschieder, "The mapillary vistas dataset for semantic understanding of street scenes," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4990–4999, 2017.
- [115] D. Feng, A. Harakeh, S. L. Waslander, and K. Dietmayer, "A review and comparative study on probabilistic object detection in autonomous driving," *IEEE Transactions* on Intelligent Transportation Systems, 2021, IEEE.
- [116] N. Bhattarai, T. Nakamura, and C. Mozumder, "Real time human detection and localization using consumer grade camera and commercial uav," 2018, Preprints.
- [117] D. Cazzato, C. Cimarelli, J. L. Sanchez-Lopez, H. Voos, and M. Leo, "A survey of computer vision methods for 2d object detection from unmanned aerial vehicles," *Journal of Imaging*, vol. 6, no. 8, p. 78, 2020, Multidisciplinary Digital Publishing Institute.
- [118] T. Sutjaritvorakul, A. Vierling, and K. Berns, "Data-driven worker detection from load-view crane camera," in *Proceedings of the 37th International Symposium on Automation and Robotics in Construction (ISARC)* (F. H. T. K. "Osumi, Hisashi", ed.), (Kitakyshu, Japan), pp. 864–871, International Association for Automation and Robotics in Construction (IAARC), October 2020.
- [119] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The clear mot metrics," J. Image Video Process., vol. 2008, pp. 1:1–1:10, Jan. 2008, Hindawi Publishing Corp.
- [120] P. Zhu, L. Wen, D. Du, X. Bian, Q. Hu, and H. Ling, "Vision meets drones: Past, present and future," arXiv preprint arXiv:2001.06303, 2020.
- [121] F. Rosique, P. J. Navarro, C. Fernández, and A. Padilla, "A systematic review of perception system and simulators for autonomous vehicles research," *Sensors*, vol. 19, no. 3, p. 648, 2019, Multidisciplinary Digital Publishing Institute.
- [122] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and service robotics*, pp. 621– 635, Springer, 2018.
- [123] Udacity, "UdaciDrone." https://udacity.github.io/udacidrone/, 2018.
- [124] T. Sutjaritvorakul, A. Vierling, J. Pawlak, and K. Berns, "Simulation platform for crane visibility safety assistance," in Advances in Service and Industrial Robotics (S. Zeghloul, M. A. Laribi, and J. S. Sandoval Arevalo, eds.), vol. 84 of Mechanisms and Machine Science, (Cham, France), pp. 22–29, Springer International Publishing, 2020.
- [125] N. Mayer, E. Ilg, P. Fischer, C. Hazirbas, D. Cremers, A. Dosovitskiy, and T. Brox, "What makes good synthetic training data for learning disparity and optical flow estimation?," *International Journal of Computer Vision*, vol. 126, no. 9, pp. 942–960, 2018, Springer.
- [126] T. Sutjaritvorakul, A. Vierling, and K. Berns, "Simulated environment for developing crane safety assistance technology," in *Commercial Vehicle Technology 2020*. *Proceedings of the 6th Commercial Vehicle Technology Symposium CVT 2020* (K. Berns, K. Dressler, P. Fleischmann, D. Görges, R. Kalmar, B. Sauer, N. Stephan, R. Teutsch, and M. Thul, eds.), (Kaiserslautern, Germany), Commercial Vehicle Alliance Kaiserslautern (CVA), Springer, March 10–12 2020. ISBN: 978-3-658-29717-6.
- [127] M. Reichardt, T. Föhst, and K. Berns, "Introducing FINROC: A Convenient Real-Time Framework for Robotics Based on a Systematic Design Approach," no. III, pp. 1–8, 2012.
- [128] T. Sutjaritvorakul, A. Nejadfard, A. Vierling, and K. Berns, "Adaptive zoom control approach of load-view crane camera for worker detection," in *Proceedings of the 38th International Symposium on Automation and Robotics in Construction (ISARC)*, (Dubai, UAE), pp. 553–560, International Association for Automation and Robotics in Construction (IAARC), November 2021.
- [129] B. Gleissner, "LiDAR and the Land Surveyor," Tech. Rep. 01, Point of Beginning (POB), Oct. 2019.
- [130] Y. Li, S. Wang, and B. Li, "Improved visual hook capturing and tracking for precision hoisting of tower crane," *Advances in Mechanical Engineering*, vol. 5, p. 426810, 2013, SAGE Publications Sage UK: London, England.
- [131] Orlaco Inc, "Product catalog, cranes vision solutions." https://cranewarning. com/wp-content/uploads/2015/07/ORLAC0%20North%20American%20Crane% 20Booklet.pdf, Sept. 2019.
- [132] N. O'Mahony, S. Campbell, A. Carvalho, S. Harapanahalli, G. V. Hernandez, L. Krpalkova, D. Riordan, and J. Walsh, "Deep learning vs. traditional computer vision," in *Science and Information Conference*, pp. 128–144, Springer, 2019.
- [133] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust physical-world attacks on deep learning visual classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1625–1634, 2018.
- [134] V. V. Dixit, S. Chand, and D. J. Nair, "Autonomous vehicles: disengagements, accidents and reaction times," *PLoS one*, vol. 11, no. 12, p. e0168054, 2016, Public Library of Science San Francisco, CA USA.
- [135] J. Hinze, "Construction safety," Safety science, vol. 46, no. 4, pp. 565–565, 2008, Elsevier Publishing.
- [136] S. Lichiardopol, "A survey on teleoperation," Technische Universität Eindhoven, DCT report, vol. 20, pp. 40–60, 2007, Citeseer.
- [137] J. W. Hinze and J. Teizer, "Visibility-related fatalities related to construction equipment," Safety science, vol. 49, no. 5, pp. 709–718, 2011, Elsevier.

