Excavator Pose Estimation for Safety Monitoring by Fusing Computer Vision and RTLS Data

Mohammad Mostafa Soltani

A Thesis
in the Department
of
Building, Civil and Environmental Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of
Doctor of Philosophy (Building Engineering) at
Concordia University
Montreal, Quebec, Canada

October 2017

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: Mohammad Mostafa Soltani

Entitled: **Excavator Pose Estimation by Fusing Computer Vision and RTLS Data for Safety Monitoring**

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

Doctor of Philosophy (Building Engineering)

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

Signed by the final Examining Committee:

Dr. Georgios Vatistas	Chair
Dr. Christian Koch	External Examiner
Dr. Sudhir Mudur	Examiner to Program
Dr. Osama Moselhi	Examiner
Dr. Tarek Zayed	Examiner
Dr. Zhenhua Zhu	Co-Supervisor
Dr. Amin Hammad	Co-Supervisor

Approved by

Dr. Fariborz Haghighat, Graduate Program Director

23/10/2017
Date of Defence

Dr. Amir Asif, Dean, Faculty of Engineering and Computer Science

ABSTRACT

Excavator Pose Estimation for Safety Monitoring by Fusing Computer Vision and RTLS Data

Mohammad Mostafa Soltani, Ph.D.
Concordia University, 2017

The construction industry is considered as a hazardous industry because of its high number of accidents and fatality rates. Safety is one of the main requirements on construction sites since an insecure site drops the morale of the workers, which can also result in lower productivity. To address safety issues, many proactive methods have been introduced by researchers and equipment manufacturers. Studying these methods shows that most of them are using radio-based technologies that perform based on the locations of the attached sensors to the moving objects, which could be expensive and impractical for the large fleet of available construction equipment. Safety monitoring is a sensitive task and avoiding collisions requires a detailed information of the articulated equipment (e.g. excavators) and the motion of each part of that equipment. Therefore, it is necessary to install the location sensors on each moving part of the equipment for estimating its pose, which is a difficult, time consuming, and expensive task. On the other hand, the application of Computer Vision (CV) techniques is growing and becoming more practical and affordable. However, most of the available CV-based techniques evaluate the proximity of the resources by considering each object as a single point regardless of its shape and pose. Moreover, the process of manually collecting and annotating a large image dataset of different pieces of equipment is one of the most time consuming tasks. Furthermore, relying on a single source of data may not only decrease the accuracy of the pose estimation system because of missing data or calculation errors, but it may also increase the computation time. Moreover, when there are multiple objects and equipment in the field of view of each camera, CV-based algorithms are under a higher risk of false recognition of the equipment and their parts. Therefore, fusing the cameras' data with data from Real-Time Location System (RTLS) can help the pose estimation system by limiting the search area for the parts' detectors, and consequently reducing the processing time and improving the accuracy by reducing the false detections.

This research aims to estimate the excavator pose by fusing CV and RTLS data for safety monitoring and has the following objectives: (1) improving the CV training by developing a method to automatically generate and annotate around-view synthetic images of equipment and their parts using the 3D model of the equipment and the real images of the construction sites as background; (2) developing a guideline for applying stereo vision system in construction sites using regular surveillance cameras with long baseline at a high level; (3) improving the accuracy and speed of CV detection by fusing RTLS data with cameras' data; and (4) estimating the 3D pose of the equipment for detecting potential collisions based on a pair of Two Dimensional (2D) skeletons of the parts from the views of two cameras.

To support these objectives, a comprehensive database of the synthetic images of the excavator and its parts are generated, and multiple detectors from multiple views are trained for each part of the excavator using the image database. Moreover, the RTLS data, providing the location of the equipment, are linked with the corresponding video frames from two cameras to fuse the location data with the video data. Knowing the overall size of the equipment and its location provided by the RTLS system, a virtual cylinder defined around the equipment is projected on the video frames to limit the search scope of the object detection algorithm within the projected cylinder, resulting in a faster processing time and higher detection accuracy. Additionally, knowing the equipment ID assigned to each RTLS device and the cameras' locations and heights, it is possible to select the suitable detectors for each equipment. After detecting a part, the background of the detected bounding box are removed to estimate the location and orientation of each part. The final skeleton of the excavator is derived by connecting the start and end points of the parts to their adjacent parts knowing the kinematic information of the excavator. Estimating the skeleton of the excavator from each camera view on one hand, and knowing the extrinsic and intrinsic parameters of all available cameras on the construction site, on the other hand, are used for estimating the 3D pose by triangulating the estimated skeleton from each camera. In order to use the available collision avoidance systems, the 3D pose of the excavator is sent to the game environment and the potential collisions are detected followed by generating a warning.

The contributions of this research are: (1) developing a method for creating and annotating the synthetic images of the construction equipment and their parts using the equipment 3D models and the real images of the construction sites; (2) creating and training the HOG-based excavator's parts detectors using the database of the synthetic images developed earlier and automatically produced negative samples from the other excavator parts in addition to the real images of different construction sites while the target object is cut from these; (3) developing a data fusion framework after calibrating two regular surveillance cameras with the long baseline to integrate the RTLS data received from GPS with the video data from the cameras to decrease the processing efforts for detecting excavator parts while increasing the detection accuracy by limiting the search scope for the detectors; (4) developing a clustering technique to subtract parts' background and extracting the 2D skeleton of the excavator in each camera's view and to estimate the 3D pose of the excavator; and (5) transferring the 3D pose data of the excavator to the game environment using

TCP/IP connection and visualizing the near real-time pose of the excavator in the game engine for detecting the potential collisions.

ACHNOWLEDGEMENT

I cannot start before expressing my greatest appreciation to my supervisor, Dr. Amin Hammad for his intellectual and personal support, and for his patience, motivation, enthusiasm, and immense knowledge. His trust in me, advice, and criticism encouraged me to fulfil my Ph.D. at my best. Also, I would like to thank my co-supervisor, Dr. Zhenhua Zhu, for his valuable academic supports. His vision helped me to consider the different aspects of my solutions.

I feel very fortunate for having my exceptional research colleagues. I cannot imagine how hard could be the course of my study without their kindness and support. Also, I am also very thankful to Mr. Khaled El-Ammari for his support in developing the virtual environment in Unity. He is a true master in his work. I would also like to thank the member of my committee, Dr. Christian Koch, Dr. Osama Moselhi, Dr. Tarek Zayed, and Dr. Sudhir Mudur for spending their precious time reading this thesis and for providing me with constructive feedback and suggestions. Indus.ai company (and it parent company, GreenOwl Mobile Inc.) was a great encouragement and motivation for applying my research in the real world. I would like to thank Mr. Matt Man, Mr. Bill Hollings, Mr. Walid Aly, and all my colleagues at GreenOwl Mobile.

When it comes to my parents, Mitra Yazdabad and Khosro Soltani, I can remember every hard time that I had during my study, since they provided me the support and hope to pass through those times. God bless them!

This research has been supported by Fonds Québécois de la Recherche sur la Nature et les Technologies (FQRNT) Bourse de Doctorat en Rechercheh, Concordia University Guy Bourassa Graduate Scholarship, and the Natural Sciences and Engineering Research Council (NSERC) Engage grant.

Dedicated to my lovely parents and those who lost their beloveds in the construction accidents

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Description
2D	Two Dimensional
3D	Three Dimensional
AR	Augmented Reality
AWCBC	Association of Workers' Compensation Boards of Canada
BIM	Building Information Modelling
CAD	Computer-Aided Design
CPU	Central Processing Unit
CRAV	Covered Range of Angles View
CV	Computer Vision
DEW	Dynamic Equipment Workspace
DyCE	Dynamic Communication Environment
EM	Expectation-Maximization
FoV	Field of Views
FN	False Negative
FP	False Positive
GPS	Global Positioning System
GPU	Graphical Processing Unit
HOC	Histogram of Color
HOG	Histogram of Oriented Gradients

HPC	High Performance Computing
HSV	Hue, Saturation, and Value
Hz	Hertz
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
IR	Infrared
IT	Information Technology
LAEW	Look-Ahead Equipment Workspaces
LiDAR	Light Detection And Ranging
MIMD	Multi-Instruction Multi-Data
MAS	Multi-Agent System
MSDF	Multi-Sensory Data Fusion
NICS	National Institute for Computational Science
NRT	Near Real-Time
OA	Operator Agents
PC	Personal Computer
PCA	Principal Component Analysis
PTZ	Pan–Tilt–Zoom
RAM	Random-Access Memory
RF	Radio Frequency
RFID	Radio Frequency Identification
RGB	Red, Green, and Blue
ROI	Region of Interest
RTLS	Real-Time Location System

SIFT	Scale-Invariant Feature Transformation
SURF	Speeded-Up Robust Features
SVM	Support Vector Machine
TCA	Team Coordinator Agents
TCP/IP	Transmission Control Protocol / Internet Protocol
TN	True Negative
TP	True Positive
USB	Universal Serial Bus
UWB	Ultra-Wideband
VICE	Virtual Interactive Construction Education
VR	Virtual Reality
WLAN	Wireless Local Area Network
XML	Extensible Markup Language

CHAPTER 1 INTRODUCTION

1.1 General Background

The construction industry is considered as a hazardous industry because of its high number of accidents and fatality rates (Haupt, 2001; Blough, 1983; Churcher & Alewani-Starr, 1997; Brown, 1997). Safety is one of the main requirements in construction sites since an insecure site drops the morale of the workers, which can result in lower productivity and may affect their personal lives. On the other hand, any accident may put the company in financial troubles (Zhang et al., 2012a). According to an article presented by the Centre for Construction Research and Training, 6,678 deaths have been reported from 1992 through 2010 in the USA, which means more than two persons per day (CPWR, 2013). The main portion of work-related fatalities is caused by the collisions of the workers with the objects and equipment. The statistics confirms that there is a serious need for introducing new approaches in order to increase the safety of construction sites and to reduce the risk of accidents. The cultural and the educational training programs for the workers are also necessary to develop the pro-active safety measurement and avoidance. Additionally, the construction industry, with the annual revenues of more than CA$110 billion in Canada (Government of Alberta, 2011) and US$1.7 trillion in the USA (AEIOHS, 2011), plays a significant role in the economy of North America. The economic and daily execution challenges are motivating efforts in practice and academia to improve productivity of construction operations while keeping them safe.

Addressing the aforementioned concerns, many studies were done to help the construction industry reducing the number of accidents by taking advantages of the available and emerging technologies.

These technologies, which are adapted to the specialized safety purposes includes: Building Information Modelling (BIM) (Guo, 2002; Akinci et al., 2002; Kiviniemi et al., 2011), vision-based systems (Chi & Caldas, 2011; Kandil et al., 2012a; Kandil et al., 2012b), serious games (Xie et al. 2006; Zhao et al. 2009; Charsky, 2010; Lin et al. 2011; Rüppel and Schatz, 2011), location-based systems such as Ultra-Wideband (UWB) (Zhang & Hammad, 2011), Radio Frequency Identification (RFID) technology (Helmus, 2007; Friedlos, 2008; Swedberg, 2008; Soltani, 2010; Chae and Yoshida, 2010; Kelm & Laussat, 2010 ; Helmus et al., 2011;); Global Positioning System (GPS) (Oloufa et al., 2003; Pradhananga & Teizer, 2013), and laser scanners (Wang et al., 2014). These technologies have been used in many research to address safety concerns (Fullerton et al., 2009; Hammad et al., 2011; Zhang et al., 2012a; Safety Shield Systems, 2012; Zhang & Hammad, 2012; Zhang et al., 2012b; Zhang et al., 2013).

1.2 **Problem Statement and Research Gaps**

In spite of the abovementioned technologies in the previous section, collisions between the static and the dynamic objects are still the main source of accidents in the construction industry. Therefore, there is a vital need for investigating more effective solutions that can detect collisions on the construction sites. The Near Real-Time (NRT) pose of the equipment is one of the most valuable requirements for evaluating the equipment safety.

In the earth moving operations, which are highly dependent on the equipment such as excavator, monitoring the safety can be determined by extracting the three Dimensional (3D) pose of the equipment. For this purpose, each part or joint of the equipment must be recognized and tracked. Although the accuracy of some of these systems is high, deploying on construction sites is difficult and sometimes impossible. Another challenge with radio-based technologies is sending and receiving omnidirectional signals in the construction site environment, which has different unpredictable objects (e.g. metallic objects) that can block or absorb the signals. The main limitation for applying any kind of the aforementioned technologies is the need to install tags, sensors, or markers on each individual resource, which is costly and/or time consuming (Teizer et al., 2007). Moreover, the high cost of the laser devices makes their deployment unaffordable for many construction contractors.

One of the growing technologies is Computer Vision (CV) based monitoring and tracking system, which is easier and cheaper to deploy compared to other technologies. The recent growth of CV-tracking systems is not limited to research and academia. Service providers are offering fully automated solutions for monitoring the safety and productivity in construction sites using CV-based methods. For instance, indus.ai, a Canadian service provider, is offering CV-based performance and safety analytics system for construction sites (indus.ai, 2016). The popularity of CV-based methods shows the applicability and affordability of these methods in construction industry. CV techniques for the pose estimation depend on detecting the parts of an articulated equipment (e.g. excavator). However, most studies in this area focused on using CV methods for detecting and localizing the construction equipment as one object regardless of the orientations of its parts (Zou & Kim, 2007; Park & Brilakis, 2012a; Azar & McCabe, 2011; Zhang et al., 2012b). Additionally, the main prerequisite for object recognition is to train the object detectors so that they will be able to find similar objects in the new images. Usually, training the detectors is a time consuming and sensitive task, which has a direct effect on the accuracy of the object recognition results. Furthermore, the blind training of a detector with a large number of samples without a solid structure can make the detector confused and create other problems such as fewer true positive detections or more false positive detections even after spending a lot of time in training with those samples. Studying the process of training the object detectors shows that annotating the Region of Interest (ROI) within each image needs a lot of time and effort (Jeon et al., 2003; Yan et al., 2008). Moreover, relying only on cameras' data may not only decrease the accuracy of the pose estimation system because of missing data or calculation errors but also it may increase the computation time. Therefore, integrating additional source of data such as Real-Time Location Systems (RTLS) should be investigated.

1.3 Research Objectives

The absence of a system with the ability to track the pose of construction equipment and to avoid collision, brings the attention of this research to investigate the possible methods. This research aims to achieve the following objectives: (1) improving the CV training by developing a method to automatically generate and annotate around-view synthetic images of equipment and their parts using the 3D model of the equipment and the real images of the construction sites as background; (2) developing a guideline for applying stereo vision system in construction sites using regular surveillance cameras with long baseline at a high level; (3) improving the accuracy and speed of CV detection by fusing RTLS data with cameras' data; and (4) estimating the 3D pose of the equipment for detecting potential collisions based on a pair of Two Dimensional (2D) skeletons of the parts from the views of two cameras.

1.4 Thesis Organization

The structure of this study is presented as follows:

Chapter 2 Literature Review: This chapter reviews the current state of the concepts, statistics, technologies, methods, and alternative solutions that may be used in this research.

Chapter 3 Research Framework: The overview of this research and the overall proposed framework are discussed briefly in this chapter. It includes the explanation of the three main modules that are used in the research.

Chapter 4 Auto Annotation of Synthetic Images: This chapter goes through the details of the proposed method for auto generation and annotation of the synthetic images and demonstrates the feasibility of the method using a proof-of-concept case study.

Chapter 5 Data Fusion and Excavator Parts Detection: The methods and techniques proposed to fuse the RTLS and video data followed by the process of detecting the excavator's parts are introduced and validated in this chapter.

Chapter 6 Skeleton Extraction and 3D Pose Estimation: In this chapter, the proposed method for subtracting the background and extracting the skeleton of the equipment is explained in detail. Moreover, the approach of estimating the 3D pose of the excavator and transferring the 3D pose data into game environment is introduced. These methods are used for detecting the potential collisions and finally the proposed method is evaluated through the case studies.

Chapter 7 Summary, Contributions, and Future Work: The summary of the work done in this research is provided in this chapter followed by the contributions. The further possibilities for extending and improving the current research are explained in the future work section.

CHAPTER 2 LITERATURE REVIEW

2.1 Introduction

In this chapter, the status of the current practices in construction projects' safety, available localization and RTLS, and CV methods are reviewed. The applications of Multi-Sensory Data Fusion (MSDF) and serious games for improving safety in construction sites. Finally, the limitations and research gaps in the available methods are highlighted.

2.2 Safety status and monitoring in Construction Projects

Reviewing the work-related fatalities in Canada shows that the construction industry has the highest rank of the accidents in Canada. Association of Workers' Compensation Boards of Canada (AWCBC) announced that a total number of 616 fatalities occurred in the construction industry between 2011 and 2013, which is 22% of the total work-related fatalities in Canada (AWCBC, 2014). Furthermore, AWCBC mentioned that the province of Alberta had the highest fatality rate in construction while Ontario, British Columbia, and Quebec had the highest number of fatalities after Alberta during that period of time.

Investigating the statistics provided by Alberta Occupational Health and Safety confirms that the construction industry in this province had the highest rate of the work-related fatalities between 2009 and 2013 with the rate of 37% among all industries in Alberta (WCB – Alberta, 2014).

Figure 2-1 shows the percentage of the construction industry's fatalities by the type of event between 2001 and 2010 in Alberta. Contacts with objects and equipment were the main of source of fatalities with 31% of all types of fatalities (Government of Alberta, 2011).

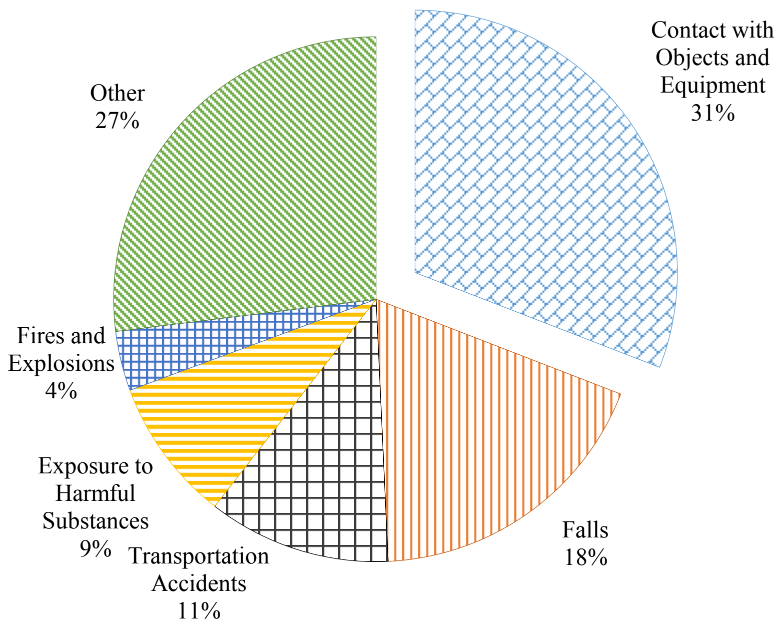


Figure 2-1. Fatalities in Alberta by type of event (Government of Alberta, 2011)

Moreover, 14% of the total number of fatalities between the years 1996 and 2003 in the U.K. have been caused by being struck by a moving vehicle (Howarth & Watson, 2009). Wu et al. (2010) stated that 428 equipment-related struck by and caught-in/between accidents were reported between 1995 and 2008 in the USA.

While studying the recorded accident scenarios that led to the fatality due to collision with the equipment on the construction sites in Alberta, a tragic accident report was found, which it is titled "Construction Foreman Killed in Motor Vehicle Collision". Studying this report shows that although there was a traffic control flag man at the construction zone but the lack of communication between the foreman and the equipment operator caused the accident. However, the operating procedure did not specify any communication requirements (AEIOHS, 2011). From this report, it can be understood that if there was a safety monitoring system, with the ability to observe and analyze the interactions between the equipment, vehicles, and workers and catch the potential collisions and warning the related people beforehand, could eliminate the need of continues communications between the operators, drivers, and workers. Therefore, automated monitoring technologies are greatly needed to improve the safety of construction projects.

Vahdatikhaki and Hammad (2015a) developed a method for generating Dynamic Equipment Workspaces (DEWs) to improve earthwork safety using RTLS. As shown in Figure 2-2, DEWs are generated virtually around the excavator, depending on the movement's speed of the parts and the dimension of the excavator. The location and pose information of the equipment are obtained from a UWB system. However, deploying the UWB system on the construction site requires installing multiple tags on the parts of the excavator and multiple sensors on the site to collect the location information of the tags. Moreover, the generated DEW may not be efficient since it only relies on the information of each equipment individually while it does not consider the risks' priorities in 3D space resulted from the interactions between each equipment and its adjacent equipment. For instance, when the excavator is loading its bucket, there will be a lower risk of accident for the objects behind the excavator.

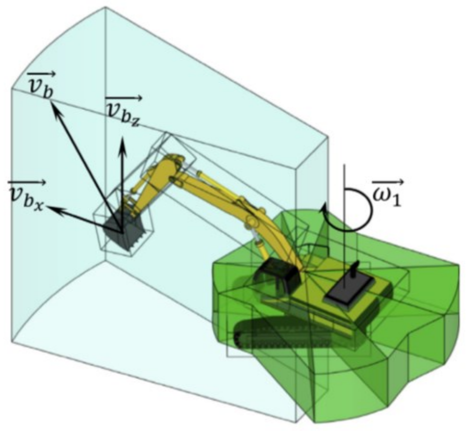
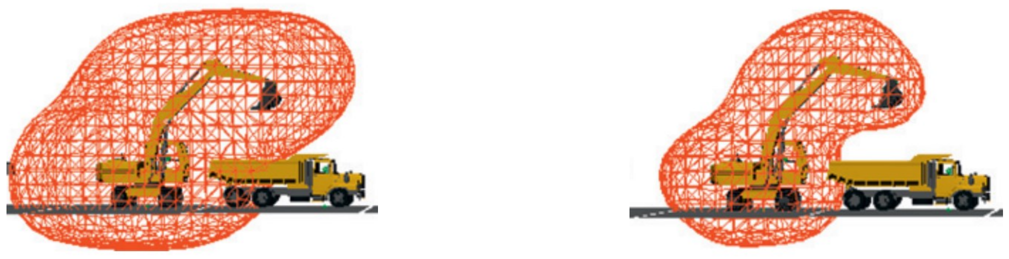

$\overrightarrow{v_b}$
$\overrightarrow{v_{b_z}}$
$\overrightarrow{v_{b_x}}$
$\overrightarrow{\omega_1}$

Figure 2-2. Schematic 3D representation of DEW of an excavator in swinging state (Vahdatikhaki & Hammad, 2015a)

Vahdatikhaki and Hammad (2015b) proposed generating equipment risk maps based on the proximity-based and visibility-based risks using the pose and state data of the equipment relative to its adjacent equipment and NRT simulation. Moreover, they used the equipment risk maps to generate Look-Ahead Equipment Workspaces (LAEWs) that can be used to identify the potential collisions. The system allows the user to customize the acceptable risk level. This would help to reduce the number of false alarms when two equipment (e.g. excavator and truck shown in Figure 2-3(a)) have to work very close to each other by increasing the level of acceptable risks (result is shown in Figure 2-3(b)). In another scenario, the user can select a lower acceptable risk for the interactions between the workers and the equipment to make the system more sensitive to the potential collisions. It should be mentioned that the CV-based methods for monitoring the safety of the construction projects are explained in Section 2.6.3.



(a) Risk level of 0.8 (b) Risk level of 0.9

Figure 2-3. Side views of LAEW of the truck (Vahdatikhaki & Hammad, 2015b)

2.3 Productivity Monitoring in Construction Projects

The construction industry, with the annual revenues of more than $110 billion in Canada (Statistics Canada, 2016) and $1.7 trillion in USA (Statistic Brain, 2016), plays a significant role in the economy of North America. The economic and the daily execution challenges are motivating efforts in practice and academia to improve the productivity of construction operations. Monitoring the construction equipment occupies a great portion of the aforementioned effects (Motwani et al., 1995; Pradhan et al., 2011). Knowing the location and pose of the equipment can help construction companies track their productivity by translating the location and pose information into pre-defined states of the equipment (Rodriguez, 2010). Vahdatikhaki and Hammad (2014) proposed a rule-based system that converts sensory data to states and comprises a knowledge base and a reasoning mechanism. The system covers all the rules and heuristics, which identify the states and their current phase in the operation. The productivity of the excavator can be estimated using the states of the excavator and the duration spent in each state. Knowing that the activity recognition methods have a direct relationship with the productivity estimation methods, these methods using CV algorithms are explained in Section 2.6.2

2.4 **Localization and RTLS Technologies**

The localization problem has received considerable attention in the area of pervasive computing as many applications need to know where the objects are located. Location information is central to personalized applications in areas such as transportation, manufacturing, logistics, and healthcare, and it is the basis for the delivery of personalized and Location-Based Services (Papapostolou & Chaouchi, 2011; Li & Becerik-Gerber, 2011). Furthermore, the precise objects location information can be used for several applications (Zhou & Shi , 2009) such as finding missing items in a storehouse (Hariharan, 2006), locating equipment in construction sites (Song et al., 2006), mobile users localization inside a building (Ji et al., 2006), collision prevention between vehicles (Tong & Zekavat, 2007), and rescuing persons in underground mines (Zhang & Yuan, 2006). Monitoring personnel movements, material locations, and construction equipment can effectively make the management of projects more productive (Khoo, 2010; Ibn-Homaid, 2002; Fan et al., 2008; Yagi et al., 2005; Grau et al., 2009).

2.4.1 Localization Levels

Papapostolou and Chaouchi (2011) defined localization as the procedure of estimating the current position of a user or an object within a specific region, indoor or outdoor. The determined position can be represented in various ways (e.g., coordinates, region, cell, hierarchical) based on the desired application or the positioning system specifications.

Razavi and Haas (2011) discussed two methods of localization: *fine-grained* localization using detailed information and *coarse-grained* localization using minimal information. Minimal techniques are easier to perform, need fewer resources and have lower equipment costs; however their accuracy is lower than detailed information techniques. Fine-grained node localization using measurement techniques can be classified into broad types based on time of flight, received signal strength, lateration and angulation, distance-estimation using time difference of arrival, pattern matching, and Radio Frequency (RF) sequence decoding techniques (Razavi & Haas, 2011). Coarse-grained node localization uses range-free or connectivity-based localization algorithms with no needs for any measurement techniques. In this class, some anchor sensors have stored information about their own location. Therefore, the locations of other sensors can be calculated based on connectivity information, such as detecting which sensor is within the broadcasting range of which other sensors. The methods in this class determine the closest known locations to the object instead of measuring the distance between an object and reference points (Bulusu et al., 2000; Simic and Sastry, 2002; Song et al., 2006). Also, tracking the physical phenomena that have limited range (e.g., physical contact with a magnetic scanner or communication connectivity to Access Point in a wireless cellular network) helps to determine the presence of an object within a specific range (Razavi & Haas, 2011).

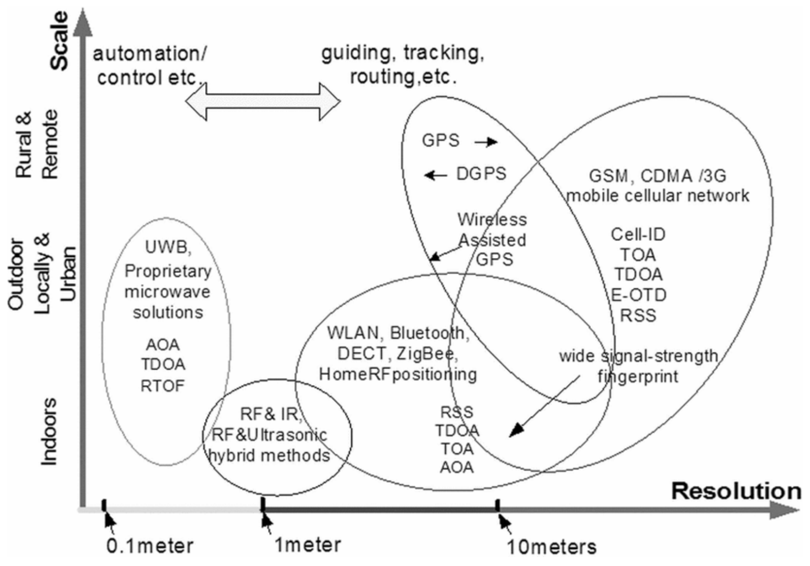
2.4.2 Localization Technologies

Basically, localization systems rely on ultrasound, magnetic, infrared, vision, RF technology, and/or CV (Pradhan et al., 2009; Hightower & Borriello, 2001). Ultrasound-based systems include a transmitter to emit the ultrasound pulses and a receiver to collect the emitted pulses and to estimate the distance between the receiver and the transmitter (Pradhan et al., 2009; Want et al., 1992). Although ultrasound-based systems have high accuracy, they need a large number of sensors, which are costly compared to RF systems (Pradhan et al., 2009; Hightower and Borriello, 2001).

Magnetic approaches are based on the measurement of the motion with the help of accelerometers, gyroscopes, and magnetometer. The rate of the motion (i.e., acceleration) comes from an accelerometer and the type and direction of the motion are provided by gyroscopes (Fraden, 2011). Additionally, a magnetometer helps the system to find the direction of motion with respect to the earth's magnetic field. Performing dead reckoning technique (Gelb, 1974) with a known rate, type and direction of motion estimates the location of an object based on an Inertial Measurement Unit (IMU). In recent years, Pradhan et al. (2009) and Jimenez Ruiz et al. (2012) proposed combinatorial methods using RFID to improve the accuracy of IMU-based localization methods.

Similar to ultrasound, infrared-based systems include a transmitter and a receiver, but they utilize electromagnetic radiation of wavelength greater than the visible light instead of sound waves to determine the distance (Pradhan et al., 2009M). Furthermore, image-based localization technologies rely on edge detection, feature recognition, and landmark detection using visual tags or image matching (Sim & Dudek, 2003) but they are mainly suitable for self-localization.

The systems that use RF in different ways to localize the position of a target are called RF-based solutions. GPS, RFID, Wireless Local Area Network (WLAN), Bluetooth, and ZigBee are some of the popular RF-based technologies used for localization. Due to their communication range and the possibility to work without any line of sight (except GPS), there is a strong trend to use these solutions for localization. Instead, GPS does not required any local infrastructure for calculating its position and is easy to deploy compared to other RF-based solutions which rely on the fixed local recievers. Figure 2-2 depicts a general overview of the current wireless-based positioning systems (Liu et al., 2007; Vossiek et al., 2003).