- [138] J. G. Everett and A. H. Slocum, "Cranium: device for improving crane productivity and safety," *Journal of construction engineering and management*, vol. 119, no. 1, pp. 23–39, 1993, American Society of Civil Engineers.
- [139] E. Winsberg, "Computer simulations in science." https://plato.stanford.edu/ entries/simulations-science/?utm_source=feedly, 2013.
- [140] J. F. Robeson, *Logistics handbook*. Simon and Schuster, 1994.
- [141] Thermopylae Sciences + Technology, "Humans process visual data better." https: //bit.ly/3s7B0PF, Sept. 2014.
- [142] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566), vol. 3, pp. 2149–2154, IEEE, 2004.
- [143] E. Rohmer, S. P. N. Singh, and M. Freese, "V-REP: A versatile and scalable robot simulation framework," in 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1321–1326, Nov 2013.
- [144] E. Rohmer, S. P. N. Singh, and M. Freese, "Coppeliasim (formerly v-rep): a versatile and scalable robot simulation framework," in *Proc. of The International Conference* on Intelligent Robots and Systems (IROS), 2013. www.coppeliarobotics.com.
- [145] A. Staranowicz and G. L. Mariottini, "A survey and comparison of commercial and open-source robotic simulator software," in *Proceedings of the 4th International Conference on PErvasive Technologies Related to Assistive Environments*, pp. 1–8, 2011.
- [146] L. Nogueira, "Comparative analysis between gazebo and v-rep robotic simulators," Seminario Interno de Cognicao Artificial-SICA, vol. 2014, no. 5, 2014.
- [147] Unity Technologies, "Unity." https://unity.com/, 2005.
- [148] Epic Games, "Unreal engine." https://www.unrealengine.com, 1998.
- [149] R. Louali, A. Belloula, M. S. Djouadi, and S. Bouaziz, "Real-time characterization of microsoft flight simulator 2004 for integration into hardware in the loop architecture," in 2011 19th Mediterranean Conference on Control & Automation (MED), pp. 1241– 1246, IEEE, 2011.
- [150] M. Lewis and J. Jacobson, "Game engines in scientific research," Communications of the ACM, vol. 45, no. 1, p. 27, 2002.
- [151] J. Gregory, *Game Engine Architecture*. CRC Press, 3 ed., 2018.
- [152] Epic Games, "Unreal engine marketplace." https://www.unrealengine.com/ marketplace/store, 2017.

- [153] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "Virtual worlds as proxy for multi-object tracking analysis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4340–4349, 2016.
- [154] Y. Cabon, N. Murray, and M. Humenberger, "Virtual KITTI 2," 2020.
- [155] A. Ruano, "DeepGTAV: A plugin for GTAV that transforms it into a vision-based self-driving car research environment," 2017.
- [156] S. Carpin, M. Lewis, J. Wang, S. Balakirsky, and C. Scrapper, "USARSim: a robot simulator for research and education," in *Robotics and Automation*, 2007 IEEE International Conference on, pp. 1400–1405, IEEE, 2007.
- [157] P. Wolf, T. Groll, S. Hemer, and K. Berns, "Evolution of robotic simulators: Using UE 4 to enable real-world quality testing of complex autonomous robots in unstructured environments," 2020.
- [158] M. Müller, V. Casser, J. Lahoud, N. Smith, and B. Ghanem, "Ue4sim: A photorealistic simulator for computer vision applications," 2017, arXiv.
- [159] M. Müller, V. Casser, J. Lahoud, N. Smith, and B. Ghanem, "Sim4cv: A photorealistic simulator for computer vision applications," *International Journal of Computer Vision*, vol. 126, no. 9, pp. 902–919, 2018, Springer.
- [160] W. Qiu and A. L. Yuille, "UnrealCV: Connecting Computer Vision to Unreal Engine," CoRR, vol. abs/1609.01326, 2016.
- [161] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Conference on robot learning*, pp. 1–16, PMLR, 2017.
- [162] Baidu, "Apollo simulation." http://apollo.auto/platform/simulation.html, 2017.
- [163] A. Brown et al., "Udacity's self-driving car simulator." https://github.com/ udacity/self-driving-car-sim, 2016.
- [164] G. Rong, B. H. Shin, H. Tabatabaee, Q. Lu, S. Lemke, M. Možeiko, E. Boise, G. Uhm, M. Gerow, S. Mehta, et al., "LGSVL Simulator: A High Fidelity Simulator for Autonomous Driving," arXiv preprint arXiv:2005.03778, 2020.
- [165] B. Wymann, E. Espié, C. Guionneau, C. Dimitrakakis, R. Coulom, and A. Sumner, "Torcs, the open racing car simulator," *Software available at http://torcs. sourceforge.* net, vol. 4, no. 6, p. 2, 2000.
- [166] M. M. Manhães, S. A. Scherer, M. Voss, L. R. Douat, and T. Rauschenbach, "UUV simulator: A gazebo-based package for underwater intervention and multi-robot simulation," in OCEANS 2016 MTS/IEEE Monterey, IEEE, sep 2016.
- [167] M. Prats, J. Perez, J. J. Fernandez, and P. J. Sanz, "An open source tool for simulation and supervision of underwater intervention missions," in 2012 IEEE/RSJ international conference on Intelligent Robots and Systems, pp. 2577–2582, IEEE, 2012.