Scale
automation/ control etc.
guiding, tracking, routing,etc.
Rural & Remote
Outdoor Locally & Urban
Indoors
GPS
DGPS
Wireless Assisted GPS
GSM, CDMA /3G mobile cellular network
Cell-ID
TOA
TDOA
E-OTD
RSS
UWB, Proprietary microwave solutions
AOA
TDOA
RTOF
WLAN, Bluetooth, DECT, ZigBee, HomeRFpositioning
wide signal-strength fingerprint
RF& IR, RF&Ultrasonic hybrid methods
RSS
TDOA
TOA
AOA
Resolution
0.1meter
1meter
10meters

Figure 2-4. Outline of current wireless-based positioning systems (Liu et al., 2007)

2.5 Computer Vision Methods and Applications

Basically, subjects under the application of CV cover by object recognition, object tracking, and activity recognition, which is usually advancement of integration between the object recognition and the object tracking. Object recognition focuses on detecting and recognizing the object within the images. Following the movement of the specific pixels within the consequent frames is called object tracking. Tracking the objects starts by detecting the objects and then tracking it. Furthermore, analyzing the output of the object recognition and tracking methods help the system to recognize the ongoing activity. In the following, the studies related to the application of CV in construction project are reviewed.

2.5.1 Object Recognition and its Applications

Currently, there are few extensive reviews on the available object recognition methods in CV. Andreopoulos and Tsotsos (2013) explained the development of these algorithms in the past 50 years including the requirements for the applications of these algorithms in the area of construction sites' monitoring. Matas and Obdrzalek (2004) and Yang (2009) divided these methods to geometry-based, appearance-based, and local feature-based methods, which are briefly discussed in the following (Tajeen & Zhu, 2014).

2.5.1.1 Geometry-Based Methods

The primitive elements of the objects such as lines, circles, etc., are the main focus of these methods. The extracted primitives of the object are hierarchically organized in the bags of boundaries (Payet & Todorovic, 2011) as the objects' templates for further comparison with the query images. These methods are invariant to the viewpoint and illumination under small degrees of occlusions, background clutter, or light variations (Mundy, 2006). The Edge detection methods such as Sobel, Prewitt, Canny, Laplacian of Gaussian, Expectation-Maximization algorithm, etc., are the main part of geometry-based methods (Ramadevi et al., 2010).

2.5.1.2 Appearance-Based Methods

The main elements in this class are the color and the texture of the object, which are extracted from the images and represented as a histogram. Then, the classifier is trained using the histograms of the object under the different illumination and pose variations. Although the appearance-based methods are robust to the variations of the object orientations and scales, they suffer from big changes of the lighting conditions and cluttered backgrounds (Matas & Obdrzalek, 2004; Tajeen & Zhu, 2014).

2.5.1.3 Local Feature-Based Methods

The local features of an object are typically invariant to the scale, illumination, and affine transformation (Matas & Obdrzalek, 2004; Grauman & Leibe, 2011; Carr et al., 2012; Tajeen & Zhu, 2014). The Scale-Invariant Feature Transformation (SIFT) (Lowe, 1999), the Histogram of Oriented Gradients (HOG) (Dalal & Triggs, 2005), and the Speeded-Up Robust Features (SURF) (Bay et al., 2006) are the common examples of local feature-based methods. The basic concept of these methods is that the descriptor first learns the interest points of the objects from the training dataset and then searches for the similar objects within the query images (Yang, 2009). Considering the nature of the construction projects with cluttered backgrounds and the continuous change in the light conditions, this class is expected to provide a better and more reliable results on the construction sites while the first two classes may outperform in the environments where the illumination condition is under control and there are few changes in the background (e.g. production line of factories).

2.5.2 Image Annotation Techniques

For creating the supervised object detectors, the training process of the object detectors starts by annotating many positive samples, which contain that object. Typically, the samples are observed one by one and the target object in each sample is identified by a rectangular bounding box to indicate the regions that are occupied by the object. The coordinates of the bounding box are considered as ROI during the training phase. In the following, different approaches for image annotation are studied; however they suffer from the long required time for manual annotation by the human.

There are a few toolboxes available for annotating and labeling the images that can be used for the training purpose. Von Ahn and Dabbish (2004) and Von Ahn et al. (2006) proposed an online computer game named Peekaboom for labeling images through an interactive system. The game is arranged so that the user determines the contents of the images by choosing the meaningful labels for them.

LabelMe is one of the well-known web-based tools in this area (Russell et al., 2008). (Fellbaum, 1998), discover the object parts, recover the depth ordering of the objects in a scene, and increase the number of the labels using minimum human supervision. A semi-automatic labeling tool in this application is able to suggest the additional labels in the new images once there are enough annotations of a particular object class. However, to achieve good results, it is necessary first to annotate many images for the target object class, which needs a lot of time, then to train a descriptor based on the annotated images. Furthermore, the user has to read the suggested labels for the new images and approve them manually, which also requires more time. There is another annotation tool that was developed by researchers at the University of Bonn based on the LabelMe toolbox (Kor & Schneider, 2007). This tool is a Matlab-based software and the user is able to manually draw a boundary around the object. Afterward, the user defines the class of the object, the view, and the sub- or super-classes of the object. At the end, an XML (Extensible Markup Language) file containing the coordinates of the boundary lines and related information is provided to the user.

Kläser (2007) created the Image Annotation Tool for the image annotation with the pixel wise masks. This software is able to label the objects by manually defining bounding boxes. Furthermore, the users can apply simple modifications such as rotation, alignment, etc. However, this tool does not have much more advantages than the tools mentioned earlier and it still suffers from the long required time for annotating each object.

Amazon Mechanical Turk (Sorokin, 2009) is another web-based annotation tool. In this tool, the user defines an annotation protocol out of four existing ones and determines what objects need to be labeled. The four protocols are two coarse object segmentation protocols, polygonal labeling, and 14-point human landmark labeling. Sorokin and Forsyth (2008)claimed that the effort cost of their tool is lower than the other existing ones. However, this tool still relies on a large amount of human efforts.

Mathworks (2014a) provides an easy way for annotating positive samples named Training Image Labeler, which is a simplified version of other tools within Matlab. The user can specify ROIs within each image interactively and receive a .mat file containing the name of the images and the corresponding bounding boxes in each image.

2.5.3 Image Segmentation Techniques

One of the methods for annotating the images automatically is the image segmentation to separate the target object from the background to do the further processing steps; therefore the image segmentation is the first important step of low-level vision (Pal & Pal, 1993). It is a process of partitioning the image into non-intersecting regions using automatic thresholding, edge-based methods, and/or morphology-based methods. Assuming an excavator as the target object, the image has to be partitioned so that the whole body of the excavator appears in one segment and the rest of the image appears in the other segments. One of the necessary steps in segmentation is to detect the edges of the object. The main edge detection algorithms are Sobel, Prewitt, Roberts, and Canny (Aybar, 2006). To make the edges more consistent, the output of edge detection should be dilated (Mathworks, 2014b). This way, the holes within partitioned segments could be filled through a connectivity mask based on morphological reconstruction (Soille, 1999). In this regard, JMicrovision (Roduit, 2006) provides a wide range of functions for morphology operating, filtering, and segmentation. Furthermore, Shoelson (2012) developed another Matlab-based application named SegmentTool, which is an interactive Graphical User Interface for segmenting images. However, both tools need efforts from the users for selecting the required thresholds to partition the target correctly. There are other tools, which have the function for segmenting the target object based on the predefined classes but they need first to be trained by different positive samples. However, these tools are not useful here because they assumed that there was no model of the target object and the segmentation results would be directly used for training.

2.5.4 Model-based Synthetic Images

Generally, the images, which are created using an artificial process such as computer-graphics techniques are called synthetic images. In contrast with the images taken by a camera from real scenes, synthetic images can be taken by a virtual camera within a 3D environment developed in a computer software. One of the early attempts to use the images of 3D models was done by Blanz et al. (1996), where they compared two object recognition algorithms using realistic 3D models. However, the images taken from the 3D models are used for both training and testing, which shows that their main focus was on the object recognition. Later, Blanz et al. (2002) proposed the fitting of a single image of a face to the 3D model of the face to identify the faces across different poses and illuminations.

Heisele et al. (2009; 2013) proposed an active learning method for object classification using 3D models. In this patent, the classifier was trained based on the synthetic images rendered from realistic 3D computer graphics models. However, the generated images had single background color, which could represent a natural scene. This research clearly showed the benefits of learning from 3D models, which are the possibility of obtaining a ground truth and controlling the synthetic image generation process through a small number of rendering parameters.

Liebelt and Schmid (2010) introduced a method for including the 3D geometry from synthetic Computer-Aided Design (CAD) models into a 2D appearance detection method. They generated the synthetic images to support their multi-view object dataset. This approach relies exclusively on a database of synthetic 3D models to represent both the appearance and the geometry of the object class (Schels et al., 2011). However, the created images using this method still do not look natural since they do not contain any real backgrounds.

In summary, none of the above-mentioned studies focused on the annotation of synthetic images for the training of the object detectors in order to reduce the time and human efforts required in manual annotation methods.

2.5.5 Camera Calibration

Although qualitative analysis of the videos can be done using uncalibrated cameras (Hartley, 1993; Salvi et al., 2002), the camera calibration is necessary for the quantitative analysis. Extracting useful information from the video frames requires the calculation of camera parameters

(Nehemiah, 2016). The situation becomes more critical when the analysis is within the stereo vision or multi-vision domain. In this case, not only the cameras have to be calibrated individually but also they have to be externally calibrated relative to each other (translation and rotation of each camera from an origin point).

Salvi et al. (2002) divided the process of camera calibration into two phases, the mathematical modeling of the camera by using a set of parameters, and estimating the values of the cameras' parameters (known as intrinsic and extrinsic parameters). Basically, intrinsic parameters refer to the coordinates of the principal point, scale factors, and skewness of two image axes (Zhang Z. , 2000), and extrinsic parameters represent the position and orientation of the camera with respect to a world coordinate system (Salvi et al., 2002). One of the widely used methods for estimating camera parameters is the *flexible camera calibration* by viewing a plane from unknown orientations (Zhang Z. , 1999). According to this method, a pattern (e.g. checker board) is used to detect the feature points in the images. These points are used to estimate camera parameters using Equation 2-1,

$$s[x \quad y \quad 1] = [X \quad Y \quad Z \quad 1]\begin{bmatrix} R \\ t \end{bmatrix} K \qquad \text{where} \qquad K = \begin{bmatrix} \alpha & c & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$ **Equation 2-1**

where *X*, *Y*, and *Z* are the world coordinates of each point, *x* and *y* are the coordinates of the corresponding image point, *s* is the arbitrary scale factor, *R* is the 3D rotation matrix of the camera, *t* is the translation of the camera relative to the world coordinate system, and *K* is camera intrinsic matrix. Within *K*, (u_0, v_0) are the coordinates of the principal point, α and β represent the scale factors in image u and v axes, and c represents the skewness of the two axes. The intrinsic and extrinsic parameters are estimated using the closed-form method explained in (Zhang Z. , 1999).

Later on, the distortion coefficients are approximated using nonlinear least-square minimization technique.

2.5.6 Stereo Vision

As shown in Figure 2-5, stereo vision uses two or more cameras with overlapping views (Torr & Zisserman, 2000). The relationship between two views is represented by the vector $\begin{bmatrix} R \\ t \end{bmatrix}$ where R and t are the rotation and translation of the coordinate system of camera 2 from that of camera 1. This relationship can be extracted based on the motion of points between the two views.

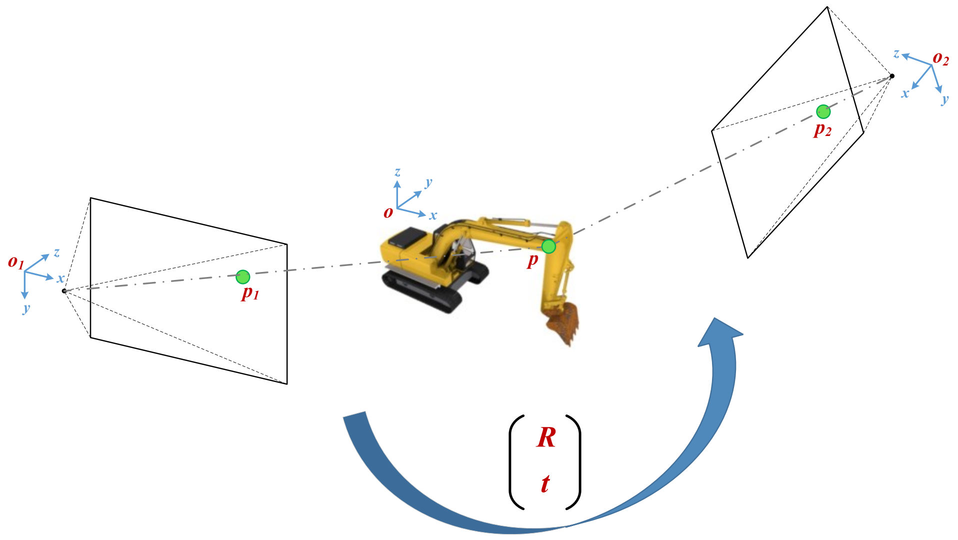


Figure 2-5. Stereo vision basic concept

In simple words, each camera can have its own world coordinate system (e.g. o_1 is the origin of camera 1 and o_2 is the origin of camera 2) but when they are working together to provide 3D information, their individual coordinate systems should be unified (new world coordinate system with the origin of o), which is called stereo cameras calibration. Assuming that the parameters of each camera are extracted by individual calibration of the cameras and the image coordinates from each camera are normalized, then the *essential matrix* E shows the relations between the two cameras. Knowing the *fundamental matrix* F for any pair of corresponding points $p_1 \leftrightarrow p_2$ (Equation 2-2), the equation for the essential matrix, which is the special case of the fundamental matrix for the normalized image coordinates, is defined by Equation 2-3 for any pair of normalized image coordinates for corresponding points $\hat{p}_1 \leftrightarrow \hat{p}_2$ (Hartley & Zisserman, 2003). The methods proposed by (Nistér, 2004; Kukelova et al., 2008) are among the widely used approaches for solving Equation 2-3. Finally, point p can be estimated using the linear triangulation method used in (Hartley et al., 1992).

$$p_2^T \times F \times p_1 = 0$$ **Equation 2-2**

$$\hat{p}_2^T \times E \times \hat{p}_1 = 0$$ **Equation 2-3**

Reviewing the ongoing related research shows a lot of interests and demands for the articulated equipment pose tracking using CV-based methods. The current state-of-art methods open new challenges to apply such methods on real construction sites with reliable performance. This research considers the available object detection methods and stereo vision methods for detecting the excavator parts and estimating the excavator 3D pose.

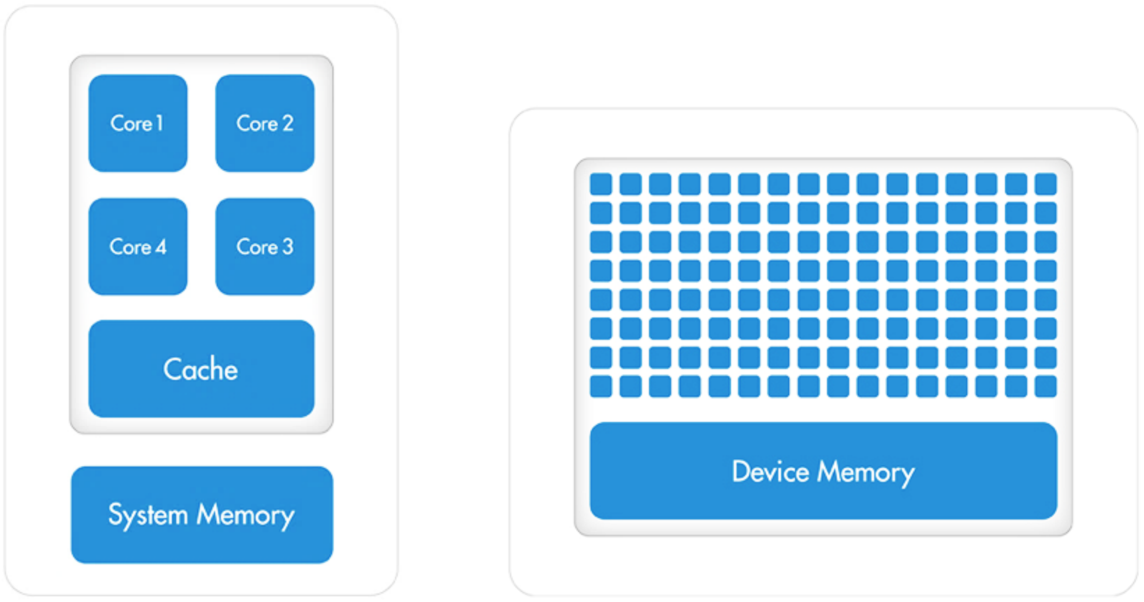
2.5.7 Applications of High Performance Computing in CV

National Institute for Computational Science (NICS) defined High Performance Computing (HPC) as the capabilities of *supercomputers* for solving the computational problems that are either too large for standard computers or would require a long processing time (NICS, 2011). These problems can come up from various research fields or industries including medicine, economics, engineering, entertainment, etc. One of the high-demand areas is CV and the image processing algorithms. Developing self-driving cars is an example application of HPC. A car driver observes the road while he/she is driving, and by recognizing a pedestrian who is crossing the road, the driver should estimate the required distance not to hit the person while predicting the future location of that person (Eagan, 2012). The whole process happens in real-time everyday within our brain. To model the mentioned process in the computer using the camera(s) or other sensors, the computer has to process the input data within millisecond(s) to avoid an accident. Therefore, there is a high demand of HPC when a process is supposed to be done in a very short time with the maximum accuracy. Basically, HPC can be developed on Personal Computers including desktops, laptops, and workstations, smartphones, supercomputers, grids, clouds, and/or any combination of these systems. The main point in HPC is the ability to handle the multiple functions efficiently and as quick as possible while receiving a large amount of data (Meneses, 2015).

In addition to the parallelism, the special hardware accelerators represent another HPC category that can perform a particular class of functions more efficiently. Graphical Processing Units (GPUs) are the popular example of the special-purpose hardware for increasing the performance of the image processing and CV processes. In simple words, GPUs are widely used for numerous simple mathematical and generic computations and deal with the large data while Central Processing Units (CPUs) are preferred to be used for less number but more complicated calculations.

As shown in Figure 2-6(a), CPUs take benefit of a more powerful processors unit, supported by the cache memory. A GPU consists of hundreds of small CPUs supported by their own memories (Figure 2-6(b)). Pulli et al. (2012) introduced the advantages of using GPU for running CV algorithms. Applying GPU can speed up the image processing and CV methods' processes from six to 30 times.

In the field of construction, Azar (2013) compared the performance of GPU versus CPU while applying HOG detectors on the video collected from the construction site and he achieved a significant improvement in terms of processing speed by using GPU. The results of this research confirmed the speedup of 25 times while using GPU during the object recognition process.



(a) CPU (Multiple Cores) (b) GPU (Hundreds of Cores)

Figure 2-6. Hardware structures of CPU and GPU (Reese & Zaranek, 2011)

2.6 Applications of CV in Construction Projects

Although there is no solid structure for categorizing the applications of CV in the construction industry, reviewing the studies in this area highlights three main categories: recognition of the construction resources, tracking their movements, and determining their current actions. In the following, each category is introduced and discussed to highlight the current research gaps and to narrow the scope of this research.

2.6.1 Object Recognition Methods for Monitoring the Construction Resources

As explained in Section 2.6.1, an object can be detected and categorized through its shape, color, and/or motion characteristics (Park & Brilakis, 2012a). The following studies take advantage of each of these characteristics (or a combination of them) to recognize the workers and/or equipment in construction projects.

Zou and Kim (2007) had an early attempt that studied the application of Hue, Saturation, and Value (HSV) color space to estimate the idle time of hydraulic excavators. Comparing Red, Green, and Blue (RGB) color space with HSV shows that the RGB color space is easy to understand, but it brings many difficulties to differentiate the object of interest from the background (especially when the light conditions change from bright to dark) since the histogram of RGB values of the target object usually overlap with the histogram of its background. On the other hand, the characteristics of hue change less by changing the light conditions in open fields since the HSV color space relies on the dominant wavelength of the perceived color. One of the assumptions in (Zou & Kim, 2007) is selecting the object of interest in the original image manually by the user, which limits the application of the proposed method.

Weerasinghe and Ruwanpura (2009) proposed detecting the workers through their hardhats. The algorithm searches for the hardhats with a known color (e.g. yellow). However, this method is not reliable for complex construction sites with a cluttered background.

Template matching is another way to detect the appearance of an entity (Cole et al., 2004), which can be applied using object eigen-image (Turk & Pentland, 1991; Monwar et al, 2007), an active shape or appearance model (Cootes et al., 1995; Cootes et al., 2001; Matthews & Baker, 2004), and a bag of words or textons (Julesz, 1981; Csurka et al., 2004; Shotton et l., 2008). Brilakis et al. (2011) used a Semantic Texton Forests (SFTs) method, which learns the appearance features of the object and context information (Shotton et al., 2008).

Another research done by Chi and Caldas (2011) introduces the background subtraction method developed first by Li et al. (2003) for detecting the construction resources. This method detects the objects by finding and removing the static pixels (background) out of the moving pixels (foreground). Knowing the foreground pixels can be used to find the moving object but it does not provide any further information about the object (whether it is a worker, a backhoe, or a truck). Therefore, the foreground pixels are passed through a pre-trained object classifier (i.e., a Bayes classifier or a neural network) and the classifier determines the type of the moving object. As a limitation, this method needs the object to be moving in the subsequent frames to be detected; but if the object is stationary, it is not recognized.

Haar-like features supported by the Adaptive Boosting algorithm (Viola & Jones, 2001; Freund & Schapire, 1997), HOG features coupled with a Support Vector Machine (SVM) developed by (Dalal & Triggs, 2005), color histogram (Swain & Ballard, 1991), and Eigen-images including color and shape information, can be adapted for construction resources' recognition (Park & Brilakis, 2012a). Park and Brilakis (2012b) proposed applying the background subtraction method coupled with Haar-cascade features. Moreover, they used HSV color space to minimize the false detections by considering the colors of the objects. They also investigated the combination of background subtraction method, HOG features with SVM classifier, and color histogram with *k*-NN (*k* Nearest Neighbors) in subsequent levels to recognize the worker at construction sites. However, both methods are limited to the moving objects.

Azar and McCabe (2012a) studied the recognition result by using HOG and Haar-Like features on the static images (Haar-HOG). They trained eight detectors to cover all views around the equipment. Moreover, they applied a Haar detector in combination with an HOG detector on the video of the site. In another scenario, the foreground object of the video was detected and then the HOG detector searches for the target object within the foreground pixels (Blob-HOG). The results show that Haar-HOG performs slightly better than Blob-HOG, while the Blob-HOG is less intensive in terms of computation.

Memarzadeh et al. (2013) proposed using HOG features in association with Histogram of Color (HOC) and their results show that HOG-HOC slightly outperformed HOG. In the research done by Tajeen and Zhu (2014), two methods previously developed by Torralba et al. (2004; 2007) and by Felzenszwalb et al. (2010) were compared in the construction environment. The idea that Torralba et al. (2004; 2007) presented is to use the shared patches of different classes through a machine learning process Figure 2-7. This concept helped to run the classifier faster while it requires less data to train since it uses the shared data across the classes.

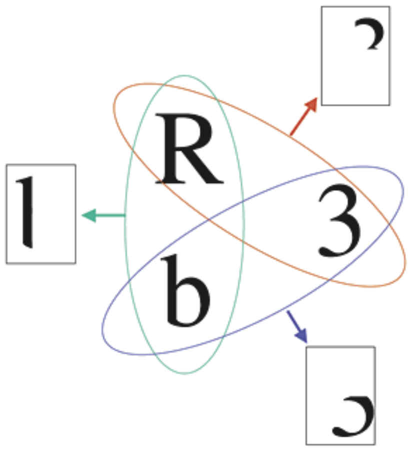


Figure 2-7. Example of shared objects' features (Torralba et al., 2007)

Felzenszwalb et al. (2010) proposed the discriminatively trained part-based model, which not only uses HOG features of the whole deformable object, but also considers HOG features of each attached subpart of that object. It also applies a latent SVM to formulate the relations between the features of the object and its parts. The comparison results provided by Tajeen and Zhu (2014) show that the part-based model performs better and is more robust to occlusions; while the sharing features method performs faster.

The investigation on the methods explained in this section shows that they suffer from the long annotation time in addition to the limited available views of the equipment.

2.6.2 Pose Estimation and Activity Recognition of Construction Resources

One of the methods for estimating the pose of articulated objects is using Time-of-Flight (ToF) camera integrated by Charged-Coupled Camera (Son & Kim, 2012); however, the ToF camera's operating range is limited to 7.5 *m* while much longer range is expected for applying on construction sites. Kashani et al. (Kashani et al., 2010) proposed the laser-based method for tracking the pose of shovel's dipper using the depth information received from the laser scanner. In addition to high the cost of deployment, the method is effective when the target object specified. In other word, for tracking more than one target object, it is necessary to identify each object and differentiate them from each other. Moreover, a model-based automatic dynamic object recognition and registration method was proposed by Cho and Gai (2013) using CAD models of the excavator parts. Although, the accuracy of the registration was from 27 to 34 *mm* but it still suffers from the high cost of the laser scanner deployment.

Lundeen et al. (2015; 2016) and Feng et al. (2015) proposed an optical marker-based method for estimating the pose of the excavators. They attached multiple markers on the surfaces of the equipment parts (Figure 2-8) and a stationary reference marker on the available permanent structure on the construction site. Recently, they claimed achieving an accuracy of 2.5 *cm* in their latest study (Lundeen, et al., 2016). This method can become practical after reducing the required size of the makers and addressing the challenges with dirt on the markers and stability of the markers.

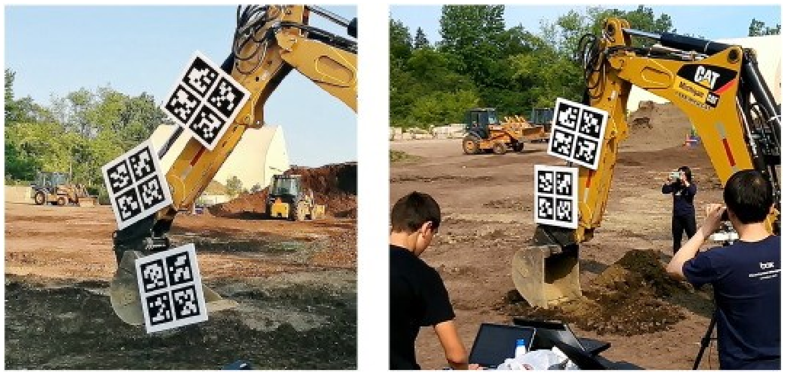


Figure 2-8. Markers installed on backhoe (Lundeen, et al., 2016)

Recently, Yuan et al. (2016) proposed using hybrid kinematic shapes and key nodes for tracking the pose of excavators. The joints of the excavator were detected using the template matching approach based on the edge detection algorithm. The method shows an error of about 0.5 *m* on the *x* and *y* axes and 1.0 *m* on the *z* axis when the target is 15 *m* far from the cameras. However, the results show that when the target was over 20 *m* far from the cameras, the error increased to over 6.0 *m*. Moreover, since the cameras are installed close to each other, there is a high risk that the angle between the excavator's side view and the cameras becomes close to 180˚, which results in self-occlusion of the excavator parts. These problems can be avoided by installing the stereo cameras several meters far from each other. Additionally, the method should be evaluated in scenarios where the cameras are installed high from the ground level since in most construction sites the cameras are located at the highest level to maximize the coverage.

Other alternatives for estimating the pose of the excavators are the commercial machine control or guidance systems. Basically, both systems are taking advantage of the sensory devices attached to the parts of an excavator. Trimble as one of the solution providers in this area claimed that their product can provide the angle of the parts with an error of up to 4% (Trimble, 2017). Although these systems can provide reliable results, attaching multiple sensors on each excavator is not only time consuming but also costly.

2.6.3 Applications of CV for Analysing Health and Safety on Construction Sites

Basically, the application of CV for the safety purposes can be studied from two aspects. The first context focuses on the workers and the methods for extracting the pose of human bodies. The other aspect focuses on the interactions between the moveable construction resources (e.g. worker and equipment) to avoid the hazardous situations.

According to the comparative study done by Han et al. (2014), the steps toward the recognition of unsafe human actions are: deriving the human body motion information, reducing the motion data dimension, creating the trajectory of the motions, and recognizing the determined action class. Moreover, the motion data are usually described in rotation angles, joint angles, position vectors, and movement directions. The results of their experiments show that the rotation angle performs better than the joint angle and the position vector.

Li and Lee (2011) developed a method to reconstruct the 3D human skeletons by obtaining 3D coordinates of each body joint from the multiple ordinary network cameras for the ergonomic analyses. Their approach elaborated on the motion identification, motion recognition, motion analysis, and motion visualization. Han et al. (2011) extended this research by applying kernel Principal Component Analysis with polynomial kernel and by using the supervised classification techniques to detect unsafe motions (Han et al., 2012).

In addition to the health related aspects, there are many studies focused on the applications of CV for the construction site's safety. Chi and Caldas (2011) proposed an image-based safety assessment approach, which elaborates on the automated spatial safety risk identification of the surface mining activities. In this research, the main earthmoving activities (e.g. loading, hauling, and dumping) and their related accidents were introduced. By investigating the causes of those accidents, all related risk factors for each accident were classified within three categories: operator error, poor operating condition, and mechanical/hydraulic failure. Reviewing the risk factors, safety regulations, and best practices showed that the data related to moving speed, proximity to dangerous areas, and proximity to other object and stopping distances are the fundamental spatial information required for developing a safe work area. Furthermore, selecting an accurate and reliable data collection device can directly affect the applicability of the proposed safety system. Chi and Caldas (2011) compared different data collection devices and the results are shown in Table 2-1. The table clearly shows that the stereo vision camera is more reliable than the other devices because of its high frame rate and long reading range.