- [168] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on* open source software, vol. 3, p. 5, Kobe, Japan, 2009.
- [169] M. Krafft, "The VR construction worker-safety training program of the future." https://unity.com/event/unite/2019/copenhagen, Sept. 2019.
- [170] D. Zhao and J. Lucas, "Virtual reality simulation for construction safety promotion," *International journal of injury control and safety promotion*, vol. 22, no. 1, pp. 57–67, 2015, Taylor & Francis.
- [171] Liebherr, "Liebherr simulations (LiSIM)." https://www.liebherr.com/en/int/ products/maritime-cranes/maritime-technology/crane-simulators-lisim/ lisim-special-page.html, 2013.
- [172] H. AlBahnassi and A. Hammad, "Near real-time motion planning and simulation of cranes in construction: Framework and system architecture," *Journal of Computing in Civil Engineering*, vol. 26, no. 1, pp. 54–63, 2011, American Society of Civil Engineers.
- [173] Y. Fang, Y. K. Cho, and J. Chen, "A framework for real-time pro-active safety assistance for mobile crane lifting operations," *Automation in Construction*, vol. 72, pp. 367–379, 2016, Elsevier.
- [174] K. Pimentel, "Interactive vr training improves construction site safety and roi." https://www.unrealengine.com/en-US/spotlights/ interactive-vr-training-improves-construction-site-safety-and-roi, 2019.
- [175] Liebherr, "Liebherr simulations solutions for construction machines." https://www.liebherr.com/shared/media/ construction-machinery/deep-foundation/digital-solutions/lisim/ liebherr-lisim-training-simulators-brochure-english.pdf, 2020.
- [176] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *Proceedings of the IEEE conference on computer vision and pattern* recognition, pp. 3234–3243, 2016.
- [177] Ministerium der Justiz, "Landeswaldgesetz (LWaldG)." https://bit.ly/3DdhOBM, 2000.
- [178] LBA, "Luftverkehrsgesetz." http://www.gesetze-im-internet.de/luftvg/index. html, 2007.
- [179] R. Pagare and A. Shinde, "A study on image annotation techniques," International Journal of Computer Applications, vol. 37, no. 6, pp. 42–45, 2012, Citeseer.
- [180] A. Barriuso and A. Torralba, "Notes on image annotation," *arXiv preprint* arXiv:1210.3448, 2012.

- [181] A. Sorokin and D. Forsyth, "Utility data annotation with amazon mechanical turk," in Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on, pp. 1–8, IEEE, 2008.
- [182] Amazon, "Amazon sagemaker ground truth." https://aws.amazon.com/sagemaker, 2017.
- [183] B. Sekachev *et al.*, "Computer vision annotation tool: a universal approach to data annotation," *Intel [Internet]*, vol. 1, 2019. CVAT.
- [184] Labelbox, "Labelbox." https://labelbox.com/, 2019.
- [185] SuperAnnotate AI, Inc., Sunnyvale, California, SuperAnnotate Desktop version 1.0 (2020), 2020.
- [186] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "Pascal visual object classes challenge 2007 (voc2007) annotation guidelines." http: //host.robots.ox.ac.uk:8080/pascal/VOC/voc2007/guidelines.html, 2007.
- [187] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "Voc 2008 annotation guidelines." http://host.robots.ox.ac.uk/pascal/VOC/ voc2008/guidelines.html, 2008.
- [188] S. Zhang, R. Benenson, M. Omran, J. Hosang, and B. Schiele, "How far are we from solving pedestrian detection?," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1259–1267, 2016.
- [189] A. Ammar, A. Koubaa, M. Ahmed, and A. Saad, "Aerial images processing for car detection using convolutional neural networks: Comparison between faster r-cnn and yolov3," arXiv preprint arXiv:1910.07234, 2019.
- [190] A. Milan, "Ground truth." http://www.milanton.de/data/#gt, 2019.
- [191] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark," in CVPR, June 2009.
- [192] M. C. Nechyba and H. Schneiderman, "Pittpatt face detection and tracking for the clear 2006 evaluation," in *International Evaluation Workshop on Classification of Events, Activities and Relationships*, pp. 161–170, Springer, 2006.
- [193] C. Romeo, "How synthetic datasets generated by a game engine can help train real-world computer vision models," July 2020.
- [194] M. M. Soltani, Z. Zhu, and A. Hammad, "Automated annotation for visual recognition of construction resources using synthetic images," *Automation in Construction*, vol. 62, pp. 14–23, 2016, Elsevier.
- [195] A. Vierling, T. Sutjaritvorakul, and K. Berns, "Dataset generation using a simulated world," in *International Conference on Robotics in Alpe-Adria Danube Region*, pp. 505–513, Springer, 2019.

- [196] Z. Kolar, H. Chen, and X. Luo, "Transfer learning and deep convolutional neural networks for safety guardrail detection in 2d images," *Automation in Construction*, vol. 89, pp. 58–70, 2018, Elsevier.
- [197] M. Neuhausen, P. Herbers, and M. König, "Synthetic data for evaluating the visual tracking of construction workers," in *Construction Research Congress 2020*, 2020.
- [198] Apple Engineers, "Improving the realism of synthetic images," Apple Machine Learning Journal, vol. 1, July 2017.
- [199] C. Atapattu and B. Rekabdar, "Improving the realism of synthetic images through a combination of adversarial and perceptual losses," in 2019 International Joint Conference on Neural Networks (IJCNN), pp. 1–7, 2019.
- [200] N. Takemoto, L. d. M. Araújo, T. A. Coimbra, M. Tygel, S. Avila, and E. Borin, "Enriching synthetic data with real noise using neural style transfer," in *Int. Congress* of the Brazilian Geophysical Society, vol. 78, 2019.
- [201] M. Maximov, K. Galim, and L. Leal-Taixé, "Focus on defocus: bridging the synthetic to real domain gap for depth estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1071–1080, 2020.
- [202] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield, "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 969–977, 2018.
- [203] A. Jain, J. Keller, and M. Popescu, "Explainable ai for dataset comparison," in 2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1–7, IEEE, 2019.
- [204] A. Tsirikoglou, G. Eilertsen, and J. Unger, "A survey of image synthesis methods for visual machine learning," in *Computer Graphics Forum*, vol. 39, pp. 426–451, Wiley Online Library, 2020.
- [205] Motec GmbH, "Motec camera system for construction equipment." https: //www.esquenet.be/sites/default/files/basic-files/construction_ brochure_en.pdf, 2016.
- [206] Motec GmbH, "MOTEC SYSTEM BUS Basis Spezifiktion MSB 2.0," tech. rep., Motec GmbH, Feb. 2018.
- [207] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," IEEE Pervasive Computing, vol. 7, no. 4, pp. 12–18, 2008, IEEE.
- [208] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "VOC2011 Annotation Guidelines." http://host.robots.ox.ac.uk/pascal/VOC/ voc2012/guidelines.html, 2011.

- [209] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results." http: //www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.
- [210] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al., "Tensorflow: A system for large-scale machine learning," in 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI) 16), pp. 265–283, 2016.
- [211] J. Husemann, "Topview Detection via CNNs," project report, Robotics Research Lab, TU Kaiserslautern, Kaiserslautern, Germany, September 30 2018. unpublished, supervised by Axel Vierling.
- [212] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference* on computer vision, pp. 740–755, Springer, 2014.
- [213] P. Dollár, "Piotr's Computer Vision Matlab Toolbox (PMT)." https://github. com/pdollar/toolbox.
- [214] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *Int. J. Comput. Vision*, vol. 88, pp. 303–338, June 2010, Kluwer Academic Publishers.
- [215] K. De and V. Masilamani, "Image sharpness measure for blurred images in frequency domain," *Procedia Engineering*, vol. 64, pp. 149–158, 2013, Elsevier.
- [216] A. Ng, "Machine learning yearning," URL: http://www. mlyearning. org/(96), vol. 139, 2017.
- [217] T. B. Lind, "Quixel CAPTURING THE WORLD." https://quixel.com/blog/ 2018/6/26/capturing-the-world, June 2018.
- [218] J.-H. Kim, "Estimating classification error rate: Repeated cross-validation, repeated hold-out and bootstrap," *Computational statistics & data analysis*, vol. 53, no. 11, pp. 3735–3745, 2009, Elsevier.
- [219] C. Zhang and A. Hammad, "Multiagent approach for real-time collision avoidance and path replanning for cranes," *Journal of Computing in Civil Engineering*, vol. 26, no. 6, pp. 782–794, 2012, American Society of Civil Engineers.
- [220] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015, Springer.
- [221] P. Zhu, L. Wen, D. Du, X. Bian, H. Ling, Q. Hu, H. Wu, Q. Nie, H. Cheng, C. Liu, et al., "Visdrone-vdt2018: The vision meets drone video detection and tracking challenge results," in *Proceedings of the European Conference on Computer Vision* (ECCV) Workshops, pp. 0–0, 2018.