Table 2-1. Comparison of data collection devices (Chi & Caldas, 2011)

Devices	**Frame rate**	**Outdoor application**	**Maximum reading range**	**Object localization**	**3D modeling**
LADAR	Slow (< 1 Hz)	Yes	Very long (> 100 m)	Yes	Yes
Flash LADAR	Fast (> 10 Hz)	No	Short (< 10 m)	Yes	Yes
Video camera	Fast (> 10 Hz)	Yes	Long (>50 m)	No	No
Stereo vision camera	Fast (> 10 Hz)	Yes	Long (>50 m)	Yes	Yes

Moreover, it is able to provide the 3D coordinates of the localized object in the outdoor environment. The overview of their proposed object recognition and tracking framework is shown in Figure 2-9. By receiving the real-time video from the construction site, the background subtraction algorithm detects the moving objects and the pixels involved for each object are formed together as one segmented region. Moreover, the corresponding RGB pixels of the segmented regions are passed through the object classification module to classify the foreground segment.

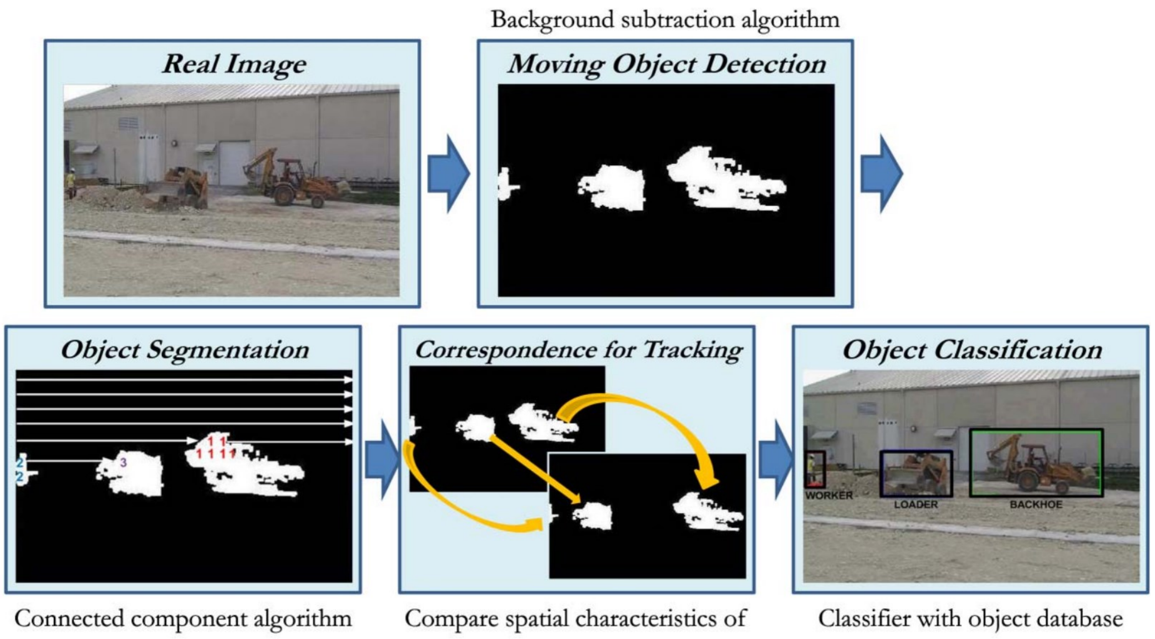

Background subtraction algorithm
Real Image
Moving Object Detection
Object Segmentation
Correspondence for Tracking
Object Classification
WORKER
LOADER
BACKHOE
Connected component algorithm
Compare spatial characteristics of
Classifier with object database

Figure 2-9. Overview of object identification and tracking process (Chi & Caldas, 2011)

The proposed framework aimed to control the speed limit, dangerous access, and close proximity violations by knowing the location of the resources and applying a set of safety rules achieved from the regulation and interviews with the expert. As a future opportunity, they suggested the integration of GPS or UWB system to the current image-based method to improve the object identification and tracking results. Moreover, an optimized camera allocation plan with proper cameras' location were recommended for the future research.

Ferrer et al. (2013) investigated the application of high-speed cameras (500 fps) and image processing methods for safety assessment on construction sites. In this research, they applied morphological image filtering and Hough transform to follow the falling objects on the sites.

Recently, Seo et al. (2015) reviewed the current status of CV techniques for construction safety and health monitoring. They studied the related research under three categories: object recognition, object tracking, and action recognition. As shown in Figure 2-10, the data can be provided by 2D or 3D images using surveillance cameras, portable cameras, Flash LADAR, RGD-D sensors, or stereo cameras. The safety and health risk identification can be investigated based on scenes, locations and actions by analyzing the collected data.

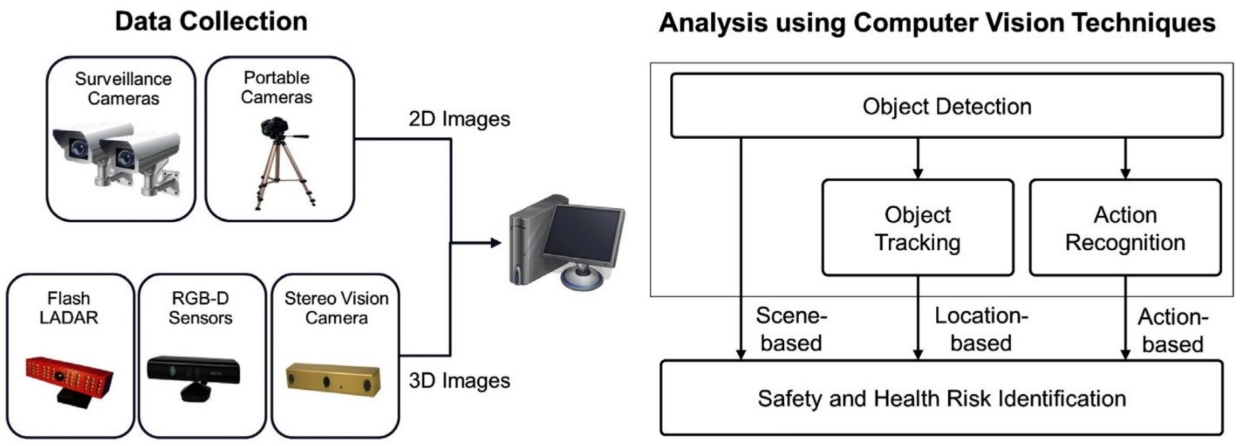


Figure 2-10. General framework for CV-based safety and health monitoring (Seo et al., 2015)

2.6.4 Applications of Stereo Vision in Construction Projects

Extracting information through a single RGB camera does not provide enough insight about the real scene since the 3D information are projected on a 2D image plane and some information is lost during this transition. Therefore, when there is a need to have access to 3D information of the scene, applying stereo imaging techniques using the 2D views from two or more cameras becomes necessary. Park et al. (2011) investigated the performance of 3D location estimation of construction workers and a vehicle. In the first configuration, two video cameras were installed at the distance of 3.8 *m* and in the second configuration they were installed at the distance of 8.3 *m* from each other. The workers and the vehicle passed in front of the cameras mostly perpendicular to the cameras focal axis within the distance of approximately 30-40 *m* from the cameras' baseline. A moving checkerboard was used for camera calibration and the board contained 7 by 9 blocks of 65 × 65 *mm*. The average localization error of 0.32 *m* and maximum error of 1.51 *m* were achieved for the 8.3 *m* baseline and the average error of 0.62 *m* and maximum error of 2.55 *m* were the result of 3.8 *m* baseline. Comparing the results shows that using the long baseline outperformed the configuration of shorter baseline, which proves the statement of Okutomi and Kanade (1993) explained in Section 6.1. Similar to Park et al. (2011), Cordova and Brilakis (2008) proposed a 3D vision system for tracking the construction personnel using two cameras. Recently, Zhu et al. (2016) designed a Kalman filter and applied on the location data achieved by Park et al. (2011). The results after applying the filter shows an improvement of 50%.

2.7 Multi-Sensor Data Fusion Applications in Construction Projects

Integration of data from the multiple sensors to improve the reliability and the accuracy of the final results is known as MSDF. Missing data and noisy data may significantly drop the accuracy of a system. Sensor data fusion can overcome these limitations by using the data collected from the different sensors. In case of missing data, the system is able to complement the missing data from one sensor, which failed from acquiring data at a certain point of time, by similar target data received from other sensors at the same time (Luo et al., 2002). In another scenario, all data collected at the certain time may be compared to converge the output toward a more reliable result.

Focusing on the location and the pose data, the fusion process can be summarized in four phases: (1) data alignment toward a common coordinates system, (2) data association using a set of rules to relate the data measured from the different sensors, (3) position estimation based on the associated data, and (4) identity estimation, which finds the sources of fed measurements for the investigated estimation (Smith & Singh, 2006; Hall, 1992).

During the last few years, many researchers investigated the applications and the benefits of MSDF on construction processes. Luo et al. (2013) studied the effect of the location-aware sensor data imperfections on the jobsite safety monitoring. Moreover, they proposed data fusion from various sources to improve the precision and recall rates.

Shahi (2012) and Shahi et al. (2014) proposed a multi-source data fusion system for the automated progress tracking of construction activities. The proposed system, which mainly focused on the activities' tracking instead of objects' tracking, integrated volumetric and positioning data with the project control information (e.g. foreman reports, schedule information, and other information sources). The results of their experiments show that their approach can improve the existing object tracking and recognition algorithms. However, the validation of the method was limited to piping activities in an industrial building project and it should to be tested on a full scale project including various activities.

El-Omari & Moselhi (2011) developed an Information Technology platform to acquire data from various technologies set on construction sites for the progress measurement purposes. The platform includes bar coding, RFID, 3D laser scanning, photogrammetry, multimedia, and pen-based computers. In addition to the automated data acquisition technologies, a planning and scheduling software system, a relational database, and AutoCAD were added to the proposed platform to assist project management teams for the decision making.

Razavi & Haas (2010) studied a modified functional data fusion model for an on-site materials tracking on the construction yard. This hybrid solution was used for automated identification, location estimation, and dislocation detection of the construction materials. They used various physical sensors, different location estimation algorithms, location contexts from the automated data collection technologies (Received Signal Strength Indicator, Positional Dilution of Precision), time and BIM (site map/layout and 3D models) as the data sources. Moreover, Dempster-Shafer theory applied in their model was found promising to the materials dislocation detection.

An automated construction activity monitoring platform was introduced by Rebolj et al. (2008) integrating image recognition-based tracking, BIM-based material tracking and the Dynamic Communication Environment (DyCE) supported by mobile computing. However, the system requires to set multiple cameras to solve the problems of obstructed elements and BIM technology has to be used to its full extent.

Rafiee et al. (2013) proposed the MSDF method for fusing data from BIM, UWB, and surveillance cameras to improve the security for indoor environments. The method estimated the locations of people in the video and compared them with the locations provided by UWB system to detect the intruders. Dibitonto et al. (2011) introduced a fusion method for tracking people. They integrated radio-based and video-based localization systems to increase the accuracy and reliability of location estimation results.

2.8 **Applications of Serious Games in Construction**

Serious games use video game elements for purposes other than pure entertainment (Djaouti et al., 2011). Serious games attempt to create instructionally sound and relevant learning experiences for a wide variety of industries such as defense, education, healthcare, emergency management, city planning, engineering, etc. (Charsky, 2010).

Different studies were done to explore the potential of serious games within the construction industry using game engines and virtual environments. These studies mainly focused on training or evaluating the architectural design of the building (Deshpande & Huang, 2011; Oerter et al, 2013). Al-Jinouri et al. (2005) investigated a simulation model, which was developed in the form of a management game as an alternative way for teaching construction planning and control. Xie et al. (2006) proposed the creation of a Virtual Reality (VR) safety-training system to assess the perceptual and behavioral impacts of the VR environments on the trainees. This system, which was reconfigurable and reusable, made it possible to create 3D virtual images and produced memorable experiences for the trainees. Additionally, the virtual environment factors, such as temperature, air composition, and visibility, were also studied and simulated in their system. Zhao et al. (2009) studied the issues concerning the electrical-related accidents within the construction industry. They discussed the benefits of using an active training approach such as VR simulation and its effects on the cognitive abilities of the users.

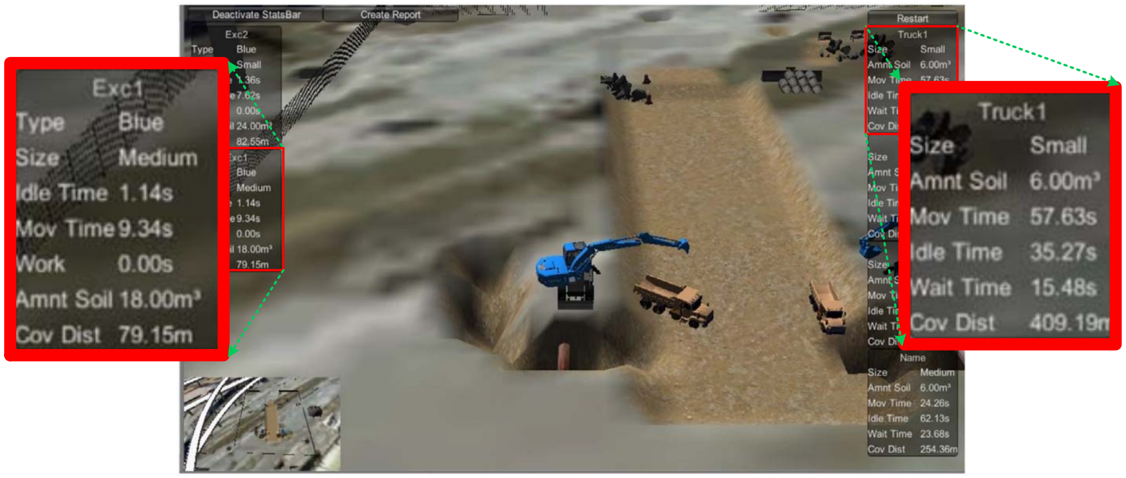
A Safety Inspector 3D video game was proposed by Lin et al. (2011) to provide a comprehensive safety training environment in which the user can understand the safety roles and walk through the game site to identify the potential hazards. A small group of students were asked to use the preliminary game system to evaluate the effectiveness of the designed system.

Under the research project of "*Serious Human Rescue Game*" at Technische Universität Darmstadt, Rüppel & Schatz (2011) proposed a serious gaming approach using BIM for the exploration of the effect of building conditions on the human behavior during the evacuation process since it is impossible to conduct the rescue tests during a real fire.

Ku and Mahabaleshwarkar (2011) described the concept of Building interactive Modeling, which provided the capability of interaction with BIM, to enhance the collaborative information and knowledge sharing. Goedert et al. (2011) proposed a framework for a Virtual Interactive Construction Education system named VICE, taking advantage of the new technologies in simulation, modeling, and semantic web and software engineering.

Amr and Mohamed (2011) proposed the utilization of video game technologies for construction operation simulation visualization; however their main focus was on the managerial level of the projects and it was limited to the visualization capability of the game engines. Furthermore, an approach for integrating the workspace management within the planning process using a serious game was proposed by Chavada et al. (2012).

Hammad et al. (2014) proposed using the game engine for integrating the macro and micro path planning methods for monitoring the safety of excavation operations. Langari and Hammad (2015) extended the study of Hammad et al. (2014) by developing a new algorithm for the path planningof the excavators. This study attempted to improve the safety and productivity by providing a higher level of support to the excavator operators. Vahdatikhaki et al. (2017) proposed a Multi-Agent System (MAS) for enhancing the coordination and safety of the earthwork operations using the game environment. Figure 2-11 shows Unity implementation of MAS and represents the captured information from Operator Agents (OAs) to be used by Team Coordinator Agents (TCAs). This system is able to provide the productivity related information such as the equipment idle time, moving time, waiting time, and the soil volume moved by the equipment.



Exc1
Type Blue
Size Medium
Idle Time 1.14s
Mov Time 9.34s
Work 0.00s
Amnt Soil 18.00m³
Cov Dist 79.15m

Truck1
Size Small
Amnt Soil 6.00m³
Mov Time 57.63s
Idle Time 35.27s
Wait Time 15.48s
Cov Dist 409.19m

Figure 2-11. Unity implementation representing typical information OAs report to TCAs (adapted from Vahdatikhaki et al., 2017)

From the safety point of view, this system applies LAEWs for collision-free path planning. The system generates the risk map of the excavator with higher assigned priority and LAEW is created for the adjacent equipment. In the example shown in Figure 2-12, Excavator 1 has higher priority; therefore the LAEW of Excavator 2 is generated relative to the risk map of Excavator 1.

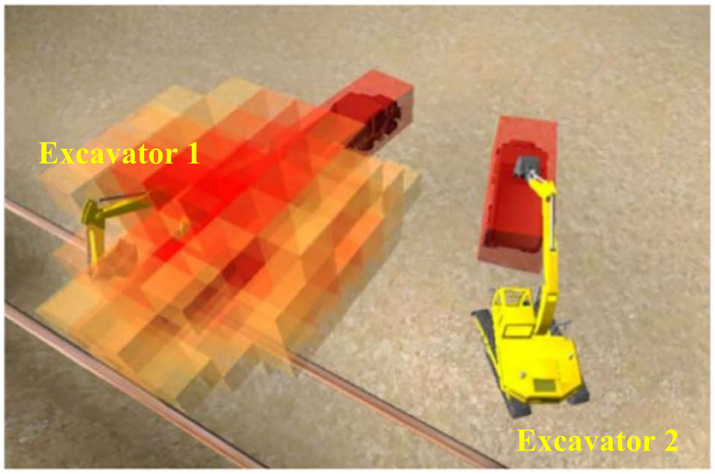


Figure 2-12. LAEW of Excavator 2 (adapted from Vahdatikhaki et al., 2017)

Concluding of applications of the serious games in construction shows that the games were mainly used for the safety training practices in construction industry. However, the recent study of Vahdatikhaki et al. (2017) opened a new chapter for applying MAS on the earthwork projects using the serious game. This would allow the other researchers to collect NRT data from the construction sites and feed them into the game environment for identifying the equipment states and detecting the potential collisions.

2.9 **Summary**

In this chapter, the current status of the safety on the construction sites was studied and the statistics showed the demand of more investigations in this area. Moreover, the collisions between the construction resources including both equipment and humans were found as high risk events on the construction sites. Moreover, the available localization and RTLS technologies were studied to select the most beneficial technologies for this research. CV-based systems were considered as the main technology for supporting the proposed method because of its ease of access and availability in addition to the high accuracy and reliability. The second selected technology was the RF-based RTLS system because of the wide coverage range while it is less dependent on having the line of sight to the tracking objects. Both GPS and UWB have their own benefits and capabilities that the user can select one of them or even both of them based on the scope and the environment of the project.

Furthermore, the general methods and applications of CV were studied and explained. The current methods for the object recognition and tracking, and image processing and segmentations were introduced. Additionally, the current applications of CV to be used on construction site were investigated for improving the safety. However, most of the previous CV-based methods in construction are dependent on the motion of the equipment. Knowing that the excavator pose estimation is limited to the marker-based methods, there is no available image archive for training the excavator parts' detectors. Moreover, the applications of MSDF were discussed as a potential area for supporting the current research considering the high requirements for real-time and accurate pose estimation in safety monitoring. In addition, the current configuration of the stereo cameras are not designed for the large construction sites.

Finally, the current applications of serious games on construction sites were introduced and the potentials of the game engines and LAEW were acknowledged. These kind of applications are effective in detecting potential collisions in the scope of safety monitoring.

CHAPTER 3 RESEARCH FRAMEWORK

3.1 Introduction

In Chapter 2, the fatality rate of work-related accidents in construction projects was reviewed. The statistics showed the importance of the active measurements of proximity among equipment and workers on construction sites. Avoiding collisions between construction resources requires NRT monitoring of the equipment and the workers' movements. For equipment with articulated parts (e.g. excavators), the scenario is more complicated since the NRT poses of the parts have to be monitored in addition to the equipment itself. The 3D pose information is used to detect the potential collisions through the methods explained in Section 2.2. Moreover, the pose information can be used for estimating the productivity. However, estimating the productivity is out of scope of this research and it remains as future work.

3.2 Overview of the Research Framework

Reviewing the available technologies shows that depending on one single technology may not result in an effective and accurate performance. For example, using GPS, UWB, Bluetooth, Active RFID, or CV can provide the approximate location of the equipment. However, calculating the NRT pose of the equipment using a single source brings a large error, which is not reliable enough for safety purposes.

Figure 3-1 schematically shows the proposed framework with the following steps: (a) construction site configuration, (b) auto-annotation of the synthetic images, (c) data fusion and the excavator parts recognition, and (d) skeleton extraction and 3D pose estimation.

As shown in Figure 3-1(a), two main technologies, that are suggested to be used in this research, are RTLS and CV. Within the RTLS category, the users can choose GPS tracker, UWB, Bluetooth, or active RFID which are four types of RTLS that can provide the location of the equipment as the initial step toward pose tracking. In addition to the RTLS system, at least two cameras are required with a reasonable distance from each other, with a full coverage of the target area and full overlapping views. Steps (b), (c), and (d) are explained in the following sections.

3.3 Auto Annotation of Synthetic Images

Since currently there is no comprehensive archive for the equipment parts' images available, it is proposed to generate the images of these parts from the 3D virtual model of the equipment. Investigating the potential of using a 3D model of the construction equipment (e.g. excavators) instead of the images of the equipment to automatically generate annotations for object recognition training is the first priority of this research. As shown in Figure 3-1(b), the 3D model can be automatically integrated with different backgrounds of the construction sites to improve the training quality. Moreover, these images need to be annotated for the training of the detectors. Therefore, an auto-annotation algorithm is proposed to accurately annotate the parts within the synthetic image. This method can save time by not involving the human efforts during the annotation process. The resulting high quality training with the well-structured annotation could significantly improve the object recognition. Moreover, this method helps the industry to have access to numerous image datasets for different equipment from different manufactures and to customize the object detectors for each target object.

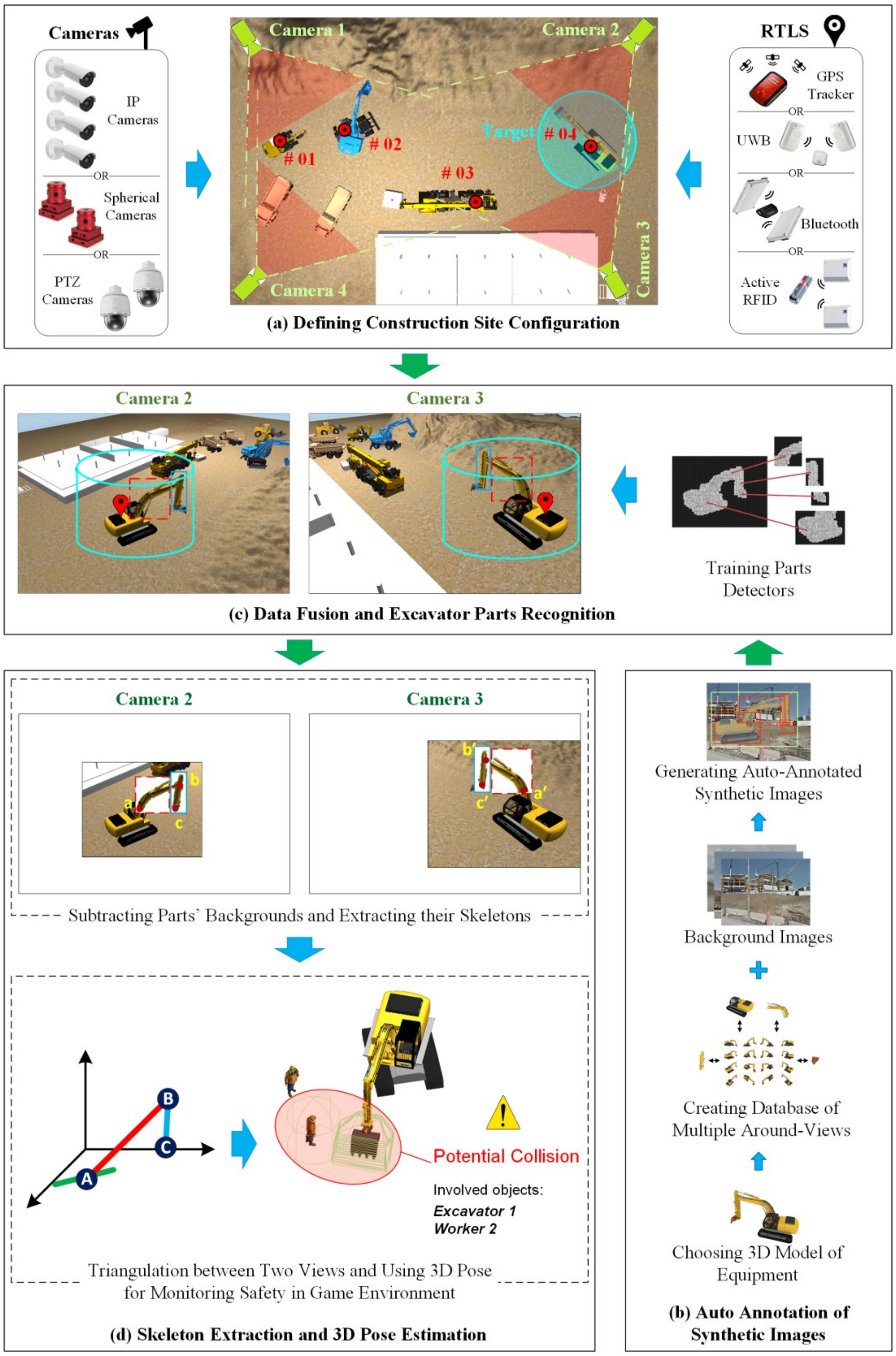

Cameras
IP Cameras
OR
Spherical Cameras
OR
PTZ Cameras
Camera 1
Camera 2
Camera 3
Camera 4
# 01
# 02
# 03
Target
# 04
RTLS
GPS Tracker
OR
UWB
OR
Bluetooth
OR
Active RFID
(a) Defining Construction Site Configuration
Camera 2
Camera 3
Training Parts Detectors
(c) Data Fusion and Excavator Parts Recognition
Camera 2
Camera 3
a
b
c
a'
b'
c'
Subtracting Parts' Backgrounds and Extracting their Skeletons
A
B
C
Potential Collision
Involved objects:
Excavator 1
Worker 2
Triangulation between Two Views and Using 3D Pose for Monitoring Safety in Game Environment
(d) Skeleton Extraction and 3D Pose Estimation
Generating Auto-Annotated Synthetic Images
Background Images
Creating Database of Multiple Around-Views
Choosing 3D Model of Equipment
(b) Auto Annotation of Synthetic Images

Figure 3-1. Schematic demonstration of the proposed framework

In addition, collecting an effective negative dataset is another challenge for training a detector model. The importance of negative samples is not less than positive samples since a robust detector not only needs to detect the target object correctly but also it has to reject the false detection. Generating negative samples related to construction projects can be automated by randomly cropping images taken from construction sites. Moreover, HPC can greatly help the proposed method to reduce the processing time for the auto-annotation and the negative sample generation by dividing the tasks between the multiple CPUs. After all, the knowledge of generating and annotating the synthetic images from 3D model is transferred to the next section for training the excavator parts' detectors. The details of the proposed method will be explained in Chapter 4.

3.4 Data Fusion and Excavator Parts Recognition

As mentioned in Section 1.3, one of the main objectives of this research is to estimate the NRT pose of the construction equipment. The proposed pose estimation method starts with collecting the real-time video data and RTLS data from the construction site after calibrating the on-site cameras. The RTLS data are linked with the corresponding video frames to fuse the location data with the video data. Knowing the overall size of the equipment and its location provided by the RTLS system, the cameras that are covering the equipment during its operation are selected. A 3D virtual cylinder is added at the center location of the target excavator in each camera's view based on the dimensions of the excavator (Figure 3-1(c)). Consequently, the video frames collected by the cameras are processed and a rectangle bounding box is cropped around the projected cylinder in each view as shown in Figure 3-1(c). Cropping the frames limits the search scope of the object detection algorithm resulting in a lower computation load and faster processing time while achieving more accurate results. Using the method explained in Section 3.3, the synthetic images of each equipment's part (e.g. dipper, boom, and body) are generated and annotated automatically. The multiple detectors from different views are trained for each part and then applied on the cropped frames to recognize the parts. The information of the detected parts is provided to the skeleton extraction methods explained in Section 3.5. Moreover, Chapter 5 will go through the details of the proposed data fusion and parts recognition methods.

3.5 Skeleton Extraction and 3D Pose Estimation

The background of each detected bounding box is removed and the pixel-wise location of each part is estimated. Then, knowing the location of each part and the kinematic relationships of the excavator's parts, the 2D skeleton of the excavator is extracted from each camera's view. The 3D pose of the equipment is calculated using the skeletons information of the excavator and the intrinsic and extrinsic parameters of the covering cameras.

The ultimate target of this research is to provide the 3D pose of the excavator using the proposed framework to the collision avoidance systems proposed by Vahdatikhaki et al. (2017) for avoiding the potential collisions. These systems were designed in a way to take advantage of a game environment. In the final phase, the NRT 3D pose of the excavator is sent to the game environment (shown in Figure 3-1(d)) and a collision avoidance system is used to detect any potential collisions and generate a warning. The steps required for supporting the skeleton extract and 3D pose estimation methods are further explained in Chapter 6.

3.6 **Summary**

In this chapter, the overview of the proposed framework was presented. The required technologies including CV and RTLS were briefly explained. Moreover, the configurations of the technologies and the methods to support the framework were discussed. In addition to the configuration of the proposed framework in Figure 3-1(a), three modules were defined toward enabling the ready-to-use data for monitoring safety of the excavators. In Chapters 4 to 6, the methods proposed in each module are explained in detail. The methods are developed and validated using real-world case studies.