- [222] D. Du, P. Zhu, L. Wen, X. Bian, H. Lin, Q. Hu, T. Peng, J. Zheng, X. Wang, Y. Zhang, et al., "Visdrone-det2019: The vision meets drone object detection in image challenge results," in *Proceedings of the IEEE/CVF International Conference* on Computer Vision Workshops, pp. 0–0, 2019.
- [223] E. Bisong, "Google colaboratory," in Building Machine Learning and Deep Learning Models on Google Cloud Platform, pp. 59–64, Springer, 2019.
- [224] Google Brain, "Configuring the object detection training pipeline." https://github.com/tensorflow/models/blob/master/research/object_ detection/g3doc/configuring_jobs.md, 2021.
- [225] R. Padilla, S. L. Netto, and E. A. da Silva, "A survey on performance metrics for object-detection algorithms," in 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), pp. 237–242, IEEE, 2020.
- [226] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009, IEEE.
- [227] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in European conference on computer vision, pp. 818–833, Springer, 2014.
- [228] Google Brain Team, "Tensorflow 1 model zoo." https://github.com/tensorflow/ models/blob/master/research/object_detection/g3doc/tf1_detection_zoo. md, July 2020.
- [229] N. Qian, "On the momentum term in gradient descent learning algorithms," Neural networks, vol. 12, no. 1, pp. 145–151, 1999, Elsevier.
- [230] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint* arXiv:1609.04747, 2016.
- [231] A. Bal and H. Palus, "A smooth non-iterative local polynomial (snilp) model of image vignetting," *Sensors*, vol. 21, no. 21, p. 7086, 2021, Multidisciplinary Digital Publishing Institute.
- [232] S. Wu, T. Zhao, C. Broaddus, C. Yang, and M. Aggarwal, "Robust pan, tilt and zoom estimation for ptz camera by using meta data and/or frame-to-frame correspondences," in 2006 9th International Conference on Control, Automation, Robotics and Vision, pp. 1–7, IEEE, 2006.
- [233] R. Atienza and A. Zelinsky, "Active gaze tracking for human-robot interaction," in *Proceedings. Fourth IEEE International Conference on Multimodal Interfaces*, pp. 261–266, IEEE, 2002.
- [234] J.-Y. Zheng, T. Sakai, and N. Abe, "Guiding robot motion using zooming and focusing," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS'96*, vol. 3, pp. 1076–1082, IEEE, 1996.

- [235] E. R. Azar, "Active control of a pan-tilt-zoom camera for vision-based monitoring of equipment in construction and surface mining jobsites," in *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction*, vol. 33, p. 1, Vilnius Gediminas Technical University, Department of Construction Economics, 2016.
- [236] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in 2011 IEEE International Conference on Robotics and Automation, pp. 3400–3407, IEEE, 2011.
- [237] E. R. Azar, "Construction equipment identification using marker-based recognition and an active zoom camera," *Journal of Computing in Civil Engineering*, vol. 30, no. 3, p. 04015033, 2015, American Society of Civil Engineers.
- [238] A. Vierling, T. Sutjaritvorakul, and K. Berns, "Crane safety system with monocular and controlled zoom cameras," in *ISARC. Proceedings of the International Symposium* on Automation and Robotics in Construction, vol. 35, (Berlin, Germany), pp. 1–7, IAARC Publications, 2018.
- [239] K. Hata and S. Savarese, "CS231a course notes 1: Camera models." https://web. stanford.edu/class/cs231a/course_notes/01-camera-models.pdf, 2015.
- [240] M. Kisantal, Z. Wojna, J. Murawski, J. Naruniec, and K. Cho, "Augmentation for small object detection," arXiv preprint arXiv:1902.07296, 2019.
- [241] Y. Cai and G. Medioni, "Persistent people tracking and face capture using a ptz camera," *Machine Vision and Applications*, vol. 27, no. 3, pp. 397–413, 2016, Springer.
- [242] Y. Movshovitz-Attias, T. Kanade, and Y. Sheikh, "How useful is photo-realistic rendering for visual learning?," in *European Conference on Computer Vision*, pp. 202– 217, Springer, 2016.
- [243] K. T and R. A., "Object detection on drone videos using caffe* framework," tech. rep., Intel, June 2018.
- [244] Y. Fang and Y. K. Cho, "Measuring operator's situation awareness in smart operation of cranes," in ISARC 2017-Proceedings of the 34th International Symposium on Automation and Robotics in Construction, pp. 96–103, 2017.
- [245] OHS, "Survey Finds High Rate of PPE Non-Compliance." https://ohsonline.com/ articles/2008/11/17-survey-finds-high-rate-of-ppe-noncompliance.aspx, November 2008. [Online; accessed 18-06-2019].
- [246] Manitowoc, "GMK3060 Product Guide." https://www.manitowoccranes. com/~/media/Files/MTWDirect/Grove/AllTerrain/GMK3060/ProductGuides/ GMK3060-Product-Guide-Metric.pdf, 2017.
- [247] Liebherr, "Mobile Crane LTM 1130-5.1," tech. rep., Liebherr-Werk Ehingen GmbH, 2018.
- [248] Renderpeople, "Scanned 3d people pack." https://www.unrealengine.com/ marketplace/en-US/product/9c3fab270dfe468a9a920da0c10fa2ad, 2019.

- [249] Adobe Systems, "Adobe Fuse CC." https://www.adobe.com/wam/fuse.html, 2014. [Online; accessed 25-Jan-2019].
- [250] Mixamo, "Mixamo." http://mixamo.com/, 2008. [Online; accessed 25-Jan-2019].
- [251] Google, "Map of Steil Kranarbeiten GmbH Co. KG, Trier, Germany." https://goo.gl/maps/6vKgnGZ2z9bmfSsz5, (n.d.). [Online; accessed 01-Dec-2021].
- [252] Dekogon Studios, "Construction site vol. 1 supply and material props." https://www.unrealengine.com/marketplace/en-US/product/ construction-site-vol-1-supply-and-material-props, 2019.
- [253] Dekogon Studios, "Construction site vol. 2 tools, parts, and machine props." https://www.unrealengine.com/marketplace/en-US/product/ construction-site-vol-2-tools-parts-and-machine-props, 2019.

Academic CV

Tanittha Sutjaritvorakul

Education

2016 — 2024	PhD, Informatik RPTU Kaiserslautern-Landau, Kaiserslautern, Germany Thesis : Top View Deep Learning Object Detection using Active Perception in Construction Environment Advisor : Prof. Dr. Karsten Berns
2013 — 2015	M. Sc. in Electrical and Computer Engineering TU Kaiserslautern, Kaiserslautern, Germany Thesis : Multiple Pedestrian Tracking Advisor : Prof. Dr. Karsten Berns
2004 - 2008	B. Eng. in Computer and Network Engineering Assumption University, Bangkok, Thailand Thesis : Digital Spectrum Analyzer Advisor : Dr. Kong Kritayakirana