CHAPTER 4 AUTO ANNOTATION OF SYNTHETIC IMAGES

4.1 Introduction

As explained in Section 2.5.2, there are several limitations of using the current annotation tools, which indirectly cause a low accuracy in object recognition. For instance, since the required time and efforts for preparing a rich positive dataset are high, the users may rely on a smaller dataset, which can result in lower accuracy of the object detection compared to a detector trained by a very larger dataset. The main difficulty for the annotation is the long time required for this task. Moreover, the required time is dramatically increased by adding more samples from different views, which represent different occlusion levels and light conditions. Additionally, training a model for each object's type within a category (e.g. if the object is an excavator, it can be manufactured by Caterpillar, Volvo, Komatsu, etc.) can make the situation more complicated. As explained in Section 3.3 and shown in Figure 3-1(b), generating the synthetic images from the 3D models and annotating them automatically are the proposed solutions in this chapter to address the afore-mentioned problems.

This chapter aims to achieve the following objectives: (1) to automatically generate equipment synthetic images using the 3D model of the equipment and real images of the construction sites as background to automatically annotate these images; (2) to consider different sizes and illumination conditions for the target object; (3) to automatically annotate the generated images of the 3D model; (4) to generate effective negative samples for training the detectors; (5) to evaluate the results of object recognition using the proposed method and to compare them with the results achieved using manual annotation of the real objects; and (6) to apply sensitivity analyses to determine the effect of the number of step angles for each detector and the effect of the Covered Range of Angles View (CRAV) within each detector.

4.2 Developing Auto Annotation Tool

Training many samples to cover all types in a systematic way may be time consuming; but on the other hand, it provides a specific descriptor for each type. This can be useful when the information about the specific type of the equipment in the scene is available. Figure 4-1 conceptually shows the data set for training an excavator descriptor. Various images are taken from many construction sites where different types of excavators are used. Additionally, they are from different positions and angles.

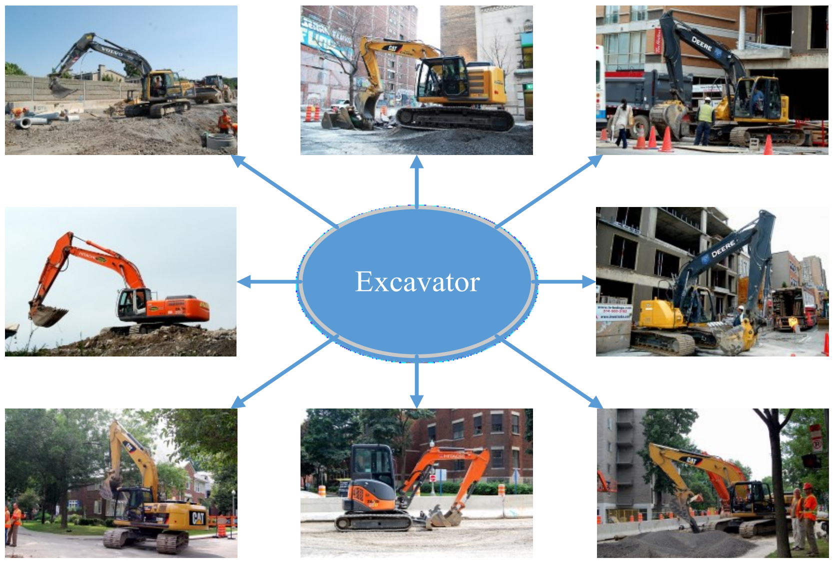


Figure 4-1. Schematic dataset for a tradition detector model

As shown in Figure 4-2, the proposed framework is divided into three phases: preprocessing, processing, and post-processing. Preprocessing starts by choosing the desired equipment and then the type. A 3D model of that specific equipment is assumed to be available and can be imported into a 3D modeling tool (e.g. 3Ds Max (Autodesk, 2015) or Google Sketchup (Trimble Navigation, 2013)). Moving to the processing phase, a virtual camera is moved around the 3D model and images of the object are captured (Figure 4-3).

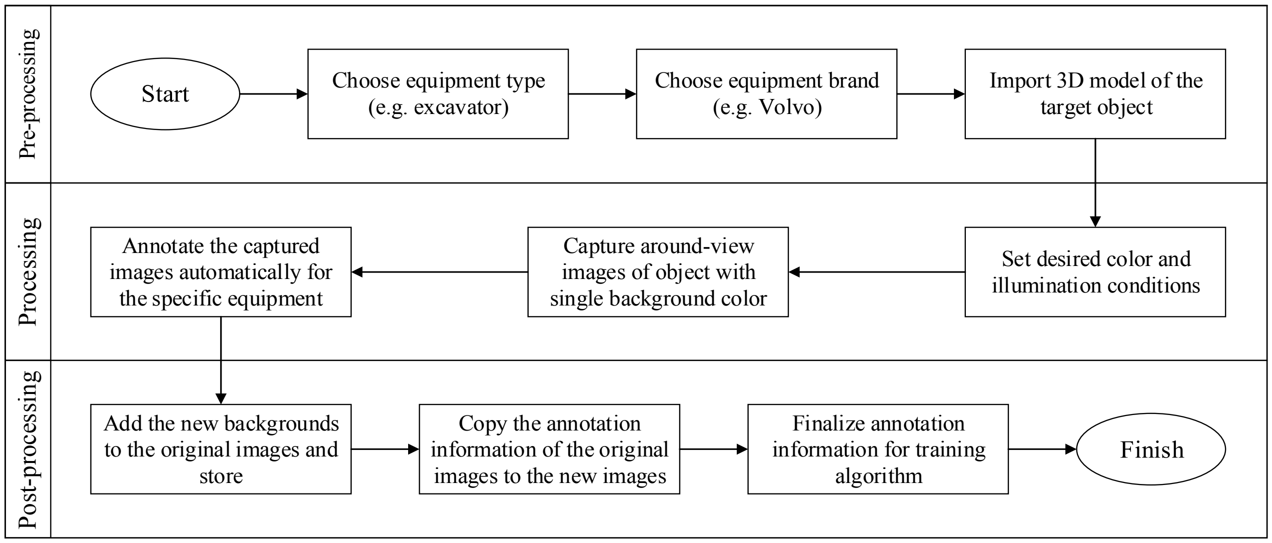

Pre-processing
Start
Choose equipment type (e.g. excavator)
Choose equipment brand (e.g. Volvo)
Import 3D model of the target object
Processing
Set desired color and illumination conditions
Capture around-view images of object with single background color
Annotate the captured images automatically for the specific equipment
Post-processing
Add the new backgrounds to the original images and store
Copy the annotation information of the original images to the new images
Finalize annotation information for training algorithm
Finish

Figure 4-2. Framework of the proposed method

This camera can take around-views images at different angles from the horizon. Since the data set is generated within the virtual environment, it is possible to include images from the same view but with different light conditions, visibility, and backgrounds. In the post-processing, the image of the construction site can be extracted from any construction image archive or from Google Earth and can be used as the background for the object's images.

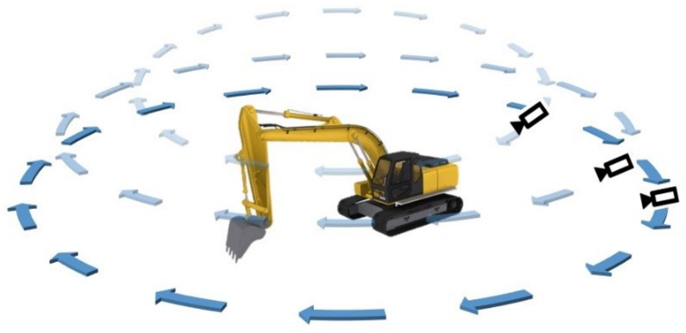


Figure 4-3. Spherical position of the camera

The captured images are transferred to the CV environment with the programming tools, such as Matlab (Mathworks, 2016) or OpenCV (Bradski., 2000), for further analysis and annotation. The images will be annotated through the method explained in the following. As shown in Figure 4-4(a), the captured image of an excavator in the virtual environment is used as foreground of the image of a construction site. Any background can be added to the foreground model of equipment to provide more images as shown in Figure 4-4(b). The same annotation information used for the raw images with the single background can be applied to the new synthetic images (Figure 4-4(c)).

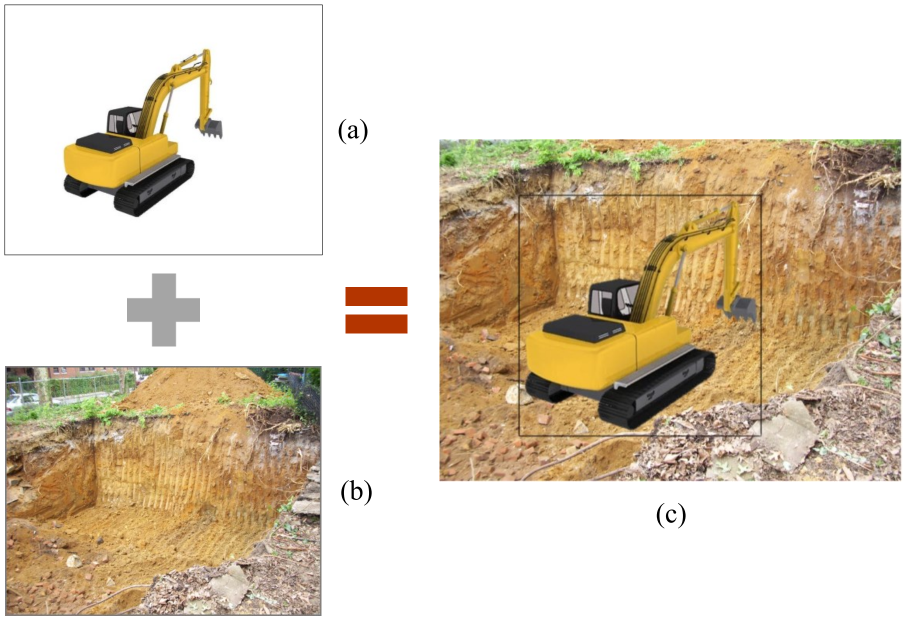


Figure 4-4. Creating synthetic image of construction equipment (adapted from Calfayan Construction, 2010)

4.2.1 Segmentation and Annotation of Images

Basically, in the current research, the 3D model renderer and the developed auto-image annotation tool work independently. In other words, the images of the 3D model of the equipment are first generated outside of the proposed toolbox using the available rendering software in the market. Since it is assumed that there is no previous knowledge about the color code and the pixels' uniformity of the single color background of the created images, these issues are considered in the development of the proposed toolbox. When the raw images coming from the 3D modeling tool with single color background are ready (Figure 4-5(a)), the annotation process starts by applying the image segmentation to recognize and separate the equipment from the single-color background. As shown in Figure 4-5(c), the edges of the target object can be calculated by applying Sobel-filter to the converted gray image (Figure 4-5(b)) from the original RGB image. Applying Dilate-filter reduces the black gaps between the surrounding edges of the object as shown in Figure 4-5(d). An algorithm presented by Soille (1999) is applied to fill the void area within the surrounding edge (Figure 4-5(e)).

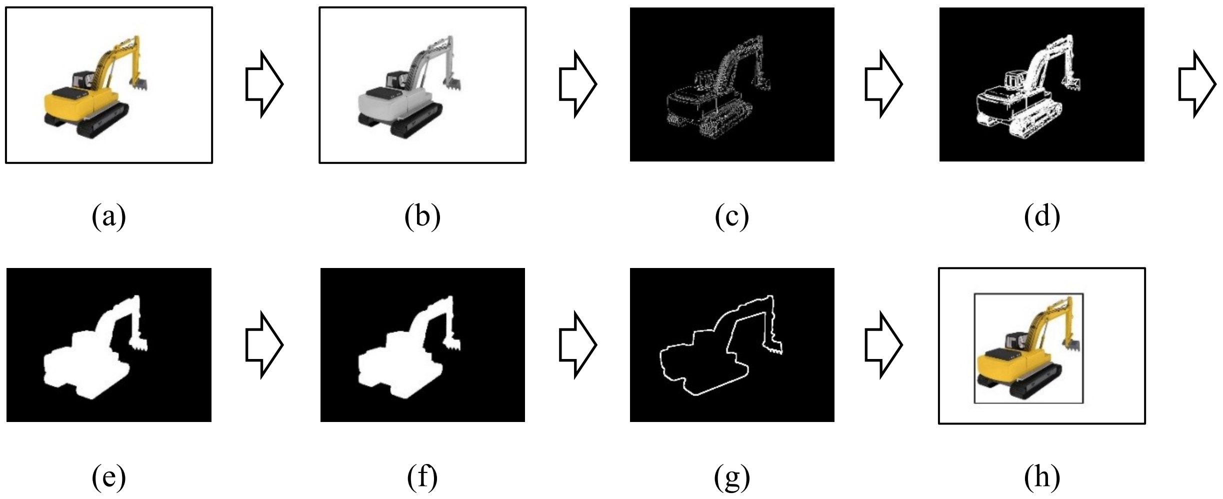
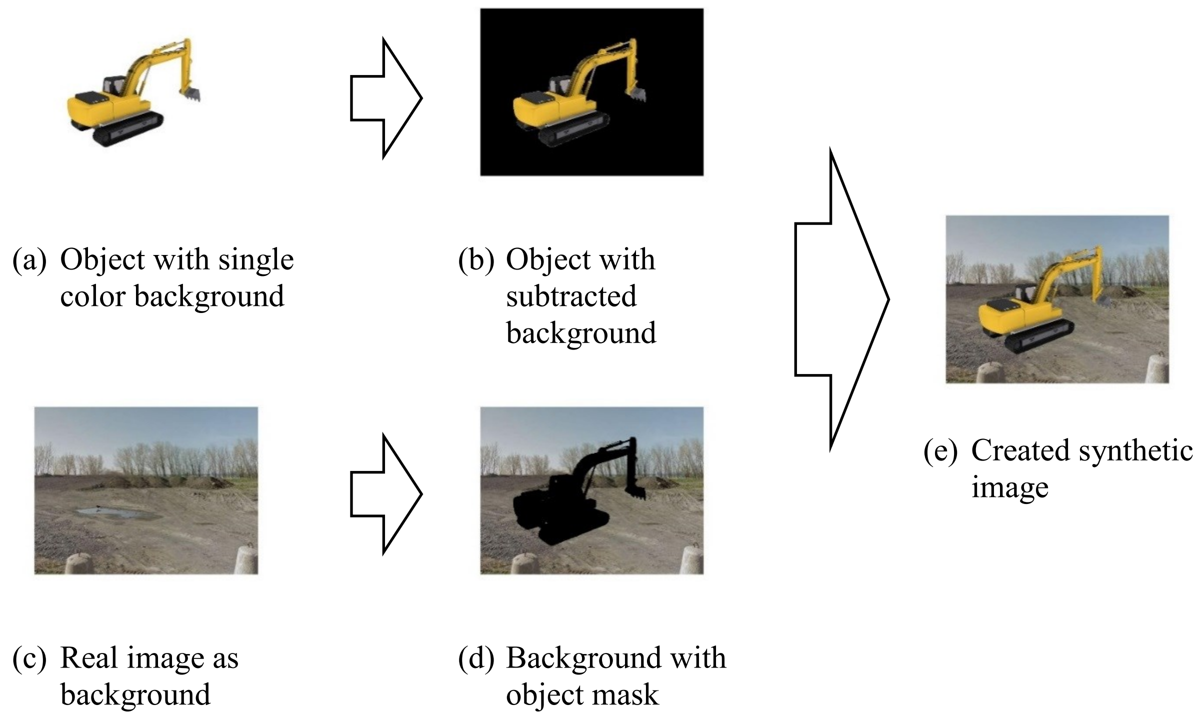


Figure 4-5. Segmenting and annotating the target object

Those groups of pixels, which are lighter than their surrounding are connected to the image border (Mathworks, 2014b; Soille, 1999). Furthermore, the area of the remaining groups of pixels other than the image background is sorted and the cluster with maximum area is chosen as the target object (Figure 4-5(f)). Figure 4-5(g) shows the calculated border of the object. A rectangular box plus a margin of *m* (should be defined by the user) is applied to surround the object as shown in Figure 4-5(h).

4.2.2 Adding the Background

The assumption for adding the background image is that both the image of the object and the image of the background have the same size. However, in case the sizes are not the same, the initial image containing the desired object can be scaled to fit in the background image. First, the single color background of the training data set (Figure 4-6(a)) should be removed. This task can be done by applying a similar approach to the one used in the previous image segmentation part resulting in Figure 4-6(b). In the next step, the corresponding pixels of the target object in the background image (Figure 4-6(c)) are changed to zero as shown in Figure 4-6(d).



(a) Object with single color background
(b) Object with subtracted background
(c) Real image as background
(d) Background with object mask
(e) Created synthetic image

Figure 4-6. Process of generating synthetic images

The final image shown in Figure 4-6(e) is achieved by combining the object image without background and the new background with a void area of the target object. Since the position of the object is not changed from the original image to the image with the new background, the same bounding-box can be considered for the new image to cover the object within the image.

4.2.3 Generating Large Positive and Negative Datasets

Training the positive samples with different sizes and qualities can improve the true positive rate of the detectors. Therefore, a pyramid structure is proposed to reduce the size of the object in the synthetic image by 50% in each step. The number of steps is defined based on the user preference. In this process, the determined bounding box is applied to the related original size of the image of the object with a single background color. The size reduction is applied only on the content of the original bounding box. The reduced box is added to a single background color with the same dimension of the original image.

Another possibility to improve the accuracy of the proposed method is to consider various illuminations, contrasts, and sharpness levels of the object while creating the synthetic images. For each feature, a wide range of values can be applied. For instance, Figure 4-7 shows three steps of illumination reduction applied on the original image. A similar function can be applied for contrast and sharpness levels. The process of changing the values of appearance variables of the images is developed before the process of adding the background. The appearance of object without background (Figure 4-7(a)) is revised based on the new illumination, contrast, and sharpness levels and the rest of the process for adding the background is followed as before.

(a) Original illumination (b) First reduction (c) Second reduction (d) Third reduction

Figure 4-7. Illumination changes

Improving the quality of positive samples has a great effect on the accuracy of the detector using these samples. Moreover, choosing the appropriate negative samples can also help the detector to reject the wrong detections. The negative samples can be parts of background images of the construction sites or any object other than the target object for detection. Although this process looks simple, the lack of related negative sample scan lead the detector to choose totally unrelated negative samples (e.g. a flower as negative sample for construction site). In this research, an auto-generation method is used to create many negative samples with random sizes, by providing a small number of images from the desired construction site excluding the target object (Figure 4-8).

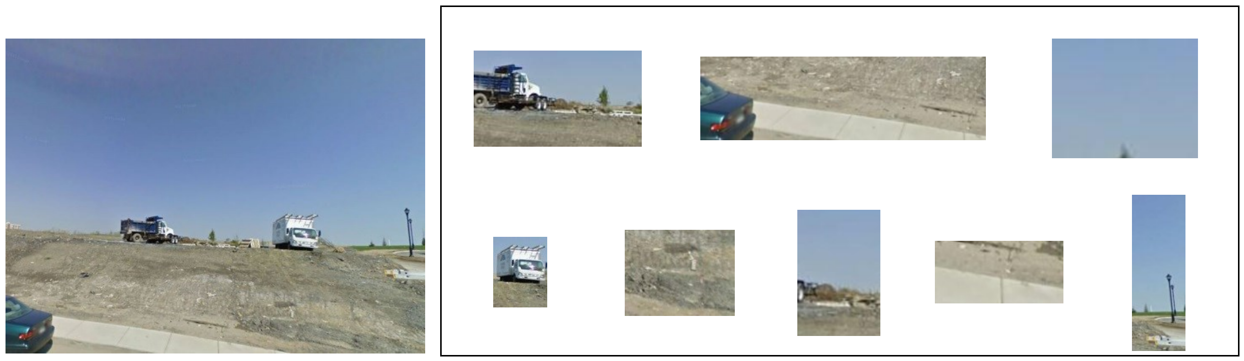

(a) Original Image (b) Randomly generated negative samples from original image

Figure 4-8. Randomly generated negative sample

The negative image sampler receives multiple images as input, which do not include the target object. The number of required negative samples, and the minimum and maximum sizes of each sample are defined by the user. The algorithm starts cropping rectangle boxes randomly out of each input image while satisfying the user-defined sizes.

4.3 Implementation and Case Study

Two case studies are provided to evaluate the proposed method. The first case study compares the detection results using real images for training the detector and using synthetic images with and without the images of the construction site as background. Moreover, the available image annotation in Matlab is used to annotate the real images manually and the required time is recorded using a stop watch (the Matlab codes are provided in Appendix A, Appendix B, Appendix C, and Appendix D). The time for manual annotation is recorded from the point that the user starts to annotate the first image and ends by finishing the last image. The second case study checks the effect of the CRAV within each detector and the number of step angles for each detector (e.g. including one image at one degree step within the CRAV, or one image every two or more degrees within the same CRAV) on the detection accuracy.

4.3.1 Initial Validation

The following case study was done to validate the proposed method. The 3D model of an excavator was taken from Google Warehouse and it was imported to Autodesk 3Ds Max for the visualization purpose. The lighting and reflection conditions are manually set to reflect realistic conditions as much as possible based on the user judgment. The background of the excavator is set to white. The virtual camera is set on the circle path around the object tilted by 75˚ from the horizon. The images were taken every 3.6˚. On the other hand, a Matlab toolbox was developed for the auto-annotation of construction equipment images generated in Autodesk 3Ds Max. Following the proposed methods, the developed toolbox is able to find the equipment in the image and then calculate the coordinates of the bounding box. Additionally, the toolbox is able to add various backgrounds and regenerate new images in addition to the white-background images.

A number of 765 (45×17) synthetic images with 16 additional backgrounds (e.g. Figure 4-9) are generated and then annotated using the proposed method. The annotation data (generated bounding box around the target object) are visually checked for all images. The evaluation shows that all synthetic images were annotated correctly, which means that the accuracy of the auto-annotation of the synthetic images is 100% as it was expected.

Four detectors are trained and evaluated in this test in which the first three detectors are prepared using the proposed method. The first object detector is trained based on HOG using the bounding-box for all the generated images with single background color. Basically, the positive samples contain 45 images of an excavator. The automatic annotation took 1.42 seconds. The negative samples used in the training are 650 images for this test and the following tests. The negative samples are the same as the negative samples in INRIA Person dataset (Dalal & Triggs, 2005). All samples have the side-view of the excavator either from the right or left side since the view of the excavator from the right side is different from its left side.

The second detector adds two new backgrounds to the positive sample set. The number of the positive samples increased to 135 (45×3) images in the second test and they took eight seconds to be annotated. In the third detector, 765 (45×17) images with 16 additional backgrounds (e.g. Figure 4-9) were used, which consumed 37 seconds for annotation. These 16 backgrounds contain 8 general views of a construction sites and eight pictures zoomed on the same views. The forth detector is based on 200 real images, which were annotated manually. The time that the user needed to annotate all images is 2,387 seconds (approximately 40 minutes). These real images are taken from different construction sites introduced in the research of Tajeen and Zhu (2014).

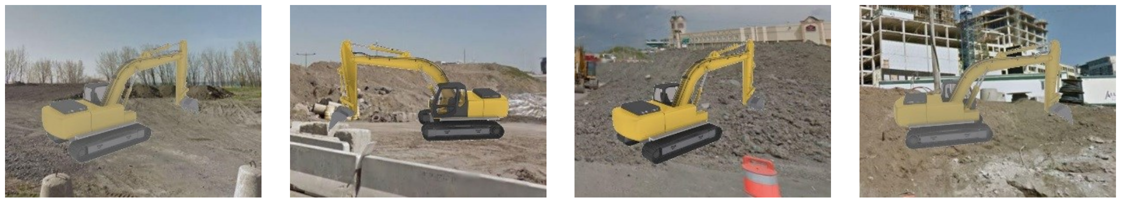


Figure 4-9. Sample of synthetic images generated by the proposed method

The results shown in Table 4-1 highlight the remarkable improvement in the computation time for the equipment images annotation. In this table, the numbers of positive samples for each type of equipment, that were used in the different research projects reviewed in Section 2.6. The estimated time for annotating each dataset is calculated by assuming 11 seconds for each image (the average time it took in this research to manually annotate each image). In the last row, the time spent for the auto-annotation of the dataset generated in this research is provided. Comparing the estimated times in this table clearly shows that the proposed method is able to greatly reduce the time and the human efforts. Moreover, the number of the images in the last row can be increased in the future by considering multiple horizontal angles of views, various types of excavators, and more background images while the required time will continue to be much shorter compared with the previous studies.

Sixty images are selected for the detection testing part. Fifty images are taken from the Internet containing an excavator from a different brand. 10 images, which do not include any excavator, are selected to verify the true negative value of the model.

Table 4-1. Comparison of estimated annotation time between the equipment image datasets

Papers	Equipment	Number of Positive Images	Estimated Time (s)
Park & Brilakis (2012a)	Wheel Loader	1,015	11,165
Chi and Caldas (2011)	Wheel Loader	150	1,650
Chi and Caldas (2011)	Dump Truck	150	1,650
Chi and Caldas (2011)	Tractor Truck	150	1,650
Chi and Caldas (2011)	Car	150	1,650
Azar and McCabe (2011)	Dump Truck	4,558	50,138
Memarzadeh et al. (2013)	Excavators	1,895	20,845
Memarzadeh et al. (2013)	Dump Truck	1,212	13,332
Tajeen and Zhu (2014)	Excavator	800	8,800
Tajeen and Zhu (2014)	Loader	300	3,300
Tajeen and Zhu (2014)	Dozer	300	3,300
Tajeen and Zhu (2014)	Roller	300	3,300
Tajeen and Zhu (2014)	Backhoe	300	3,300
Proposed Method	Excavator (only side views)	765	37

As shown in Table 4-2, (True Positive) TP refers to the positive instances, which are recognized correctly (more than 50% overlap between the detected bounding box and the ground truth), (True Negative) TN refers to the negative instances, which are correctly labeled as negative, (False Positive) FP refers to the negative instances, which are wrongly considered as positive, and (False Negative) FN refers to the negative recognitions, which are wrongly labeled as negative.

Moreover, the precision, recall, and accuracy values are calculated using Equation 4-1, Equation 4-2, and Equation 4-3 (Tajeen & Zhu, 2014; Taylor, 1997).

$$Precision = \frac{TP}{(TP + TN)}$$ **Equation 4-1**

$$Recall = \frac{TP}{(TP + FN)}$$ **Equation 4-2**

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$ **Equation 4-3**

Comparing the tests with real images and virtual images with a single background color shows improvement in precision but reduction in recall and accuracy (Table 4-2). Precision and accuracy are increased by adding two backgrounds but still the recall is lower than the detector based on real images. The problem with recall is solved when 16 images are used as background.

Table 4-2. Comparison of excavator detection results

Type of images	Number of positive images	TP	TN	FN	FP	Precision (%)	Recall (%)	Accuracy (%)	Annotation Time (s)
Real Images	200	27	7	15	11	71	64	57	2,387
3D Model with Single Background	45	23	10	26	1	96	47	55	1.42
3D Model with 2 Backgrounds	135	29	9	19	3	91	60	63	8
3D Model with 16 Backgrounds	765	42	3	1	14	75	98	75	37

This results also in a reasonable improvement in precision and accuracy. Based on the results of the tests, it can be concluded that training positive samples using a 3D model of the construction equipment can make the detectors more accurate. Obviously, the time spent for training using the virtual samples is dramatically lower than the time spent for training using real images.

4.3.2 Sensitivity Analysis

This case study aims to determine the effect of the CRAV and the number of step angles for each detector on the precision, recall, and accuracy of each detector. The same basic setting of the first case study is used. In the first analysis, the left view of the excavator is selected with a 90˚ CRAV. As shown in Figure 4-10 the camera is relocated every one, two, four, eight, sixteen, and thirty-two degrees within the 90˚. Four scale sizes are considered and each one is half of the previous one. Moreover, the illumination condition is increased in three levels from shiny to dark and 14 background images are added to each raw image. Six detectors are trained individually using the HOG method and the same testing dataset used in Section 4.3.1 is used to evaluate each detector.

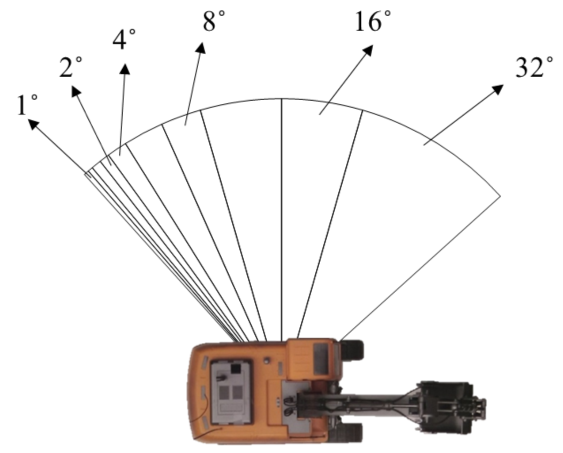


Figure 4-10. Guide plan for step divisions

The results of the analysis are summarized in Table 4-3. Increasing the angle between image collection locations reduced the number of true positives. Consequently, precision, recall, and detection accuracy were decreased. Based on the achieved performances, relocating the camera at every one degree shows the maximum accuracy of the detector.

Another analysis is done to determine the effect of increasing the CRAV for each detector. While the focus of all images in this test are on the left-side of the excavator as shown in Figure 4-11, the CRAV is decreased from 90˚ to 6˚.

Table 4-3. Results of excavator detection for different step divisions

Degree (s)	Number of positive images	TP	TN	FN	FP	Precision (%)	Recall (%)	Accuracy (%)
1	16,200	46	3	0	11	81	100	82
2	8,100	42	3	1	14	75	98	75
4	4,140	40	5	3	12	77	93	75
8	2,160	34	5	9	12	74	79	65
16	1,080	24	8	19	9	73	56	53
32	540	19	10	21	10	66	48	48

Based on the results in Table 4-4, using a wider CRAV in training of the detector results in higher detection rate; however, both true positive and false positive values increased at the same time. These increments raised the recall value and accuracy but reduced the precision of the detector. Overall, it can be said that it is more promising to have at least 45˚ CRAV to achieve reasonable results.

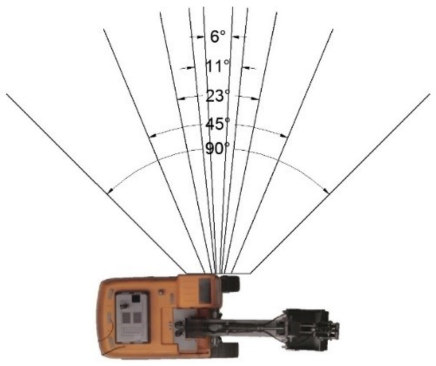


Figure 4-11. Guide plan for tested CRAV

With respect to the results provided in this section, the object detectors, which were trained using the synthetic images showed promising accuracy. Moreover, the auto generation of the negative sample, combined with different target object sizes and illumination levels boosted the performance of the detectors.

Table 4-4. Results of excavator detection for different CRAV

Field of View	Number of positive images	TP	TN	FN	FP	Precision (%)	Recall (%)	Accuracy (%)
90˚	16,200	46	3	0	11	81	100	82
45˚	8,100	42	5	4	8	84	91	80
23˚	4,140	30	7	17	6	83	64	62
11˚	1,980	24	8	22	6	80	52	53
6˚	1,080	29	8	13	10	74	69	62

4.4 Summary and Conclusions

This chapter investigated a new approach of generating and annotating synthetic images automatically using a 3D model of the equipment to recognize in the images of construction sites. The results show that the proposed automatic annotation method using synthetic images is able to play the role of real images captured from the construction site for training propose This method could be used to create comprehensive image datasets for various construction equipment that can be applied for training of vision-based equipment detectors. Since the synthetic images do not need the users to go to the construction sites and to capture the images of the equipment under various conditions, it can save a lot of time to apply a system for monitoring the equipment on sites. Moreover, the automatic annotation significantly reduced the required time for defining ROI compared to traditional annotation methods. The synthetic images based on the 3D model of construction equipment can be used for training the detector, which was able to recognize the target object within a new dataset. The accuracy of true positive detections, precision, and recall were increased by generating more synthetic images with more backgrounds related to the construction environment.

The main advantage of the proposed method is that it can significantly reduce the time required for annotating the images of construction equipment. On the other hand, well-structured annotation results help the detector algorithm to provide a higher true detection rate. Furthermore, the results show that selecting appropriate negative samples helps the detector distinguish the true positive from the false negative correctly. For instance, to differentiate an excavator from a mobile crane in the same construction site, it is necessary to use the mobile crane as a negative sample in training the detector model.

The conclusions of this chapter are as follows: (1) The rendered 3D model of the construction equipment can be used to produce images with a single color background that can be annotated automatically using the proposed method; (2) Synthetic images can be produced by integrating the images of the 3D model with the desired background from different construction sites, different size of the objects, and different illumination conditions for the training phase; (3) The auto-annotation process can be used for the synthetic images based on the annotation results of the images with the single-color background; (4) A large number of negative samples can be produced from the image of different construction sites automatically while the target object is cut from these images; and (5) The results of the HOG detectors using the proposed method were better than those obtained from a detector based on manual annotation of real images of construction equipment. The results showed that the proposed method is able to reduce the annotation time by more than 90% while the accuracy of the object recognition is improved by training more synthetic images.

However, the proposed method has the following limitations: (1) The created images using a random background may sometimes look far from the reality. For instance, the background image could be taken at the street level but the image of the 3D model could be taken from the height of two meters above the street level; therefore a mismatch appears in the synthetic image; (2) Weather conditions are another difficulty for creating this kind of synthetic images. Foggy and rainy weathers will cause the scene to look different than when the weather is clear; and (3) The efficient number of detectors, which can cover all views around the objects needs to be determined. These limitations should be investigated to make the synthetic images of the construction site more natural and the detectors should be trained using different sizes of images to cover various qualities and sizes of the target image. Additionally, using Augmented Reality (AR) techniques can help by matching the view of the 3D model and the background (Furht, 2011).

CHAPTER 5 DATA FUSION AND EXCAVATOR PARTS DETECTION

5.1 Introduction

In order to estimate the pose of excavators, the video obtained from stereo cameras are used as the main source of data. However, off-the-shelf stereo cameras usually have a short baseline (i.e. less than 1 *m*). Okutomi and Kanade (1993) stated that having two cameras close to each other decreases the accuracy of the depth estimation for the objects far from the cameras. The proposed pose estimation approach uses two surveillance cameras as stereo cameras that can be installed at a distance of more than 10 *m* from each other, and at a level of more than 10 *m* from the average ground level of the construction site. This setting of the stereo cameras requires special attention to the calibration process of the cameras. Moreover, relying on a single source of data may not only decrease the accuracy of the pose estimation system because of missing data or calculation errors, but it also may increase the computation time.

The objectives of this chapter are: (1) to fuse the RTLS data and two or more cameras data by synchronizing the time and the coordinate systems of the cameras and RTLS; (2) to evaluate the applicability of stereo cameras with long baseline at a high level; (3) to generate the database of the synthetic images for the excavator's parts and to annotate the images automatically for the training purpose; and (4) to train multiple detectors from different views for each part and then apply on the target images and video frames to recognize the parts from any point of view around the excavator.

As earlier pointed in Section 3.4 and shown in Figure 3-1(c), applying RTLS is proposed as another source of data, which are fused with cameras' data to improve the speed and accuracy of CV processing. The RTLS data, providing the location of the equipment, are linked with the corresponding video frames from two cameras to fuse the location data with the video data. Knowing the overall size of the equipment and its location provided by the RTLS system, the related part of the frame is cropped to focus only on the pixels that show the equipment. Cropping the frames limits the search scope of the object detection algorithm, resulting in a lower computation load and faster processing time.

This method is not limited to the use of specific cameras or RTLS devices. They may be selected depending on the available budget, configuration of the site, the required quality of the results, and their other possible applications on the site. For instance, the fixed IP cameras are widely used for the surveillance propose; spherical cameras are more suitable for providing maximum coverage; and Pan-Tilt-Zoom cameras provide the flexibility of rotating and zooming of the view. RTLS technologies such as GPS, UWB, RFID, Bluetooth, etc., are applicable technologies for localizing the equipment using radio waves. It is assumed that all operational areas of the construction site are covered by at least two camera views and the RTLS infrastructure. As shown in Figure 3-1(a), one RTLS transmitter (e.g. GPS tracker or UWB tag) with a Unique-Identification (UID) number is assigned to each equipment. The name, type, and specifications of each equipment are stored in a database based on the equipment UID. Moreover, all of the trained detectors are stored in the same database. Using the location of the RTLS transmitter installed near the center point of the target equipment, the cameras containing the transmitter in their Field of Views (FoVs) are selected for further analysis. In the example shown in Figure 3-1(b), Camera 2 and Camera 3 are selected. A 3D virtual cylinder is added at the center location of the target excavator in each camera view based on the dimensions of the excavator. Comparing the size of the cylinder in the view of Camera

2 and the view of Camera 3 shows that the cylinder in the view of Camera 2 is smaller than the one in the view of Camera 3 due to the longer distance between the target excavator and Camera 2. In the next step, part detection is shown in Figure 3-1(c). A rectangle bounding box is cropped around the cylinder in each view. Within each bounding box, multiple object detectors trained for the excavator parts are applied. The details of each phase are explained in the following subsections.

Moving from data fusion phase to the parts recognition phase, this step is to process each frame of the video captured from each camera individually for recognizing the part, which is the pre-requisite for extracting the skeleton of the excavator from the view of that camera. Using the auto-annotation method explained in Section 4.2, the synthetic images of the excavator parts are generated and multiple parts' detectors are trained.

5.2 **Data Collection and Fusion Phase**

This phase is invariant to the type of the target equipment, which means that it will be the same for any construction equipment. Accurate superimposing of the RTLS data on video frames is highly dependent on the precise calibration of the cameras. As shown in Figure 5-1, this phase has the following processes: (1) The calibration phase starts by preparing the checker board. The size of the squares on the checker board should be large enough to be visible when the board is far from the cameras. As explained in Section 2.5.5, the camera calibration process in this chapter is based on a checker board pattern, which has odd number of black squares on one direction and even number on the other direction. The interest points are the intersections of the squares. (2) Many synchronized images are captured from the available cameras. (3) A local coordinate system is defined to transform all the cameras and RTLS coordinate systems into a single one. (4) The interest points are used to estimate the intrinsic parameters of each camera. After calibrating each camera separately, the next step is the estimation of the extrinsic parameters of each camera based on the unified coordinate system. In this process, the rotation and translation of each camera are estimated relative to this system. In case of more than two cameras, every set of two cameras are calibrated together.

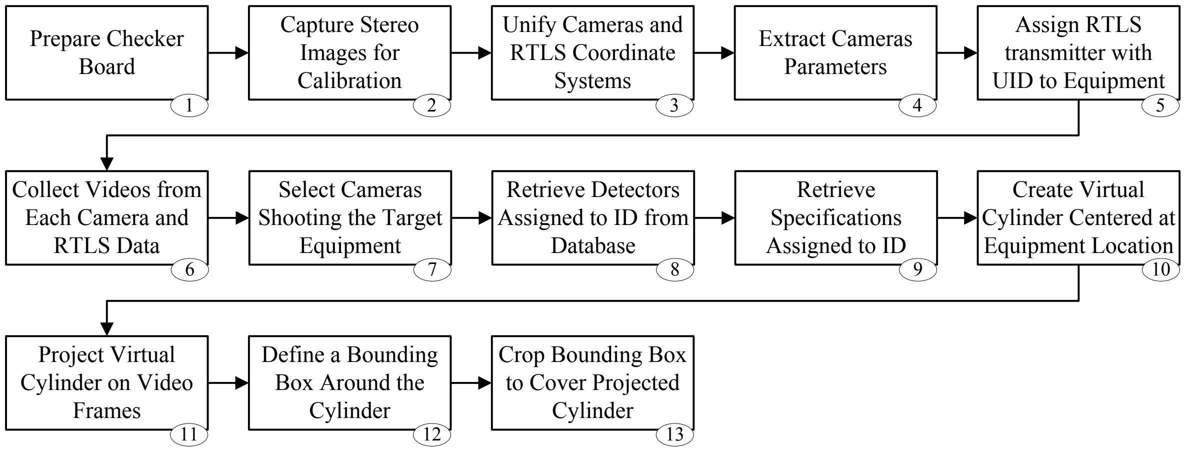

Prepare Checker Board
1
Capture Stereo Images for Calibration
2
Unify Cameras and RTLS Coordinate Systems
3
Extract Cameras Parameters
4
Assign RTLS transmitter with UID to Equipment
5
Collect Videos from Each Camera and RTLS Data
6
Select Cameras Shooting the Target Equipment
7
Retrieve Detectors Assigned to ID from Database
8
Retrieve Specifications Assigned to ID
9
Create Virtual Cylinder Centered at Equipment Location
10
Project Virtual Cylinder on Video Frames
11
Define a Bounding Box Around the Cylinder
12
Crop Bounding Box to Cover Projected Cylinder
13

Figure 5-1. Processes of data collection and fusion phase

Fusing the RTLS and cameras data can reduce the search area during localization of each part. In order to support the claim that limiting the search area during localization of each part can decrease the computation time, the processing time for searching for each part when increasing the size of the search area can be estimated. As shown in Figure 5-2, h_i is the height and $w_i$is the width of the image at the i^{th} level of the scale pyramid ($i \rightarrow \{1, 2, \dots, n\}$) and n is the highest level of the pyramid, h' is the height ($h' \leq h_i$) and w' is the width ($w' \leq w_i$) of the sliding window, and q is the overlap length of the sliding window with its predecessor or successor windows. The ratio of the searching area of the image at i^{th} level over the area of the sliding window s_i is calculated using Equation 5-1. Equation 5-2 is used to calculate the number of sliding windows (r_i) that have to be checked for detecting the target object. Assuming that for checking each sliding window ε amount of time is required, the total time for all sliding windows is $r_i \times \varepsilon$.

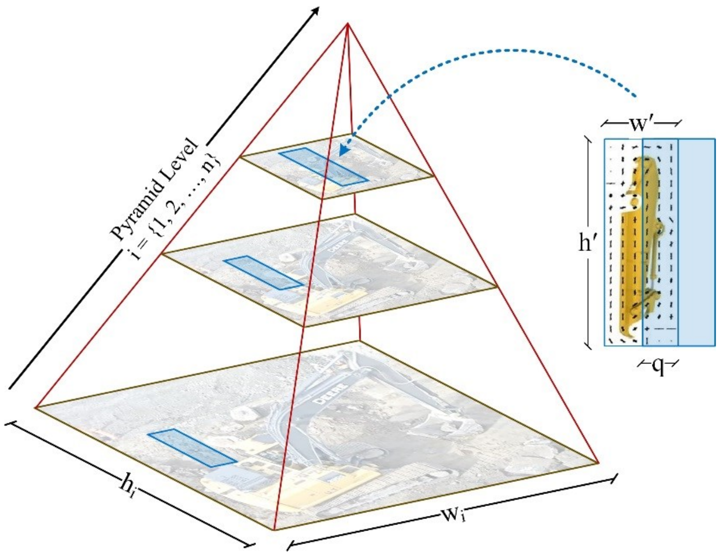


Figure 5-2. Configuration of image pyramid and sliding window (adapted from Felzenszwalb et al., 2008)

$$s_i = \sqrt{\frac{h_i \times w_i}{h' \times w'}}$$ **Equation 5-1**

$$r_i = \left(\frac{(s_i - 1)}{(1 - \frac{q}{w'})} + 1\right)^2$$ **Equation 5-2**

Equation 5-2 can be used to estimate the processing time for searching for each part when increasing the size of the search area. The processing time for three scenarios are investigated when there are zero, 50%, and 75% overlaps between the sliding windows and *i* is 1. Figure 5-3 shows the relation between increments in the rate of the image's searching area and processing time for all sliding windows within each frame. The rate is varied from 1 to 40. Having the rate of one means that the size of frame is equal to the size of sliding window. The vertical axis shows the processing time for each increment rate. For instance, when the size of the frame in pixels is 1920

× 1080 ($h_1 \times w_1$) and the size of sliding window in pixels is 96 × 54 ($h' \times w'$), s_1 is 20. From Figure 5-3, it can be determined that for this size ratio, the processing time will 400 ε, 1,521 ε, and 5,929 ε, for no overlap, 50% overlap, and 75% overlap, respectively. Assuming that the size of each sliding window is fixed, when the size of image is increased, the processing time is increased dramatically. Therefore, having smaller search area to apply each detector can save a lot of time.

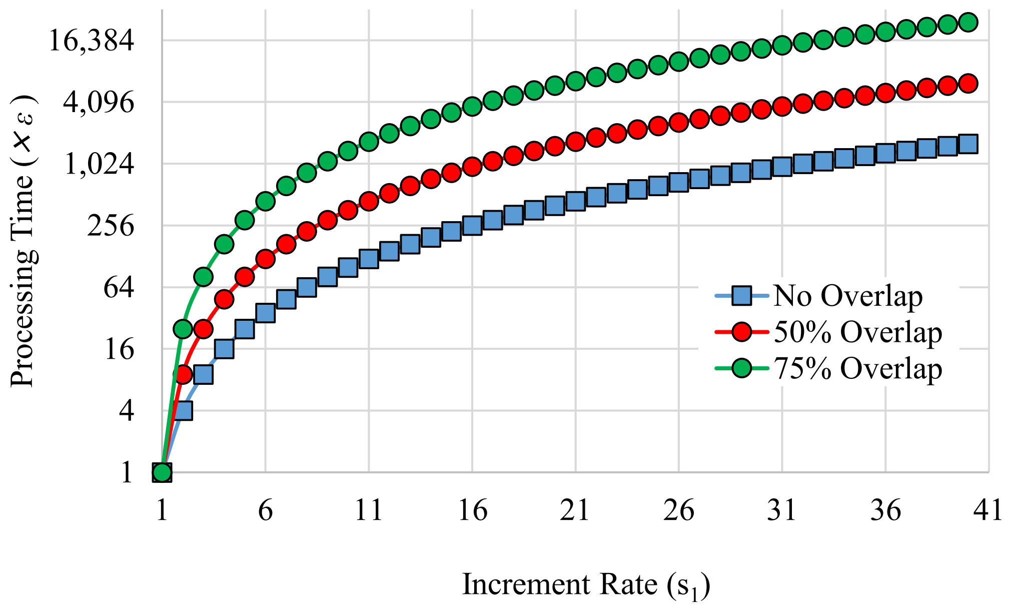
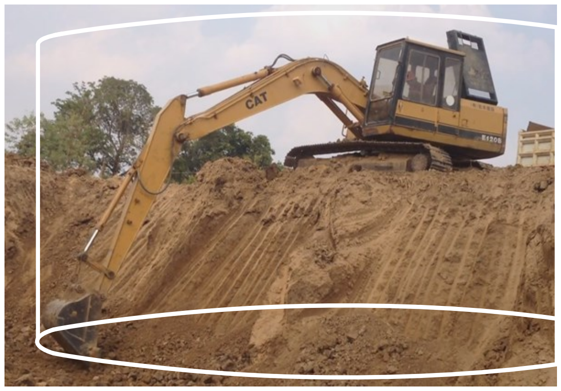


Figure 5-3. Relationship between increment rate and processing time

(5) After calibrating the cameras and RTLS, an RTLS transmitter with a Unique Identification (UID) number is assigned to an equipment. (6) The next step is collecting the videos from each camera and the location data of all RTLS transmitters. It should be noticed that the following process requires precise time synchronization between video frames and RTLS data. (7-9) The cameras that are shooting the location of the target equipment are selected and the related parts' detectors and the specifications of that equipment are retrieved from the database based on the UID of the transmitter. (10) A 3D virtual cylinder centered at the equipment location at time *t* should be generated using the dimensions of the equipment. The cylinder should cover the extreme

poses of the excavator including the poses corresponding to the maximum digging depth (Figure 5-4(a)) and maximum reach length (Figure 5-4(b)). The required information regarding the extreme poses can be extracted from the specifications document of each excavator. A sample data sheet for an excavator is provided in Figure 5-5(a), which shows the dimensions of the excavator at different poses.


(a) Maximum digging pose (E-digging, 2016)

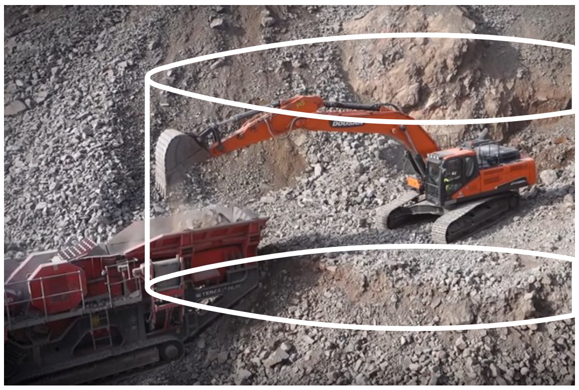
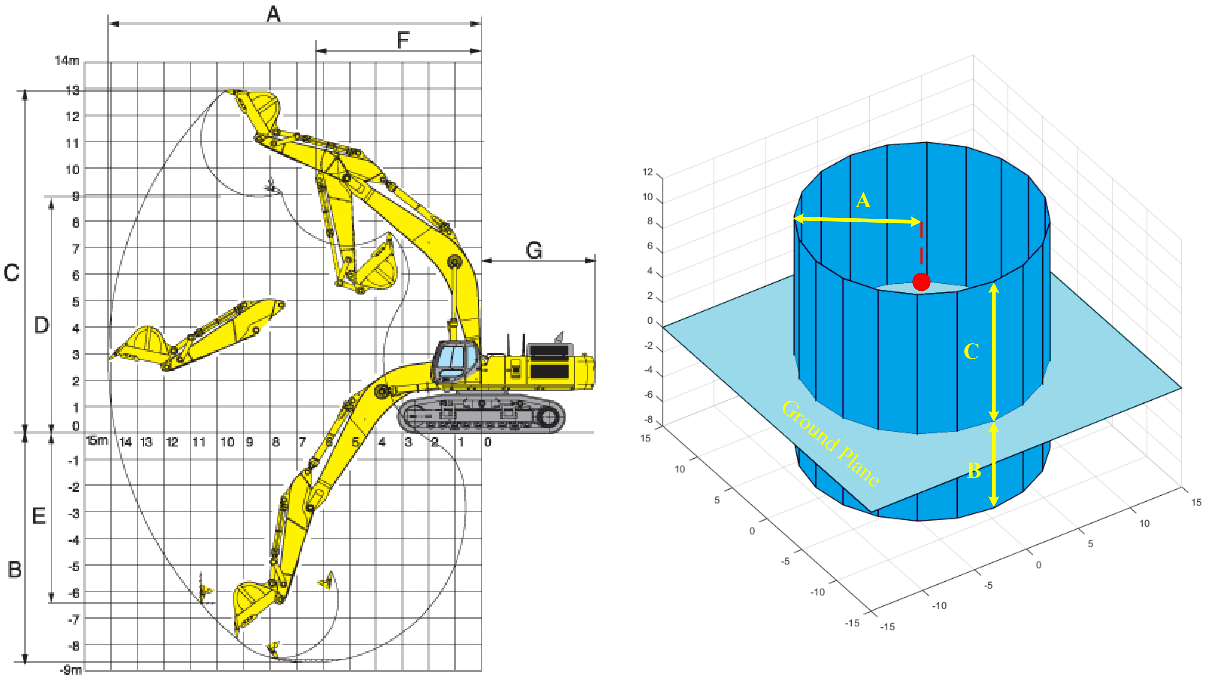

(b) Maximum reach pose (KUBO-SK, 2015)

Figure 5-4. Extreme poses of excavator

The most important variables for creating the virtual cylinder shown in Figure 5-5(b) are *A* (maximum reach length), *B* (maximum digging depth), and *C* (maximum cutting height). Additionally, it is assumed that the ground does not have slope within the excavator contact surface with the ground. (11) After creating the 3D virtual cylinder using the aforementioned variables, the cylinder should be projected on the video frame at time *t* of each selected camera. There are two main options for this task. The first one is to project the virtual cylinder first onto the 2D pixels coordinates of the video frame at time *t* and then re-project from this camera onto the view of the next camera using the relative translation and rotation between these two cameras. The second approach is to project the virtual cylinder onto the view of each camera separately. However, the first approach may suffer from the errors included in estimating the relative translation and rotation between two cameras. (12) The next step is to define a bounding box around the projected cylinder on each view. (13) The content of the bounding box is cropped from the original image and passed to the next phase.



(a) Excavator specification (SUMITOMO, 2017) (b) Cylinder dimensions

Figure 5-5. Defining virtual cylinder based on excavator specifications

5.3 Excavator Parts Detection Phase

As schematically shown in Figure 3-1(c), the method creates synthetic images of an excavator and its parts using the 3D model of the excavator and some images from different construction sites to be used as background. Knowing that each part of the excavator looks different from different points of view, it is necessary to have multiple detectors for each part. Each detector should cover a range of views that are close to each other. In this research, the around view of the excavator is divided into six zones as shown in Figure 5-6 (left, right, front-left, front-right, back-left, and back-right). The idea of dividing the around view into six zones came from Azar and McCabe (2011) and Rybski et al. (2010) where the range of views was divided into eight zones. In the aforementioned research, passenger vehicles and dump trucks were the target of recognition. After investigating the difference between those object and the excavators that are going to be recognized in the current research, it was found that six zones are enough to provide similar results. Moreover, training the detectors also requires many effective negative samples that can reduce the chances of false detections. The detectors are then applied on the video frames from the construction site to find the parts of the excavator.

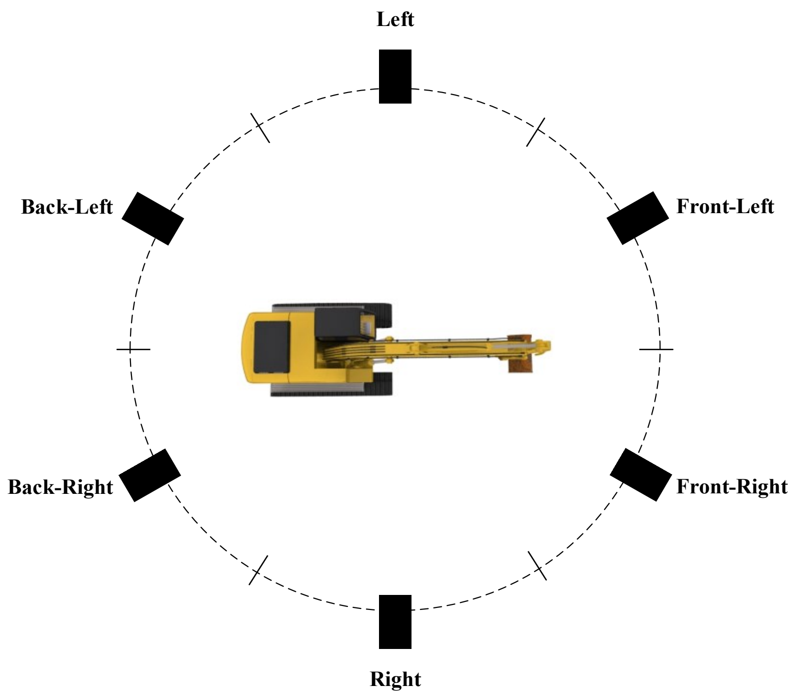


Figure 5-6. Camera view zones around the excavator

5.3.1 Generation of Positive Images

As it was mentioned earlier, the 3D model of the excavator is used for creating the around-view images. This research assumes that the background of these images is set to a single color. The images are overlaid on the real images of the construction sites as background to create the synthetic images. The same process is repeated for each part of the excavator.

First, the image of each part with the single color background (Figure 5-7(a)) is segmented to recognize the part, and a bounding box is drawn around each part (Figure 5-7(b)). Then, the generated boxes are added to the image of the whole excavator (Figure 5-7(c)) from the same view, and the background image (Figure 5-7(d)) is added to the image. As a result, Figure 5-7(e) shows the annotated image of the excavator's part (dipper in this example).

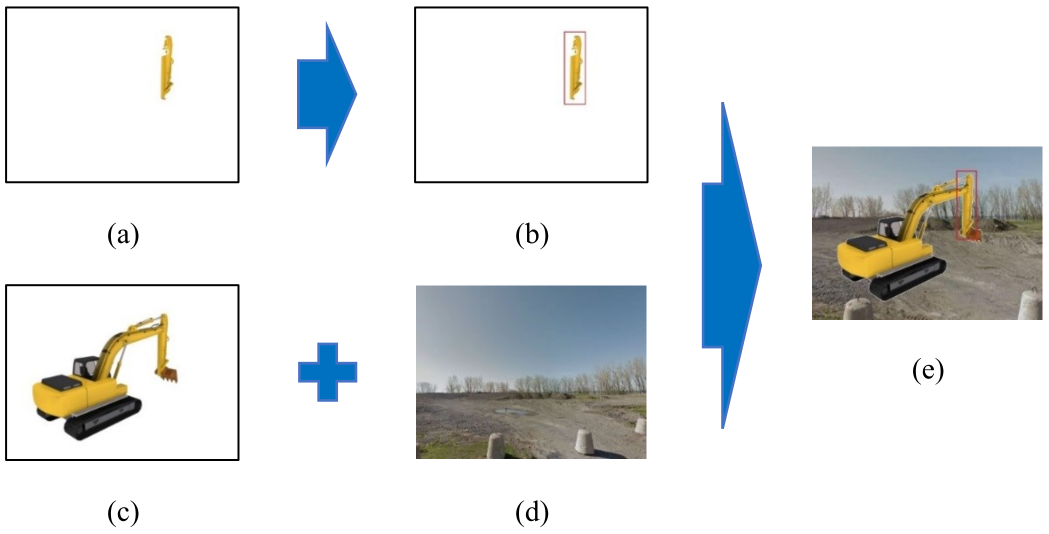
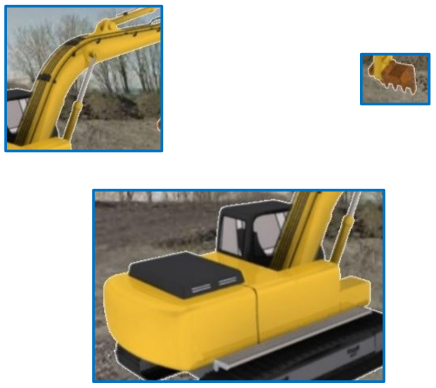


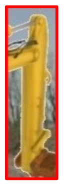
Figure 5-7. Parts auto-annotation process

5.3.2 Generation of Negative Images

Knowing that effective negative images can greatly help with the reduction of false positive rate, the method of auto-generation of negative samples proposed in Section 4.2.3 is used. Additionally, the negative samples should be extended so that it is possible to differentiate each part of from other parts. The need for extra negative samples is fulfilled by using the bounding boxes around the parts other than the target in the previous steps. As an example, the images of the boom, bucket, and body of the excavator (Figure 5-8(a)) are used as negative samples for training the detector for the dipper (Figure 5-8(b)).



(a) Negative Samples



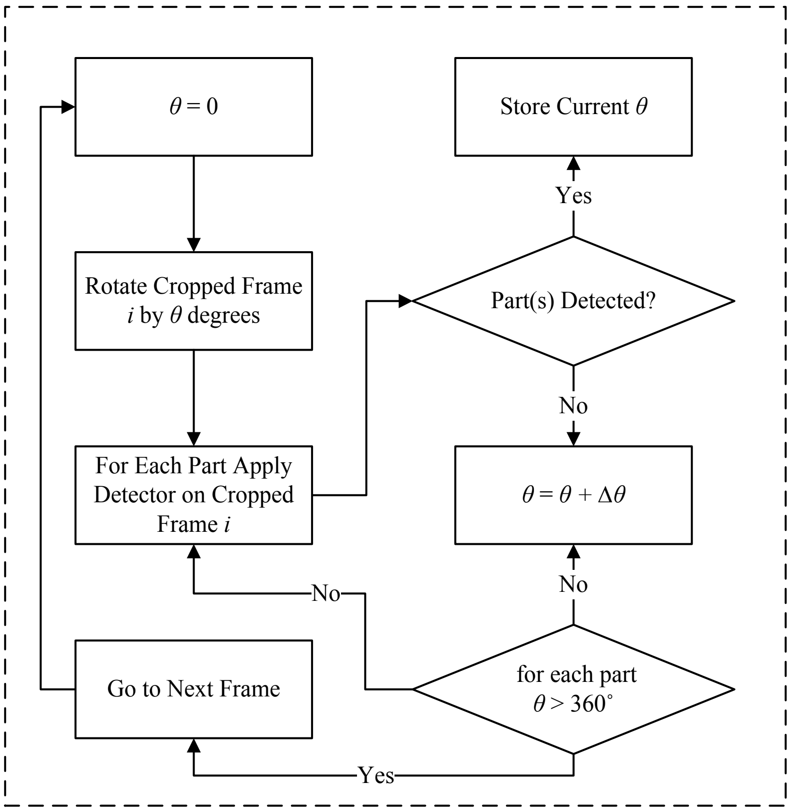
(b) Positive Sample

Figure 5-8. Training samples for the dipper detector

5.3.3 Part Detection Strategy

The detectors should be applied on each frame from each camera view independently. The process flow of the proposed approach is shown in Figure 5-9. One of the difficulties for detecting the excavator parts is to have a robust system for detecting the boom and dipper while their orientations are changing. A common practice is to apply a sliding window on the image and each window is checked for whether it contains the target object or not. For instance, the dipper detector is designed for the vertical pose of the dipper as shown in Figure 5-10(a) but having a rotated dipper as shown in Figure 5-10(b), the previous detector is not able to localize the dipper anymore.

Basically HOG-based detectors are not invariant to rotation, which means that if the trained object appears with a different orientation than the one in the training dataset, the detector will not be able to detect the object. Three methods are possible for this case: (1) Training more detectors to cover various orientations of the part. Since there should be a rectangle box around the dipper in the training images, by rotating the dipper, the shape of the vertical rectangle around the dipper is changed to a nearly square shape where the dipper looks like a diagonal element within the bounding box (as shown in Figure 5-10(b)). However, this may decrease the accuracy of the background subtraction since the portion of the background that needs to be processed for subtraction is increased.

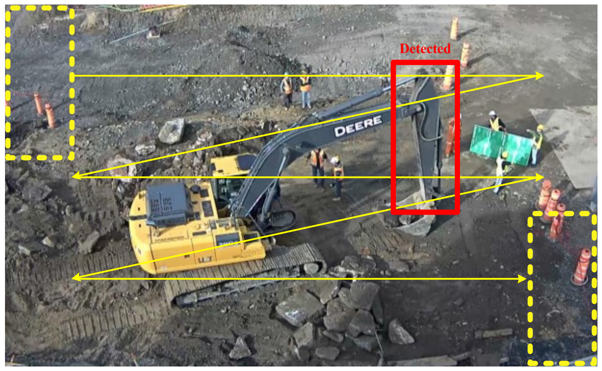


$\theta = 0$
Store Current θ
Yes
Rotate Cropped Frame i by θ degrees
Part(s) Detected?
No
For Each Part Apply Detector on Cropped Frame i
$\theta = \theta + \Delta\theta$
No
No
Go to Next Frame
for each part $\theta > 360^{\circ}$
Yes

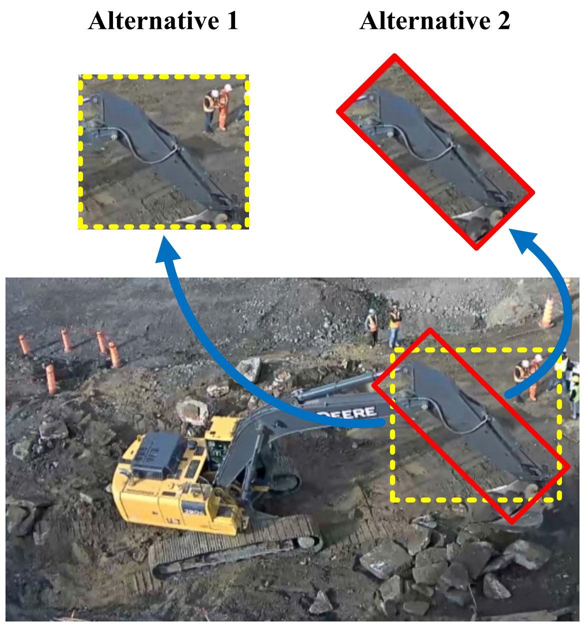
Figure 5-9. Proposed processes for 3D pose estimation phase

(2) Applying multi-phase classifier to detect and estimate the orientation of the object (Villamizar et al., 2010) and training detectors for orientation-related feature then applying rotation invariant detectors (Liu, et al., 2014 and Kittipanya-ngam & Lung, 2010) are among the popular solutions for recognizing the rotated objects (as shown on Alternative 2 in Figure 5-10(b)). However, training the features other than the appearance related features adds more uncertainties to the detectors. (3) Another method proposed by Chiang and Wang (2014) uses the common HOG detector without considering the rotation features during the training phase. Rather, this method rotates the given image successively and it applies the detector at each angle step as shown in

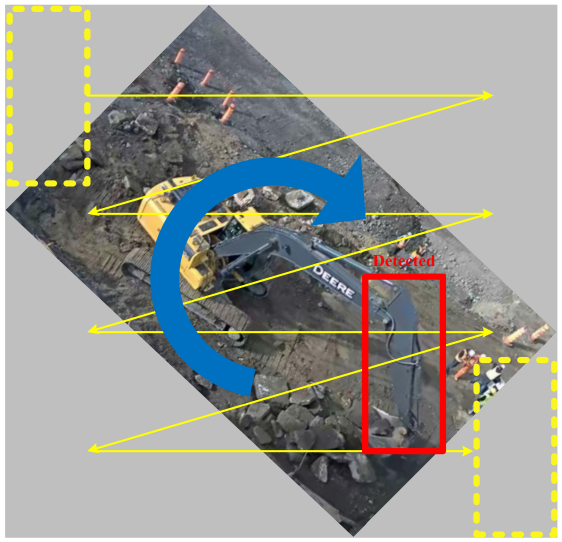
Figure 5-10(c). It should be considered that this method has a high computational cost since it requires to test the classifier several times over the image (Liu, et al., 2014 and Kittipanya-ngam & Lung, 2010). In this research, the method of Chiang and Wang (2014) is selected in spite of the high computation cost since this method is expected to provide a more robust system.



(a) Dipper with vertical pose



(b) Dipper with diagonal pose



(c) Rotating image inverse to rotation angle of dipper

Figure 5-10. Detection complication due to changes in dipper orientation

Having a detector for the initial pose of each part is proposed while it is trained to have lower number of false positive detection by sacrificing for the higher number of false negatives. This helps the detector to only catch the parts very close to the training dataset. As shown in Figure 5-9, if the detector cannot find any dipper in the first trial with $\theta = 0°$, a $\Delta\theta$ is added to θ and the new θ is checked whether it is allowed to be increased or if all the possibilities for rotating the frame have been tried. In case that there is no more chance for rotating the frame, the algorithm will go to the next frame otherwise the detector is re-applied on the rotated frame. Moreover, for the parts that underwent rotations during the detection phase, those rotations have to be inversed at the time of moving from the local to global coordinate systems.

5.4 Implementation and Validation

The proposed framework was evaluated by implementing the aforementioned methods in Matlab 9.1 (Mathworks, 2016). This section covers the implementation and validation of the proposed method in three aspects: cameras calibration, data fusion, and part detection. The implementation process for each of them is explained in a separate subsection followed by the validation for each phase. Moreover, three datasets are used to validate the proposed method: (1) the validation dataset of the static images, (2) the consecutive frames extracted from a video, and (3) stereo views from two cameras. The first test aims to evaluate the performance of the parts detectors on various images with different resolutions. Another objective of the first test is to apply the proposed method on different types of excavators from different manufacturers and observing the results of using a general model of the excavator. The second test focuses on an excavator similar to the one used in the training dataset where the excavator in the video performs a full cycle of its tasks. The last test covers all the aforementioned aspects using the data collected from a construction site in Montreal, Canada. As shown in Figure 5-11, four High Definition (HD) cameras were installed on the top of the adjacent building at the height of 13.10 *m* and the distance between each two adjacent cameras was 13.85 *m*. The distance between the cameras and the excavator varied from 20 to 50 *m*. Knowing that the excavator was moving within the blue circle marked as *Target*, Camera 1 and Camera 2 selected since the excavator is within their FoVs.

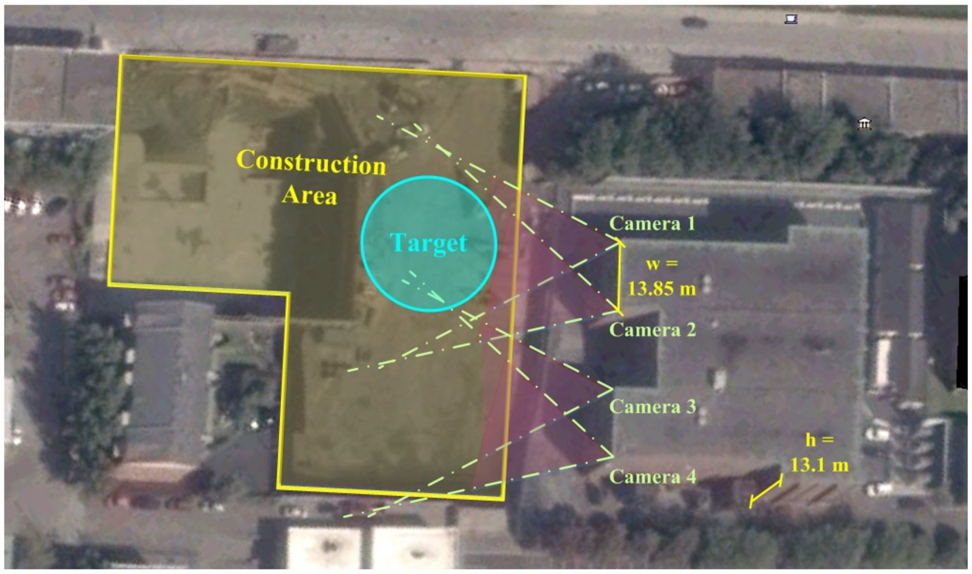
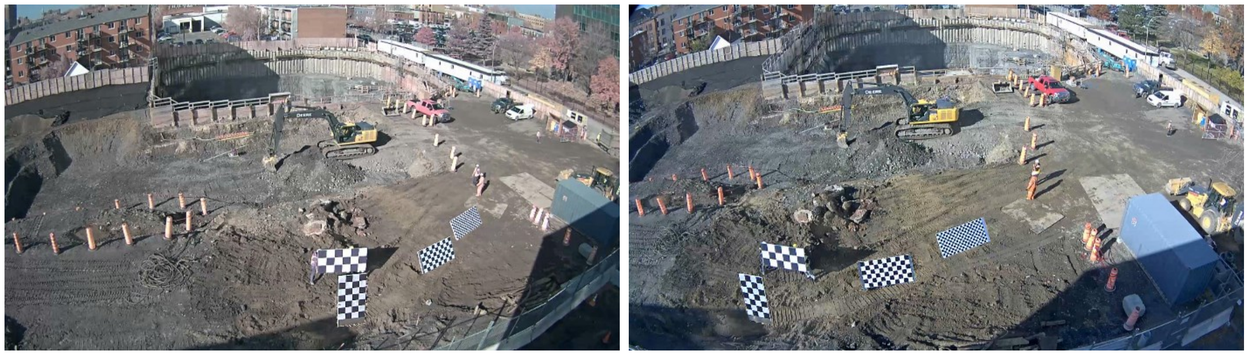


Figure 5-11. Site layout and cameras' configuration

A GPS tracker (QSTARZ BT-Q1000eX (2016)) was used in the experiment as the RTLS technology. It could log the location data at a maximum rate of 10 Hertz (Hz) for 24 hours or at the rate of 1 Hz for 42 hours. The data were stored in the internal memory and then transferred to the server for further processing using a Universal Serial Bus (USB) cable. However, it is possible to add a Bluetooth, WiFi module, or Raspberry Pi single board computer (2017) that can make it possible to stream the data directly to the server in real time. The tracker was placed in the cabin of the excavator operator close to the front window to have better visibility of the satellites. The implementation was done on a workstation with Intel i7 Quad-Core processor, 16 gigabytes Random-Access Memory (RAM), and NVIDIA Quadro 600 graphics card.

5.4.1 Cameras Calibration

The first step for estimating the 3D pose was to calibrate each camera individually and then to find the orientation and translation from one camera to other camera. The origin of the world coordinate system was assumed at the center of camera 1. For calibrating the cameras, four checker boards were printed with the square sizes of 20×20 *cm*, 30×30 *cm*, 40×40 *cm*, and 50×50 *cm* and they were attached on a panel with the dimension of 2.44 *m* × 1.22 *m* as shown in Figure 5-12. The reason behind the use of four boards was to compare the errors for each pattern size. The results can be further used as a guideline, which can help future studies in selecting the appropriate square size of the checker board pattern.



(a) View of Camera 2 (Left) (b) View of Camera 1 (Right)

Figure 5-12. Example of four checker boards with four square sizes

In each trial, one checker board was used and it was placed at different locations with different orientations. It was tried to locate the board within the overlapping view of both cameras at various distances from the cameras. This process was repeated for the four checker boards and at each trial about 100 images were recorded by each camera. After checking each image visually, a smaller number of the images were selected based on the clear visibility of the board in each pair of images.

After collecting the image data, the intersecting points between the squares on the board were extracted manually using Camera Calibration Toolbox for Matlab developed by Bouguet (2004). The actual and the pixel distances between the points were used for estimating the extrinsic and intrinsic parameters of each camera using Equation 5-1. The stereo parameters of the two cameras were estimated using the parameters of each camera individually and the locations of the intersecting points on each image pair using Equation 5-2. The cumulative probabilities of the calibration errors of each checker board for Camera 1 and Camera 2 are shown in Figure 5-13 and Figure 5-14, respectively.

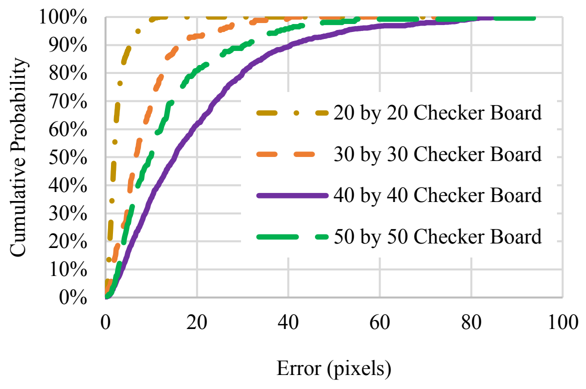


Figure 5-13. Cumulative calibration errors of Camera 1

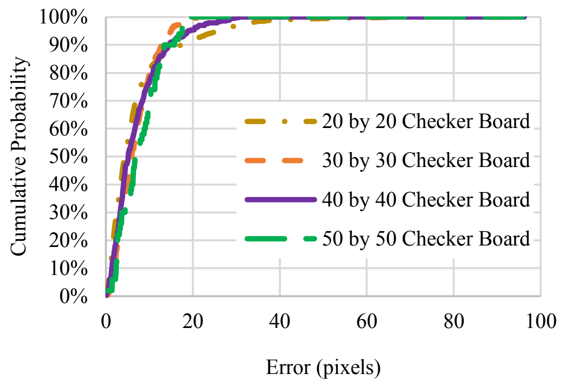


Figure 5-14. Cumulative calibration errors of Camera 2

The checker board with the squares size of 20×20 *cm* shows lower averaged error. The results of Camera 1 have a notable difference compared to Camera 2 where the errors for different boards are close to each other. This may be because the checker boards in Camera 1 views were close to perpendicular to the camera focal axis, which helped the algorithm in estimating the distances between the intersecting points more accurately than the situation with inclined views of the checker boards.

After calibrating each camera separately, the rotation and orientation of Camera 2 were calculated relative to Camera 1 as the origin of the world coordinate system. Four sets of stereo cameras parameters were calculated for the four types of the checker boards. Evaluating the accuracy of the stereo parameters was done by triangulating the intersecting points of each checker board. The distance between the estimated 3D points were calculated and compared with the actual values (i.e. 20, 30, 40, or 50 *cm*). The cumulative probabilities of errors for all four checker boards are shown in Figure 5-15. The best accuracy was achieved when using the checker board with 50×50 *cm* squares and the lowest belonged to the board with 20×20 *cm* squares.

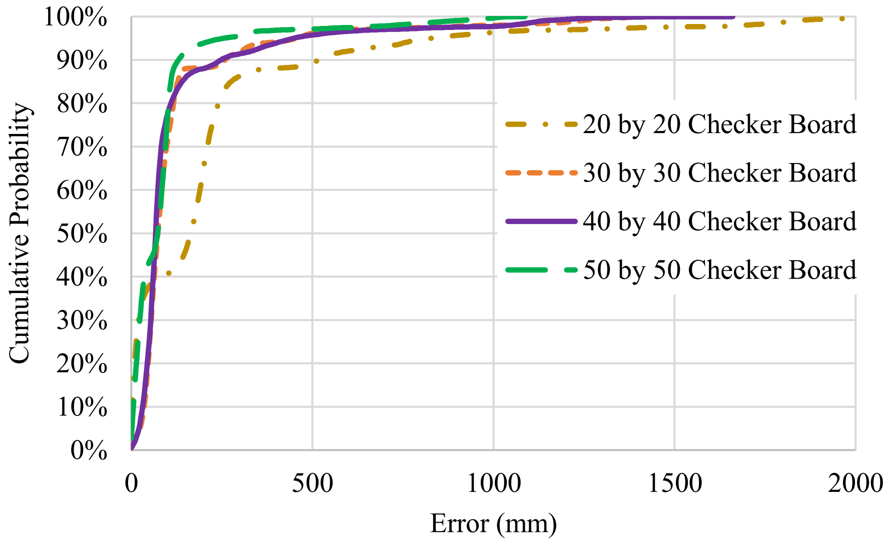


Figure 5-15. Triangulation errors using four different stereo camera parameters

It was expected to have more accurate outcome when using the checker board with the smaller square size but having the best results by using the larger square sizes shows that for the large areas like construction sites, having a checker board with smaller size of squares is not necessarily helpful. This is because when the squares are smaller, the intersecting points are closer to each other and the calibration algorithm cannot differentiate these small distances. The camera's parameters obtained using the 50×50 *cm* checker board are used in the next steps of the case study.

5.4.2 GPS and Video Data Fusion

In order to project the GPS data on the video frames of each camera, the coordinate systems of the GPS and the two cameras have to be aligned. Camera 1 on the construction site is defined as a reference point for this purpose and the three coordinate systems were adjusted to that point to have the same origin. The GPS data was stored as *.kmz* file in the memory of the tracker during the experiment. After finishing the experiment, the data were transferred to the computer using the USB port. Since the update rate of the GPS tracker was 10 Hz and the update rate of the cameras was 30 Hz, a common rate of 10 frames per second with the lag of 6 milliseconds between two consecutive frames is used. This adjustment is necessary to synchronize the GPS data with the correct video frames at a specific time.

The excavator working on this construction site during the data collection was Deere-290G with the maximum reach length of 10.27 *m*, maximum cutting height of 10.26 *m*, and maximum digging depth of 7.05 *m* (John Deere, 2016), which represent *A*, *B*, and *C* shown in Figure 5-5(b), respectively. The origin location of the cylinder was set by the coordinates of the GPS at time *t*. The next step is to project the virtual cylinder for each collected GPS coordinates on the corresponding frames of camera 1 and camera 2. As shown in Figure 5-16, a box was defined around the projected cylinder on each frame and the content of this box was cropped for further processes.

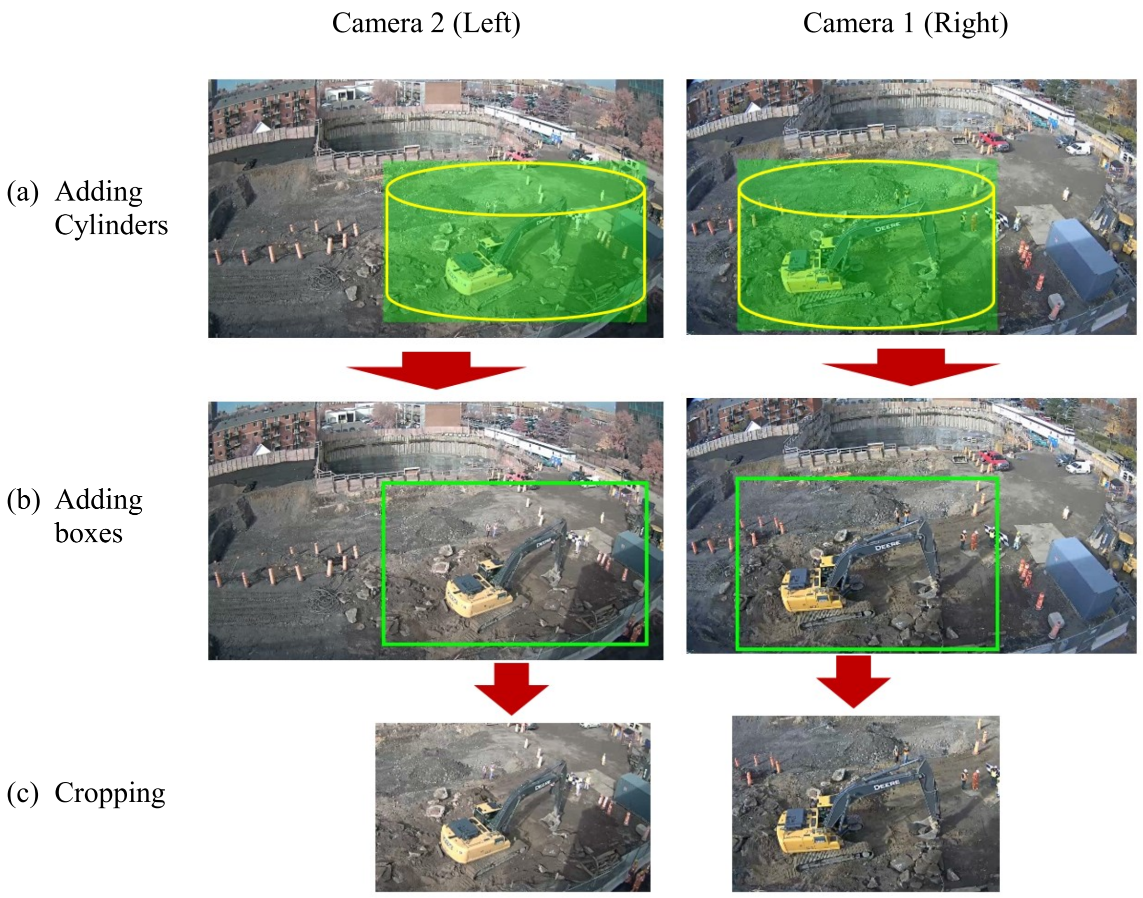


Figure 5-16. Cropping frames based on GPS data

5.4.3 Part Detection

The implementation starts by importing the 3D model of the excavator, which is taken from Google Warehouse, in to Autodesk 3Ds Maxs (Autodesk, 2015) for rendering. Since the lighting condition should be set to match the brightness level of the selected background images, the lighting of the model is visually set to produce close effect to the brightness level of the background images. For instance, when the background is dark, the lighting level of the added virtual excavator cannot be bright. As shown in Figure 5-17, 17 virtual cameras, every 10˚ latitude from the equatorial plane of the sphere are defined to capture the around-view images. The first top camera is set at the altitude of 85˚ on the north direction and the last one at 75˚ on the south direction. Each camera takes the images while turning around the excavator every 1˚ with a white color background. In the first round, all cameras capture the excavator while the whole body is shown. In the second, third, forth, and fifth rounds, only the body, boom, dipper, and bucket is shown, respectively, at a time while the other parts are whitened. As shown in Figure 5-18, whitening is preferred over hiding the other parts in order to consider the possible blockage of the target part by the other parts.

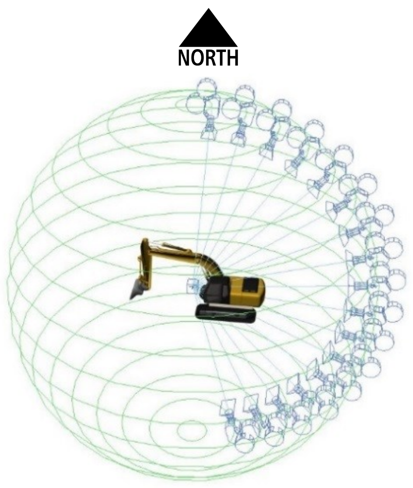


Figure 5-17. Spherical positions of the camera

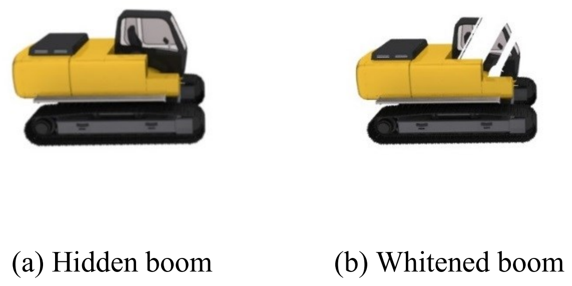


Figure 5-18. Excavator image generation alternatives

Figure 5-19 shows the data architecture within the database of the excavator images. The first layer of the database consists of five children datasets, each for the body, dipper, boom, bucket, and all parts together. In the second layer, 17 datasets are created for every 10˚ latitude under each of the five aforementioned children starting at +85˚ and finishing at -75˚. Considering each latitude angle as a parent at the third level, the children are all the horizontal around-views every one degree. Figure 5-20 shows how the latitude and horizontal angles are defined as α and β, respectively, whenever the camera is focusing on the center point of the excavator.

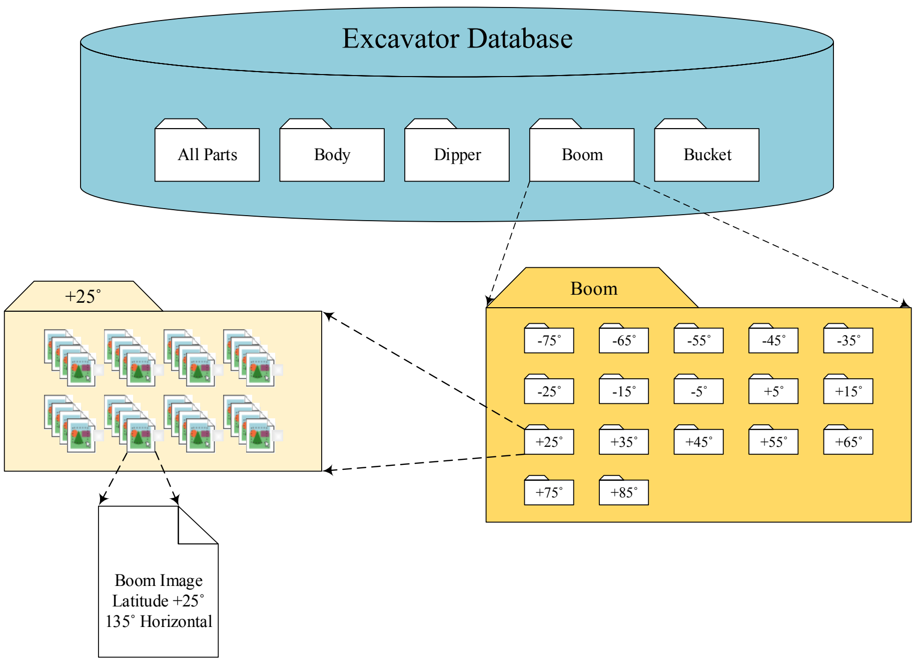
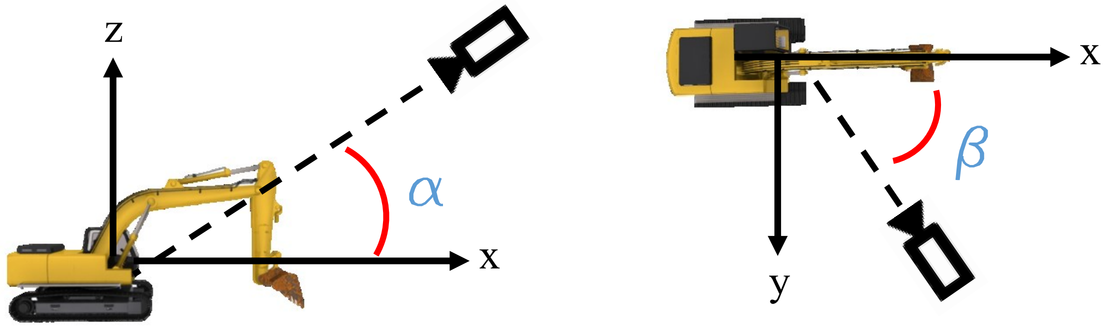


Figure 5-19. Excavator image data structure



(a) Latitude (b) Horizontal angle

Figure 5-20. Attributes definition

As provided in Table 5-1, at each latitude angle there are 360 images for every one horizontal angle, and each part dataset includes 17 latitude angles with 6,120 images in total. The root layer has five divisions containing 30,600 raw images. In order to have a comprehensive database, all views of the excavator are recorded. However, some views are used depending on the site conditions. For instance, the images stored for the view of the excavator at latitudes -65˚ or -75˚

are mainly applicable in the case that the excavator is working on top of a hill while the camera is shooting the scene from the bottom of the hill.

Table 5-1. Excavator image database layers

Data Layers	Excavator	Divisions	Latitudes
Content	5 Divisions	17 Latitudes	360 Horizontal Angles
Number of Images	30,600	6,120	360

The first test is designed to evaluate three cases for each part using a validation dataset of static images of the excavators as explained in Table 5-2. Testing on static images help to evaluate the performance of the detectors on different images from different construction sites with different brands of the excavator while using the video frames limits the test to one brand and one construction site. Fourteen images from different real construction sites (in addition to one white color background) are selected as background to generate the synthetic images. Moreover, three lighting conditions are set for each case. It is assumed that the camera is installed at a higher elevation and the initial range of the latitude angles is between +55° and -5°.

Table 5-2. Tests configurations

	Latitude Range (α)	**Horizontal Angle (β)**	**Size(s)**	**Positive Images**	**Negative Images**
Case 1	-5°– 55°	45°– 135°	1	28,665	116,260
Case 2	5°– 45°	45°– 135°	1	20,475	83,500
Case 3	5°– 45°	45°– 135°	3	61,425	247,300

Moreover, this range is reduced to +5° to +45° for exploring the effect of shorter range on the detection results. In addition to the latitude, there are 360 possible horizontal angles for the camera at each elevation and it can be divided into six zones (as explained in Section 5.3).

However, in the test on the static images, most of the images were from the right and left views. Therefore, it was sufficient to apply two detectors (i.e. from right and left) The first detector for Case 1 uses the images viewed from the latitude of -5˚ to +55˚ and the range of 45˚ to 135˚ of the horizontal angle for its training and the size of the excavator in each image is set to the largest possible size of the excavator in one image (i.e. the maximum size of the excavator that can be shown in an 800 by 600 pixels image). The 91 images from one quarter of the around-view multiplied by 3 lighting conditions, 15 backgrounds, and 7 latitudes result in generating 28,665 synthetic images. In Case 2, the latitude is changed from the range of -5˚ to +55˚ to the range of +5˚ to +45˚ while the other parameters are kept the same. The reduction in the upper bound of the latitude focuses the detector more to the side view. In Case 3, two more sizes of the excavator are added to Case 2 (as shown in Figure 5-21, half, and quarter in addition to the original size of the excavator shown in the image). This change increases the number of the synthetic images for Case 3 to 61,425 images (91 images from one quarter of the around-view multiplied by 3 light conditions, 15 backgrounds, 5 latitudes, and 3 sizes).

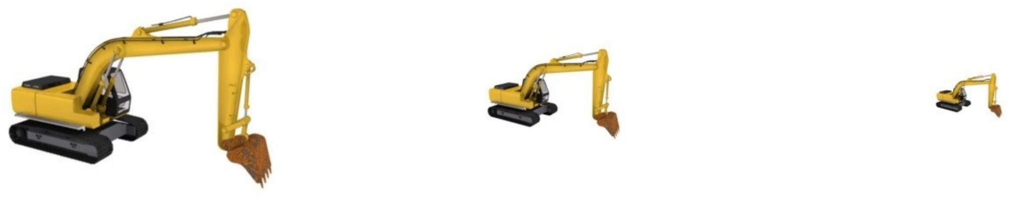


Figure 5-21. Target object multiple sizes for training

Considering 3 sizes helps the detector to recognize the target at different distances from the camera. These three cases are repeated for the dipper, boom, and body but the bucket is taken out of the test because it is usually covered by dirt and soil and it is very hard to differentiate it from the soil.

As mentioned in Section 5.3.2, the auto negative sampler generates 1,600 images randomly out of 14 backgrounds (15 minus one white color background) as the fixed negative dataset. Moreover, the positive images of the parts, other than the target part, are added to the negative sample dataset. The algorithm provided by Dalal and Triggs (2005) is used for extracting and training HOG features of the training image dataset. These features are used for training the detectors with automatic template size. It means that the algorithm uses the median size of the annotated target as the template size. However, the dimensions of all training images are fixed to 800 by 600 pixels.

The detectors for each part and each case are trained and applied on a set of 100 different real images including 75 images of real excavators (there are one to three excavators in each image) and 25 image without any excavator to evaluate the performance of the proposed method. The size of these images is about 250 by 190 pixels. As shown in Table 5-3, the precision and accuracy for detecting the dipper are reduced by decreasing the latitude angles and increasing the number of sizes of the dipper.

Table 5-3. Results of tests on static images

		Precision (%)	**Recall (%)**	**Accuracy (%)**
Dipper Detector	Case 1	61	41	47
	Case 2	74	35	46
	Case 3	85	79	76
Boom Detector	Case 1	84	95	83
	Case 2	87	79	76
	Case 3	94	63	70
Body Detector	Case 1	75	94	75
	Case 2	76	36	46
	Case 3	77	56	58

The best results for the dipper detector was achieved in Case 3. The boom detector and body detector have their best performances in Case 1. The best detector for each part is selected to be used in the next tests on the video frames.

5.4.3.1 Tests on Video Frames

The second test is applied on 250 consequent frames of a video captured from a construction site in Vancouver. The size of the frames are 380 by 360 pixels. There is an excavator and a truck in this video and the excavator is digging, hauling and dumping the soil. The cycle of the mentioned process contains the body's rotation of about 180° for θ_1. Knowing that the excavator in this video can be watched from different views, six detectors for each part are trained to cover all views of the excavator. For each part and each view, the dimensions of the training windows are different. The sizes of the training windows for each detector and each part are provided in Table 5-4.

Table 5-4. Sizes of detection windows (pixels)

	Front-Left/ Front-Right	**Left/ Right**	**Back-Left/ Back-Right**
Dipper Detector	110×32	60×32	80×24
Boom Detector	62×32	32×50	33×32
Body Detector	32×37	32×42	32×47

Observing the video frames shows that the angles of the boom and the dipper (referred as θ_2 and θ_3 in Figure 6-2) are approximately in the range of 0° to 45°. The results in Table 5-5 show the achieved precision, recall, and accuracy for detecting the dipper, boom, and body. The detection accuracies are 79%, 88%, and 85% for the dipper, boom, and body, respectively.

Table 5-5. Results of parts detection on video frames in the first test

	Precision (%)	Recall (%)	Accuracy (%)
Dipper Detector	80	88	79
Boom Detector	85	86	88
Body Detector	85	89	85

Figure 5-22 shows the sample output of the detection algorithm after recognizing the parts and adjusting their bounding boxes. In addition to this construction site, a number of short videos from other construction sites were analyzed and similar results were achieved on those videos.

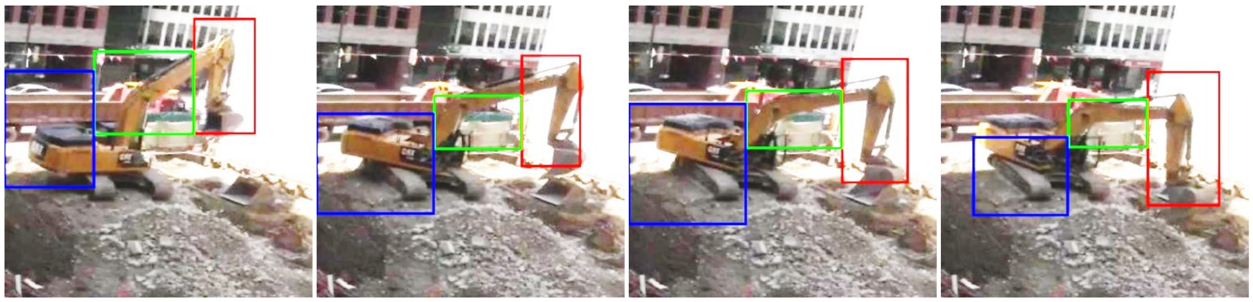


Figure 5-22. Output of the parts' detection module

5.4.3.2 Test on Stereo Cameras

The third test was applied on the cropped frames done in Section 5.4.2. The part detectors were applied on each frame to find the boom and the dipper of the excavator. The process is done simultaneously on the corresponding frames of both cameras. Moreover, γ is assumed to be 5% but further sensitivity analysis can be done for selecting this value.

In order to show the effect of cameras and GPS data fusion on the detection accuracies, two scenarios were tested. In the first scenario, the data were fused and in the second scenario the detectors were applied on the original frames without including the GPS data. The precision, recall, and accuracy of the detectors for the boom and the dipper are shown in Table 5-6. The comparison between fused and unfused data shows the improvement of the accuracies after fusion. The detection accuracies of both parts with fusion are close to each other for both cameras (i.e. 94% for the dipper and 96% for the boom).

Table 5-6. Results of boom and dipper detection on video frames in the second test

		Precision (%)		**Recall (%)**		**Accuracy (%)**	
		Left Camera	Right Camera	Left Camera	Right Camera	Left Camera	Right Camera
Boom	With Fusion	97	96	94	98	94	94
	No Fusion	46	69	33	66	24	51
Dipper	With Fusion	98	98	98	98	96	96
	No Fusion	52	69	58	80	50	72

5.5 Discussion

Calibrating two independent cameras with the long baseline installed at the high level was a challenging task. Using the checker boards with the squares' size of 50×50 *cm* address this challenge. However, since the size of 50×50 *cm* was the maximum size used in this research, it is not clear whether a larger size can improve the calibration results or not. There is always a trade-off between the resolution of a photo and the maximum recognizable points far from the camera. As shown in Figure 5-23, it is assumed that the distance in reality between the points P_1 and P_2 is d_P while the distance between the camera and the two points is d_{OP}, the distance on the image plane between the projected point Q_1 and Q_2 is d_Q, and f is the focal length of the camera. In the calibration process, d_P is referred to the size of the squares' printed on the checker board. Depending on the specification of the camera, if d_P is so small and d_{OP} is so large that d_Q is smaller than a pixel on the image plane then calibrating the camera for the points further than d_{OP} from the camera and closer to each other than d_P is meaningless. To overcome the aforementioned problem, either d_Q should be increased or d_{OP} has to be reduced.

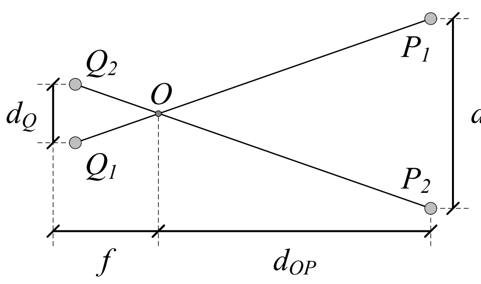


Figure 5-23. Relationship between points and their projections on the image plane

A rough comparison between the tests on the static images and the video shows that applying the detectors on both tests results in almost similar results. The only remaining concern is that when the parts appear smaller than the training sizes shown in Table 5-4, they cannot be detected and if the training sizes are decreased, then there is a chance of increasing the false detections.

5.6 **Summary and Conclusions**

Reviewing the literature showed the limitations and requirements of the available technologies for estimating the pose of the equipment and that CV-based technologies are great candidates for this purpose. However, relying on a single technology can result in missing some data and/or lowering the accuracy of estimations. Therefore, CV combined with a RTLS were proposed in this research to address these problems. Installing a GPS tracking device in the cabin of the excavator was helpful to narrow down the search scope of the CV-based detectors for recognizing the boom and dipper of the excavator.

Calibrating the stereo cameras with the long baseline was another challenge in this research. Using different checker boards with the different sizes of squares proved that lower calibration error can be achieved by using the board with larger squares. The average calibration error of each camera was and the average error of stereo calibration was less than 100 *mm* while the error was less than 500 *mm* with the confidence level of 95%. Projecting the GPS data on each camera view was another task done in this study and the locations of the excavator obtained from the GPS were correctly shown on each frame. The boom and the dipper were detected on the video frames with the accuracy of 94% for the boom detector and the accuracy of 96% for the dipper detector.

This chapter also investigated a new approach of detecting the excavator parts using the concept of synthetic images, followed by extracting the skeleton of the excavator. The results show that the synthetic images can play the role of real images captured from the construction site for training the excavator parts' detectors. A comprehensive image database for various parts of the excavator was developed for training the vision-based excavator parts' detectors. This method can save a lot of time during the training phase not only because it does not require the users to go to the construction sites and to capture the images of the excavator under various conditions but also because it does not require the users to annotate the training dataset manually. The best accuracies of the detectors trained by the synthetic images for the dipper, boom, and body were 79%, 88%, and 85%, respectively. The average accuracies of detecting the boom and the dipper on the stereo frames after fusion the cameras data with RTLS data were 94% and 96%.

The conclusions of this chapter are as follows: (1) Two regular surveillance cameras were successfully used as a stereo vision system on a large construction site and a guideline was developed for calibrating multiple cameras with long baseline using sensitivity analysis; (2) Data fusion was effectively used to integrate RTLS and video images to decrease the processing efforts for detecting excavator parts while increasing the detection accuracy by limiting the search scope for the detectors; (3) The parts' detectors were trained using the database of the auto-annotated synthetic images for the excavator's parts; and (4) The data fusion and part detection methods were tested in three case studies and the results proved the applicability of the proposed methods.

However, the proposed method has the following limitations: Detecting the bucket of the excavator is a challenge since the appearance of the buckets may change considerably by contacting the soil. One of the possible solutions can be the use of moving object detector as a preliminary step while searching for the bucket. This would help to limit the search space for the bucket detector. Furthermore, the frames are currently processed individually and this sometimes causes missing data. Object tracking algorithms (e.g. Kalman Filter or Particle Filter) can help the proposed method in handling the missing or noisy data. Another limitation of this research is that it assumes the location of the excavator on a flat terrain with no slope and it defines the virtual cylinder based on this assumption. However, in reality, the excavator may be on a slope; therefore the cylinder has to have the same angle as the slope of the terrain. To overcome this limitation, it is recommended to include an IMU as an additional source of data, which not only can address the aforementioned problem related to the slope but also can provide information about the direction of the excavator, which can help the CV algorithm to select the detector matching the actual relative direction of the equipment with respect to the cameras out of multiple detectors.

CHAPTER 6 SKELETON EXTRACTION AND 3D POSE ESTIMATION

6.1 Introduction

After fusing RTLS and CV data and applying the parts' detectors on the images received from two cameras that have overlapping FoV, this chapter mainly focuses on extracting the skeleton of the excavator and estimating the 3D pose of the excavator. Within the bounding box of each detected part provided by the methods in Chapter 5, the background of each part is subtracted. The foreground pixels belonging to each part are segmented and the skeleton of the excavator from the view of each camera is extracted. The 3D pose of the equipment is calculated by applying triangulation between the pixels coordinates of each joint of the skeleton using the cameras' parameters. Finally the 3D pose of the excavator is transferred into a game environment for safety measurements.

This chapter aims to achieve the following objectives: (1) to develop a clustering technique to subtract the background of the detected part and create a binary segment from the foreground; (2) to extract the skeleton of each part by applying morphological operations on the binary segments and find the joints of the excavator parts from each camera's view; (3) to estimate the 3D pose of the excavator by triangulating each pair of joints from two cameras' views; and (4) to transfer the estimated pose into the game engine and detect the potential collisions.

6.2 Process of Estimating 3D Pose of Excavator using Stereo Vision

As shown in Figure 6-1, the process of estimating the 3D pose of an excavator is divided into three steps. Step one focuses on subtracting the background, step two highlights the method of skeleton extraction, and step three explains the process of estimating the 3D pose and transferring the data into the game environment for safety monitoring.

An excavator has four degrees of freedom when it is stationary, which are represented by the angles shown in Figure 6-2 (θ_1, θ_2, θ_3, and θ_4). It is necessary to mention that in this research it is assumed that the center of the camera view is set to watch the gravity center of the excavator.

This is mainly defined to set the target object in the middle of the synthetic image. Moreover, this configuration is only set for creating the images from the 3D virtual model before the training phase while in the testing phase there is no special configuration. Therefore, the transformation between the coordinate systems of the camera and the excavator has rotational relationship in three dimensions while the translation between them can be considered as zero. Estimating all the mentioned orientations is required for providing the excavator skeleton in addition to the relative location of one of the four joints (considering the top-left corner of the original frame to be the origin point of the 2D image coordinate system). As mentioned in Section 6.1, since the current study is based on the video received from one camera, estimating the four angles in 3D real environment by analyzing only the 2D frames is not possible. Alternately, the 2D orientations of the skeletons for the boom and the dipper can be estimated upon analyzing the video frames from one camera.

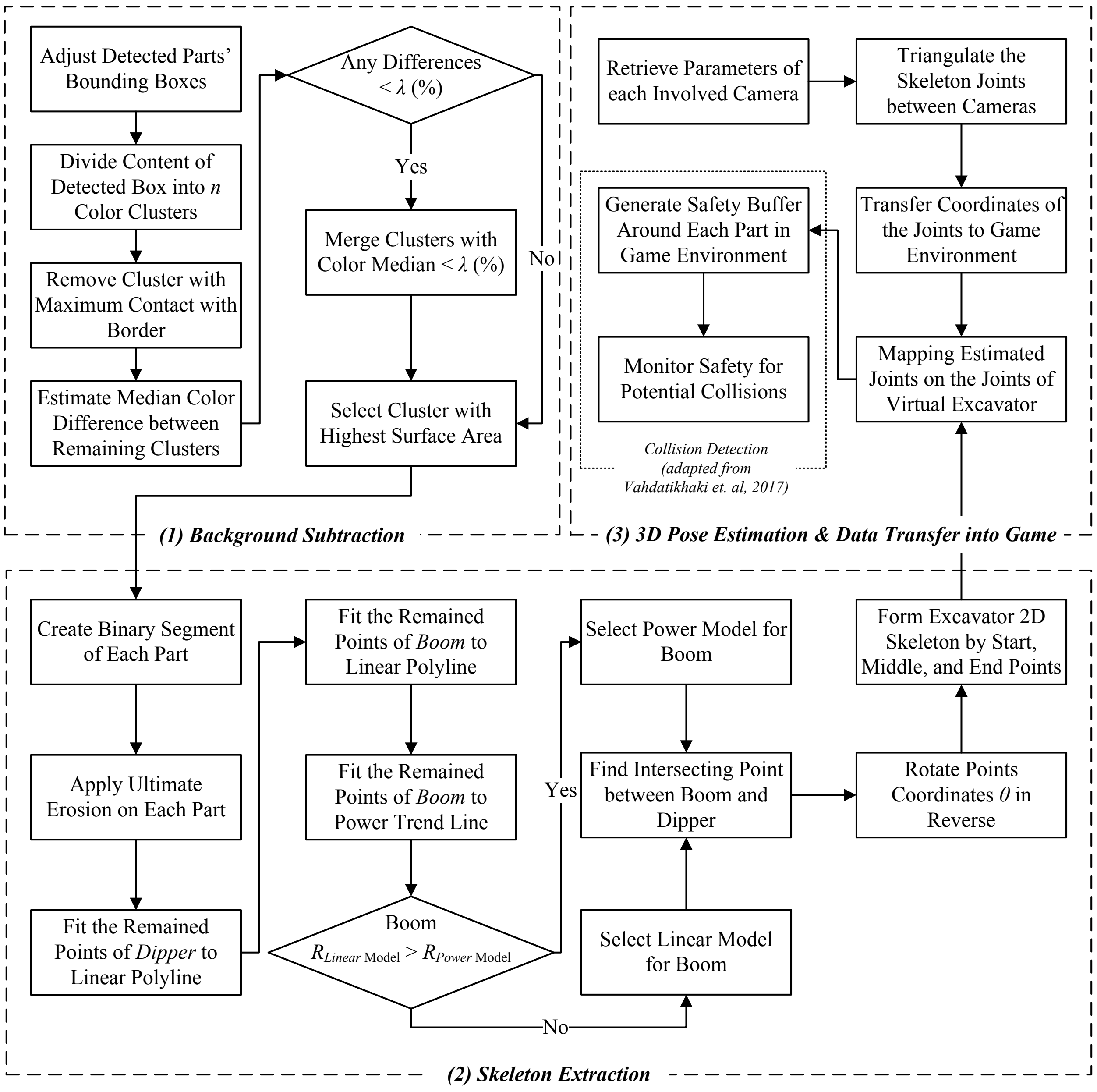

Adjust Detected Parts' Bounding Boxes
Divide Content of Detected Box into n Color Clusters
Remove Cluster with Maximum Contact with Border
Estimate Median Color Difference between Remaining Clusters
Any Differences < λ (%)
Yes
No
Merge Clusters with Color Median < λ (%)
Select Cluster with Highest Surface Area
(1) Background Subtraction
Retrieve Parameters of each Involved Camera
Triangulate the Skeleton Joints between Cameras
Transfer Coordinates of the Joints to Game Environment
Mapping Estimated Joints on the Joints of Virtual Excavator
Generate Safety Buffer Around Each Part in Game Environment
Monitor Safety for Potential Collisions
Collision Detection (adapted from Vahdatikhaki et. al, 2017)
(3) 3D Pose Estimation & Data Transfer into Game
Create Binary Segment of Each Part
Apply Ultimate Erosion on Each Part
Fit the Remained Points of Dipper to Linear Polyline
Fit the Remained Points of Boom to Linear Polyline
Fit the Remained Points of Boom to Power Trend Line
Boom R_Linear Model > R_Power Model
Yes
No
Select Power Model for Boom
Find Intersecting Point between Boom and Dipper
Select Linear Model for Boom
Rotate Points Coordinates θ in Reverse
Form Excavator 2D Skeleton by Start, Middle, and End Points
(2) Skeleton Extraction

Figure 6-1. Process of estimating 3D pose of excavator using stereo vision

The corresponding angles required for providing the 2D orientation (i.e. the projection of the angles on the plan of the image) are shown in Figure 6-3 (θ'_2, θ'_3, and θ'_4). Obviously, it is not meaningful to estimate θ_1 when the images cover more the long side view of the excavator and less the top view.

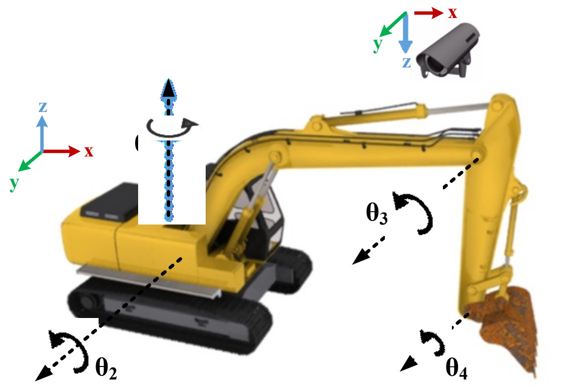


Figure 6-2. 3D kinematic representation of an excavator

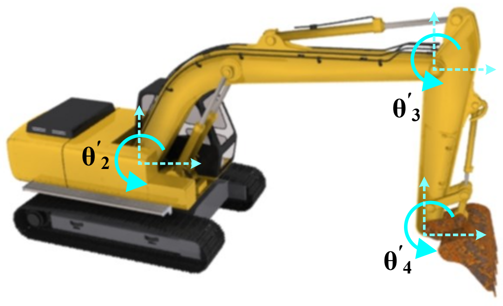
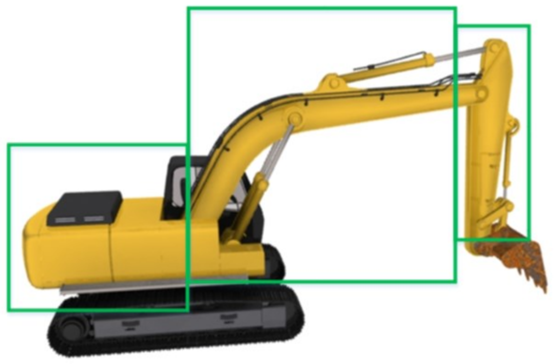
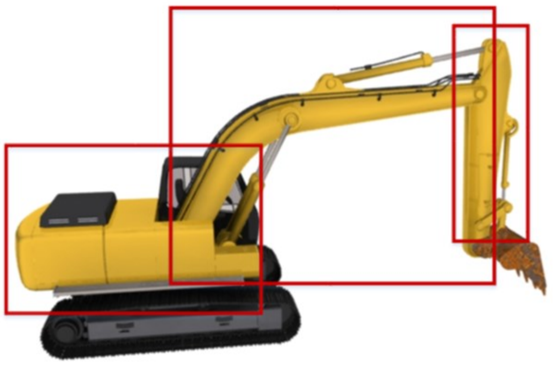


Figure 6-3. Image-based 2D kinematic representation of an excavator

6.2.1 Background Subtraction

In an ideal situation (Figure 6-4(a)) it is expected that each detected bounding box should surround its target object and be in contact with the bounding box of the adjacent parts (if applicable). However, this expectation does not happen in many situations even if each part is detected correctly. As shown in Figure 6-4(b) or (c), it is possible that the bounding boxes are larger or smaller than the expectation. The difference in the sizes causes serious problems for estimating the skeleton of the target part by wrongly considering a portion of the adjacent parts in the estimation process. For instance, as shown in Figure 6-4(b), the bounding box around the boom covers a portion of the dipper, which ultimately results in a wrong shape of boom's skeleton. Also, this problem may be produced when the bounding box is smaller than expectation. As a result, there is no contact on the borders of two connected parts. Addressing the aforementioned problems, the order of connections between every two parts are considered and checked to adjust the size of the bounding boxes to avoid unacceptable overlaps (i.e. Figure 6-4(b)) or gaps between two adjacent parts (i.e. Figure 6-4(c)).


(a) Ideal situation


(b) Bounding boxes with overlaps in between

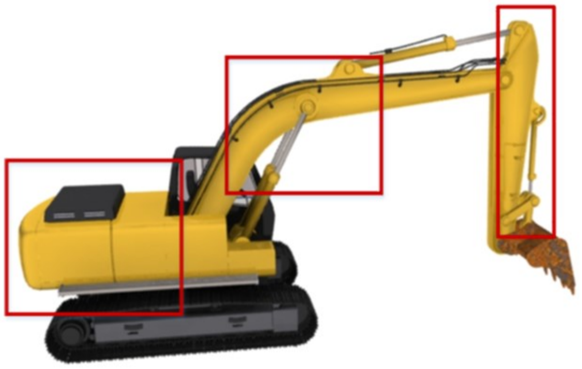

(c) Bounding boxes with gaps in between

Figure 6-4. Bounding boxes with/without gaps and overlaps

When facing an overlap between the dipper and the boom, the dipper has higher priority because it gives better detection results therefore the intersected side of the boom's bounding box is moved to reach to the intersected side of the dipper. In a scenario that the boom and the body have an overlap, the boom receives higher priority and the side of the body bounding box is moved to touch the intersected border of the boom with the body. When there is a gap between the dipper and the boom, the boom bounding box is extended to reach to the intersection side of the dipper. In cases that the gap is between the boom and the body, the body's bounding box is extended to touch the intersecting border of the boom with the body.

After detecting each part, it is required to find the skeleton of that part in order to estimate its orientation. The initial step for creating this skeleton is to remove the unnecessary pixels of the detected area known as background of the part. The pixels occupied by each part within the recognized bounding box should be segmented as one area. Knowing the pixel-wise location of the part area helps to find the orientation and position of the part in the image coordinate system.

As it was proposed in the research of Yuan et al (2016), the edge detection algorithm might be useful for extracting and segmenting excavator parts. However, the cluttered construction scenarios significantly affect its segmentation effectiveness. For example, the Sobel-filter and Dilate-filter were applied to the converted gray image (Aybar, 2006; Mathworks, 2014b). As shown in Figure 6-5(a) and Figure 6-5(b), the RGB image was converted into gray image, then the Sobel-filter is applied on the gray image (Figure 6-5(c)). Dilate-filter processes the result of the previous step and the new outcome (Figure 6-5(d)) was processed by the hole-filling filter (Figure 6-5(e)) (Soille, 2013).

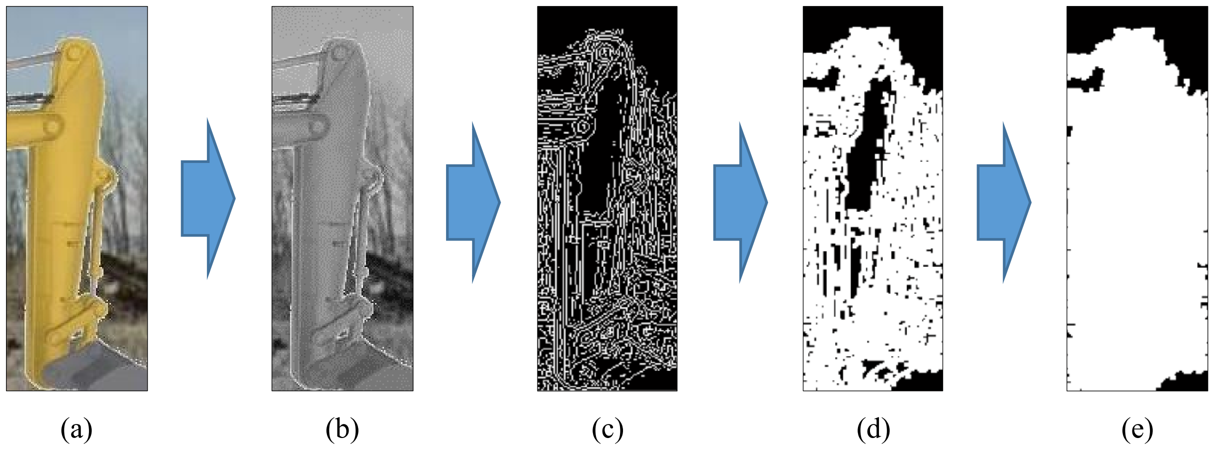


Figure 6-5. Part background subtraction based on edge detection

Unfortunately, the preliminary result was not reliable since applying the edge detection methods on the images with a cluttered background does not provide a precise segmented result. Since the edge detection does not provide reliable results, another segmentation technique is explored. This method, which is a color-based segmentation, uses *k*-Means clustering technique (Lloyd, 1982). The first step in this method is to convert the RGB color space to CIE L*a*b* color space, which is visually more distinguishable (Figure 6-6). CIE L*a*b* refers to the color space where *L* stands for lightness and *a* and *b* for the color-opponents green-red and blue-yellow (Hunter, 1958). Then, *k*-means clustering for two clusters is applied on the converted color space to divide the image into two separate areas. The reason that two clusters are considered is based on the assumption that within any detected bounding box there is one excavator part as foreground and the rest of the pixels are part of the background.

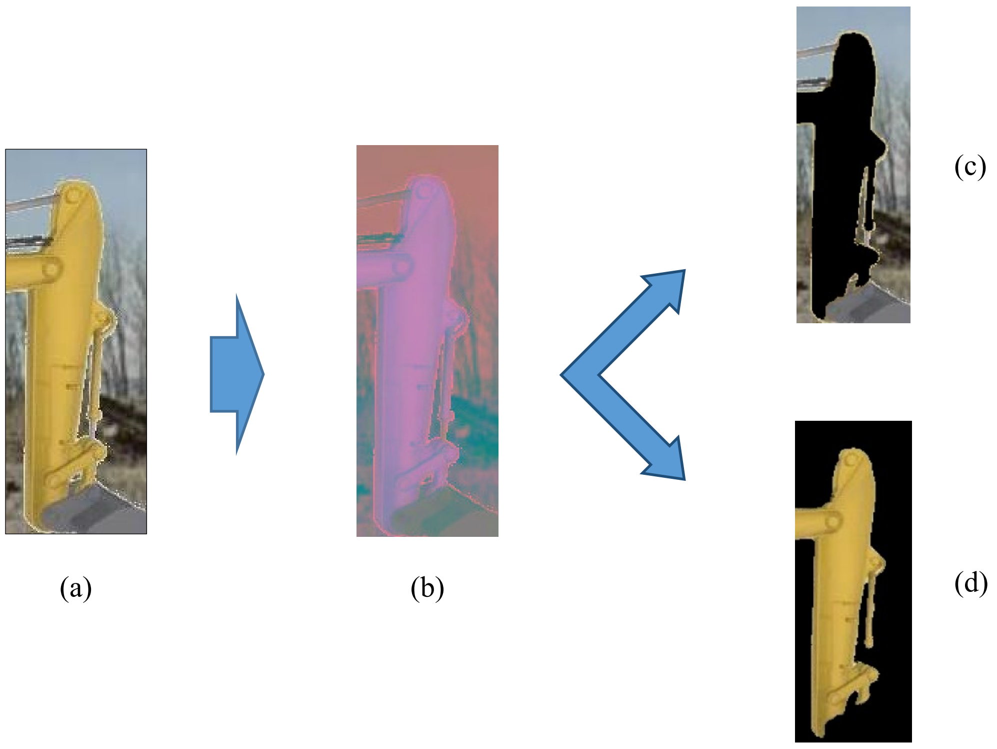


Figure 6-6. Part background subtraction using *k*-Means clustering

This process is repeated for the dipper, boom, and body. Knowing that there are two clusters for each part, the algorithm has to accept one cluster for each part and reject the other. Reviewing the specifications of the different excavator's manufacturers shows that usually the dipper and boom have the same color (e.g. both are yellow, orange, black, etc.) but the body can be in a different color (e.g. Volvo).

Observing the content of the detected bounding boxes shows that their backgrounds have the maximum contact with the borders of the bounding boxes (Figure 6-7(b)) compared to their foreground areas (Figure 6-7(a)). This rule can be applied on the clustering results of dipper and boom. In an ideal situation, the excavator part with color unity is distinguishable from the background but in reality the foreground cluster may include areas other than the target part, which need to be removed as much as possible. To address this problem, the morphological operations (Soille, 1999; Soille, 2013) are used to segment the content of the foreground cluster and to select the segment with the largest area as the input of the skeleton estimation algorithm.

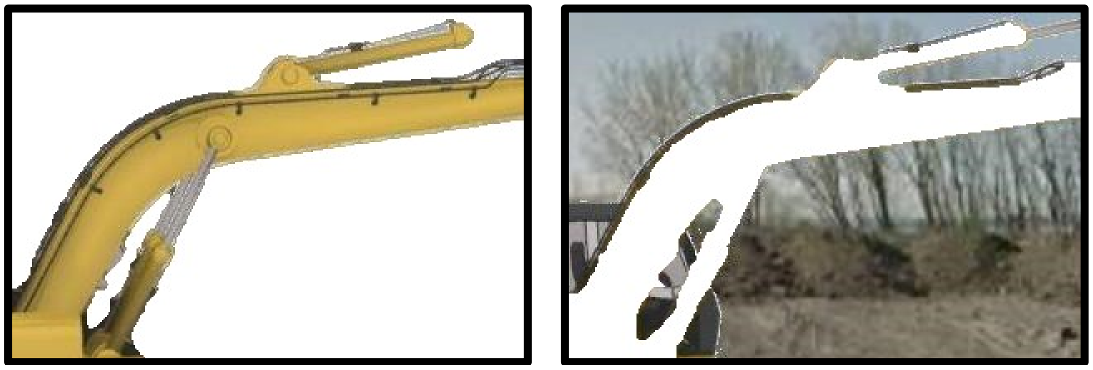

(a) Foreground (b) Background

Figure 6-7. Correct cluster selection

As shown in Figure 6-8, the content of each detected box has to be divided into two clusters based on the CIE L*a*b* color space, one as foreground and the other one as background. The content of the detected box is divided into two clusters, which have the farthest distance between their color median. As a result, the pixels with the colors close to the median color of the shiny object are considered as one cluster and the remaining pixels go for the second cluster. Therefore, when the color of the parts are not shiny enough to be considered under the first cluster, they will be considered as part of the second cluster, which is named as background.

This limitation happens when there is another object with bright color in addition to the part and the background resulting in some difficulties in subtracting the background correctly. For example, as shown in Figure 6-9, the area surrounded by the green line belongs to the boom and the rest should be subtracted as background. However, since the letters written on the boom, the worker safety vest and light blue pant look brighter than the other objects in this scene, each of them may take the winner cluster for itself.

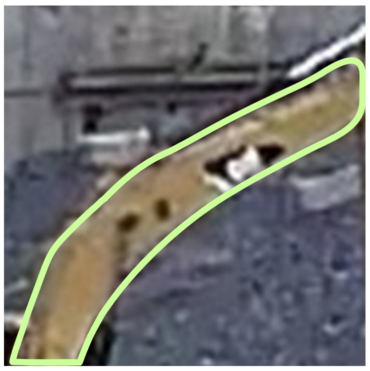


Figure 6-8. Single recognizable foreground

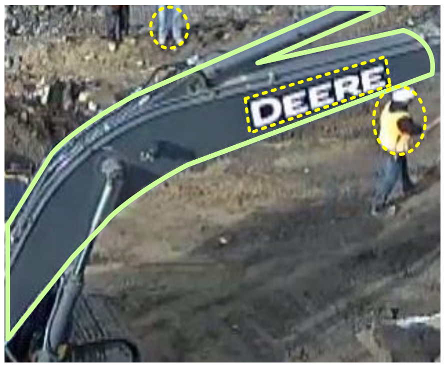


Figure 6-9. Multiple recognizable foregrounds

Assuming that these confusing objects are most probably smaller than the excavator parts, in this research it is proposed to add another constraint to the clustering algorithm to tackle this problem. This constraint defined by increasing the number of clusters from two to a larger number *n* (e.g. 10). The number of clusters can be further studied by applying a sensitivity analysis. The content of the detected box is divided into *n* clusters with *n* median colors. The cluster with the largest contact with the border of the bounding box is removed as the main background then the differences between the color median of the remaining clusters are calculated. The clusters with differences lower than λ% are merged together. This threshold adds an additional constraint. Moreover, it helps to avoid false division of one object into multiple clusters because of small contrasts. Finally, the cluster with the largest surface area among the remaining clusters is selected as foreground and it is converted to a binary image for skeleton extraction purpose.

6.2.2 Skeleton Extraction

Extracting the skeleton of the part is the next step toward estimating the orientation of each part. Skeletonizing or thinning the binary images is the process of removing the boundary pixels of a segmented object without allowing the segment to break apart (Mathworks, 2016; Lam et al., 1992; and Pratt, 1991). As shown in Figure 6-10, the boom of the excavator (Figure 6-10(a)) is segmented in black color (Figure 6-10(b)). The thinning algorithm is then applied on the segmented area to extract the skeleton of the boom using the morphological operations (Soille, 1999; Soille, 2013). The end point of the boom on the left side and right side are named *a* and *c*, respectively.

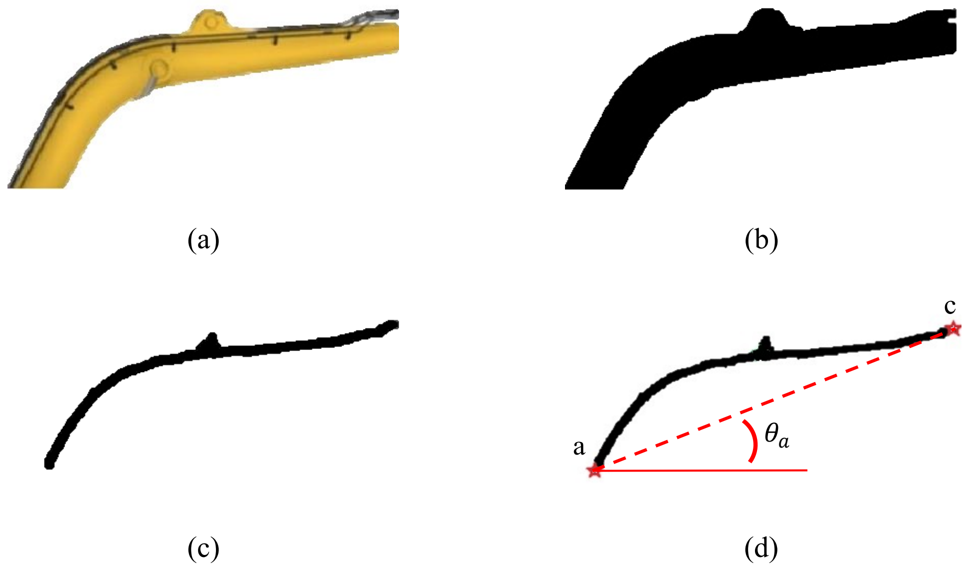
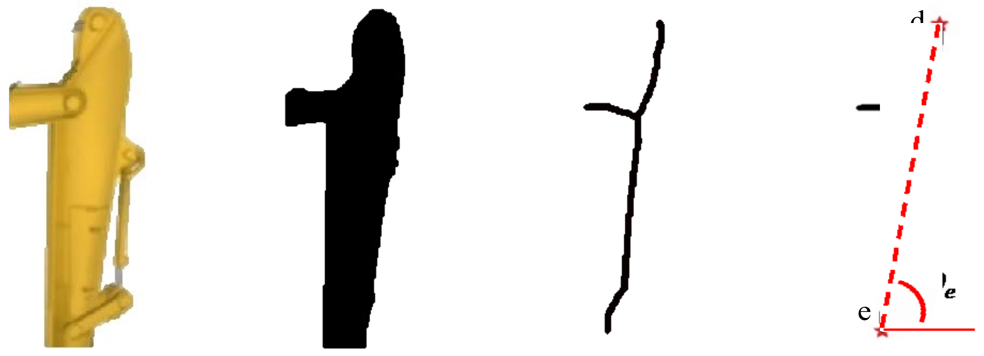


Figure 6-10. Process of boom orientation estimation

The same process is repeated for the dipper of the excavator to be skeletonized as shown in Figure 6-11. The end point of the dipper at the upper and lower sides are named d and e, respectively, while the branch point of the dipper (usually in contact with the boom) is named c. Knowing the start and end points for the boom and the dipper can be used to calculate the configuration of their skeletons.



(a) (b) (c) (d)

Figure 6-11. Process of dipper orientation estimation

Generally, the skeleton of the excavator mainly depends on the skeletons of the boom and the dipper. Considering the boom and the dipper together as the arm of excavator, the skeleton of this arm can be presented by θ_a and θ_c as shown in Figure 6-12. θ_a is the same as the orientation angle of the boom but θ_c is calculated using θ_e and the following Equation 6-1.

$$\theta_c = \theta_e - \theta_a \qquad \textbf{Equation 6-1}$$

However, this method suffers a lot from noisy images, especially those images of the excavator captured from a far distance since it directly relies on the binary pixels of the foreground that may be mixed with the background during the clustering phase. Therefore, in this research, extracting the skeleton of each part is performed by applying the ultimate erosion (Maintz, 2005) of the segment. The remaining points are then used for representing the skeleton of the part.

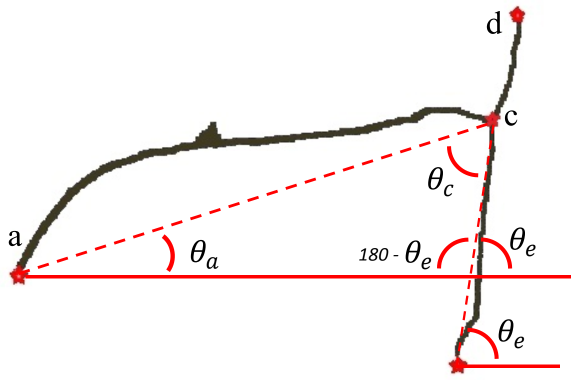
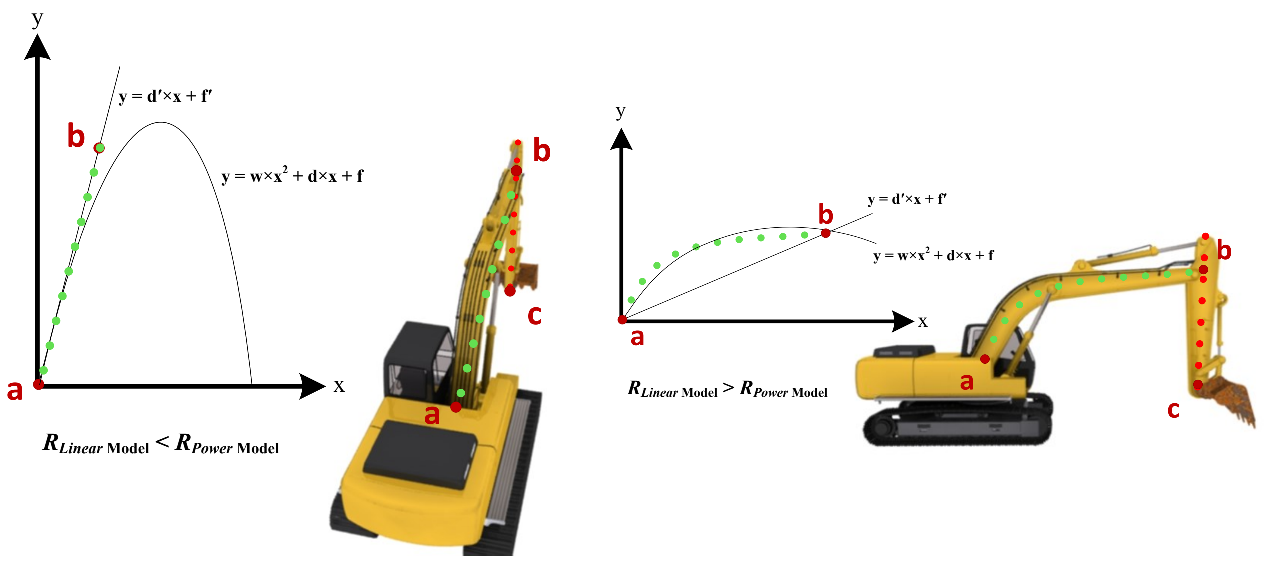


Figure 6-12. Excavator arm representative angles

The representation can be done by fitting the points to an individual curve for each part. Observing the appearance of the dipper shows that a linear polynomial is the best fitting trend line while for the boom there are two scenarios. In the first scenario, the boom is being watched from the back of the excavator (Figure 6-13(a)) and it is closer to a linear polyline with the residual value of $R_{Linear\ Model}$.


y
x
$y = d' \times x + f'$
$y = w \times x^2 + d \times x + f$
a
b
c
$R_{Linear\ Model} < R_{Power\ Model}$
$R_{Linear\ Model} > R_{Power\ Model}$

(a) Back view (b) Side v iew

Figure 6-13. Interest points of excavator skeleton

When the boom appears from the side of the excavator, a power trend line with the residual value of $R_{Power\ Model}$ is better fitted (Figure 6-13(b)). The proposed method compares $R_{Linear\ Model}$ and $R_{Power\ Model}$ and then selects the fitting trend line with the smaller residual value. Having one line for the dipper and one for the boom, the intersection of two curves is calculated to find the coordinates of the intersection point (point *b*). The other ends of the two curves are selected (i.e. point *a* for the boom and point *c* the dipper). In some cases where the boom or the dipper is occluded by each other (watching the excavator from back or front) the aforementioned process will be slightly different.

6.2.3 3D Pose Estimation and Transferring into Game Environment for Monitoring

As shown in Figure 6-14, the 3D pose of the excavator is calculated by triangulating the 2D coordinates of three interest points from each camera (e.g. points m_L, p_L, and n_L from the left view and m_R, p_R, and n_R from the right view) using the cameras' parameters. It should be mentioned that the upper body detector is not used since the body has only rotation around its *z* axis and this rotation is equal to the rotation of the boom around the same axis. Furthermore, the bucket detector is not applied on the frame as the preliminary tests in Section 5.4.3 showed that the detection results of the bucket were not reliable because the bucket is usually covered with soil.

Although the 3D pose estimation of the excavator is the main target of this research, the estimated pose should be further used for safety monitoring applications. As mentioned in Section 2.8, the excavator state identification and LAEW, proposed by Vahdatikhaki et al. (2017), are two suitable applications that can use the pose information for monitoring safety.

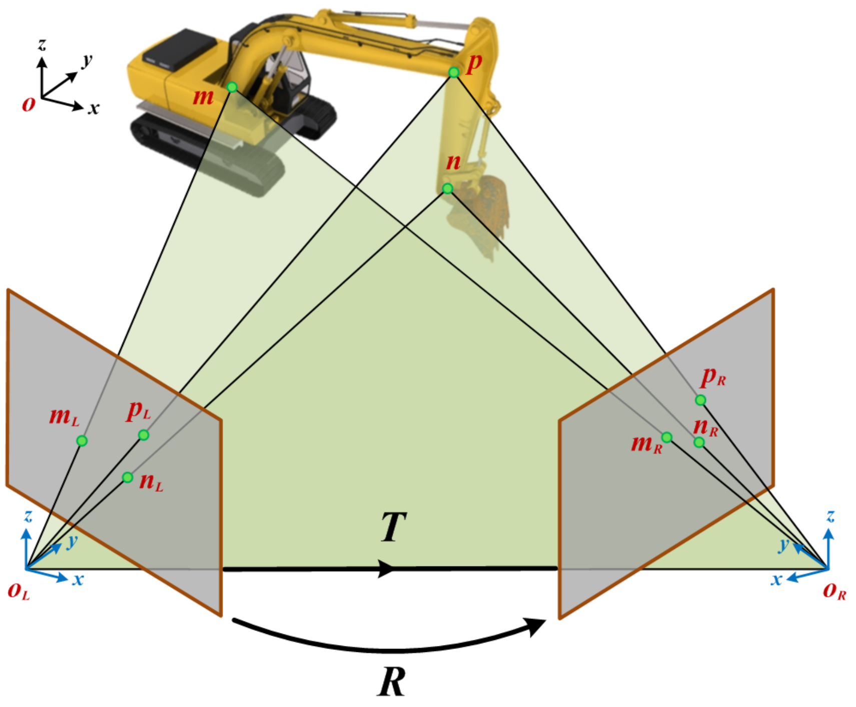


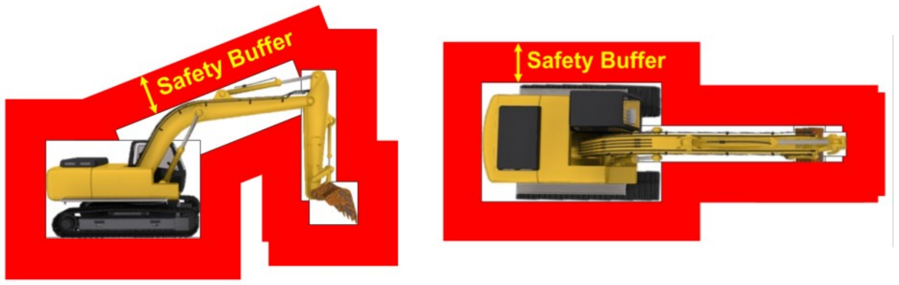
Figure 6-14. Joints triangulation between two views

Reviewing the work of Vahdatikhaki et al. (2017) shows that the supporting methods are developed in a game environment since the games can simulate various real-life scenarios that can be very costly and dangerous. In order to apply state identification and LAEW, the processes of parts recognition, skeleton extraction and 3D pose estimation can be developed inside the game engines directly. Moreover, all the location and dimensions data of the static and moving object can be replicated in the game engine. However, sometimes these processes might be better implemented in other platforms using special tools. In these situations, the estimated 3D poses should be transferred into the game engine. This data-transfer requires a stable communication between the game engine and the data processing platform. This communication can be established using 3-Way Handshake method and then the data is transferred under Transmission Control Protocol using Internet Protocol (TCP/IP) (Postel, 1981). This protocol is widely used by many tools, devices, and programs between the client and server. The client usually refers to the program which requests a service or resource and server refers to the program, which provides the service or resource to the client(s) (Bender, 2009).

Additionally, the 3D models of the equipment, the safety tools, and the temporary structure and offices can be imported into the game engine to consider the static objects on the site for evaluating the potential collisions with the moving objects.

6.2.4 Avoiding Collisions using 3D pose data

In this research, a simplified form of LAEW is adapted from the work of Vahdatikhaki et al. (2017) as proof-of-concept for monitoring the safety of the excavator using the fused RTLS and cameras data. As shown in Figure 6-15, a bounding box around each part of the excavator is created. A safety buffer, defined by the user, is added to each rectangle. This buffer should be defined based on the minimum required time for the operators or workers to avoid the collision after receiving a warning. Figure 6-15(a) shows the configuration of the buffer from the side-view of the excavator and Figure 6-15(a) shows it from the top-view.



(a) Side-view (b) Top-view

Figure 6-15. Defining a safety buffer

The rectangle-shape of the safety buffer is used for the representation in Figure 6-15 while a more realistic approach is to off-set the boundary of the excavator shape relative to the required size of the buffer. Additionally, a more detailed safety buffer discussed earlier as LAEW can be used. LAEW also considers the movement speed of each part and the low visibility risk of the operator for generating LAEW. Another concern while using the concept of safety buffers is the close proximity of the equipment working together. For instance, when the excavator is loading a truck, a close proximity between the excavator and the truck is unavoidable. Therefore, it is necessary to define the equipment that need to work in a close proximity. This would help to avoid false hazard detection.

6.3 Implementation and Validation

Since estimating the 3D pose of the excavator includes multiple steps, the implementation and validation in this chapter is divided into four steps in order to calculate the error at each step separately.

6.3.1 Background Subtraction

The algorithm developed in Matlab (Mathworks, 2016) starts by receiving the output of the part detection algorithm along with the relative location of the detected box to the top-left corner of the original frame. Out of the three detectors for the dipper, boom, and body, the results of the body's detector are not used for skeleton estimation since the body includes other parts with different color and shape (e.g. cabin and engine cab), which makes the skeleton estimation difficult for the proposed algorithm. As part of the clustering method explained in Section 6.2.1, the content of each box is converted from RGB to CIE L*a*b* color. In the first test, the outcome of the test on video frames discussed in Section 5.4.3 is used and background of each detected part is divided into two clusters and the cluster with the minimum contact with the border of the box is considered as foreground (cluster containing the part pixels). In this video there was no bright object on the scene except the yellow excavator. As shown in Table 6-1, background subtraction precision and recalls for the dipper and boom are 97% and 81%, respectively. The achieved accuracies for the dipper and boom are 94% and 69%, respectively.

Table 6-1. Results of foreground segmentation in the first test

	Precision (%)	**Recall (%)**	**Accuracy (%)**
Dipper	97	97	94
Boom	81	81	69

In the second test, the results of the test on stereo cameras introduced in Section 5.4.3.2 are investigated and the background of each detected part is subtracted using the two methods explained in Section 6.2.1. The parts of the excavator in this test are black and there are multiple bright moving object on the site. The first method divides the content of the detected bounding box into two clusters while the second method divides the content into 5 initial clusters. The results are shown and compared in Table 6-2. The achieved accuracies proved the claim explained earlier in Section 6.2.1 regarding the limitations of using two cluster in presence of bright objects.

Table 6-2. Results and comparison of background subtraction methods in the second test

	Two-Clusters Method		**n-Clusters Method**	
	Camera 1	Camera 2	Camera 1	Camera 2
Boom	76%	68%	80%	100%
Dipper	66%	48%	96%	100%

6.3.2 Skeleton Extraction

The developed process is applied on the detection results achieved in Section 6.3.1 to evaluate the performance of the proposed skeleton estimation algorithm (skeletons shown in Figure 6-16). The errors are measured by calculating the difference of the actual values of θ_a, θ_c, and θ_e and the values estimated by the proposed method. The actual values (ground-truth) are extracted manually.

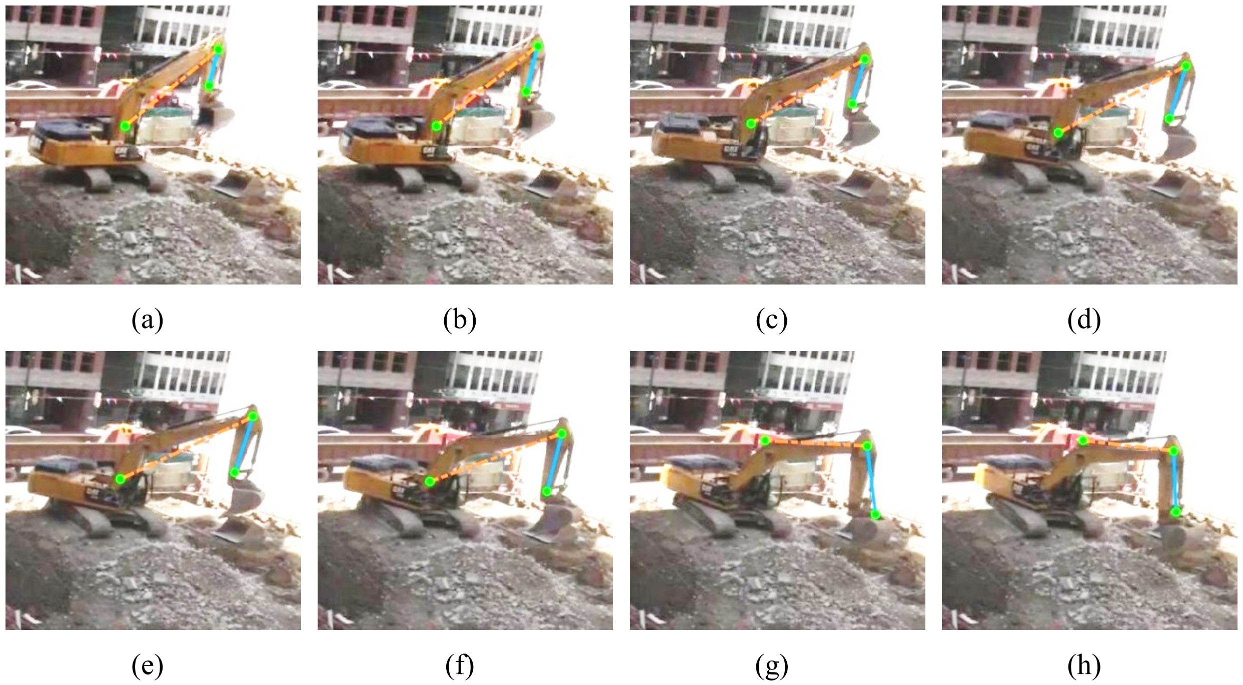

(a) (b) (c) (d) (e) (f) (g) (h)

Figure 6-16. Output of the skeleton estimation

The actual values (ground-truth) of the excavator skeleton is prepared by first extracting the pixel-wise coordinates of the points a, b, and c (shown in Figure 6-13) manually and then calculating the angles θ_a, θ_c, and θ_e.

As shown in Figure 6-17, the orientation of the dipper and the angle between the dipper and the boom were estimated with error less than 4° with confidence level of close to 95%. The orientation estimated for the boom reached to the same confidence level with the error of 8°. The maximum error of 18° was obtained in this test.

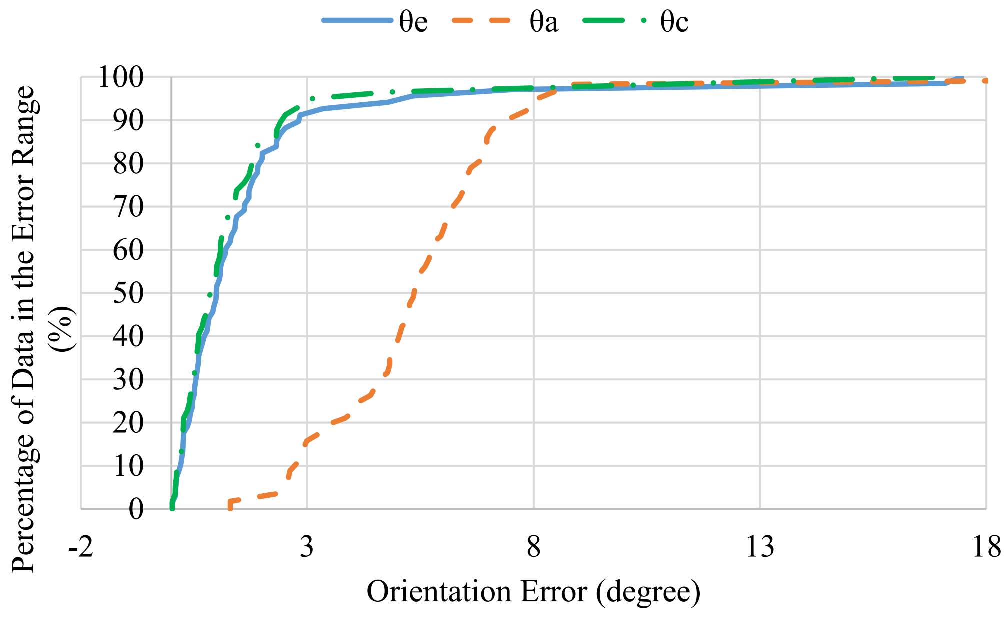


Figure 6-17. Results of the estimated parts orientations error in the first test

Moreover, the extracted parts were segmented and the skeleton of the boom and the dipper were extracted separately. The global points representing the skeleton of the dipper were fitted to a linear polynomial model while the global points of the boom may be fitted to a linear polynomial model or power model depending on the criteria explained in Section 6.2.2. After finding the intersection point of the boom and dipper using the aforementioned mathematical models and selecting the end point of each part, the 2D skeleton of the excavator can be represented by three points including the intersecting point. The results skeleton extraction based on n-clusters background subtraction method are shown in Table 6-3 and compared with results achieved by extracting the skeleton based on the two-clusters method. The 2D angles of θ_a and θ_b were measured and compared with the ground truth and the average errors with standard deviations in parentheses are shown in Table 6-3. The results shows that relying on n-clusters background subtraction method outperforms. Moreover, the example of skeleton estimation process can be found in Appendix I.

Table 6-3. Results and comparison of skeleton extraction methods in the second test

		Two-Clusters Method		*n*-Clusters Method	
		Camera 1	Camera 2	Camera 1	Camera 2
Boom - $\boldsymbol{\theta_a}$	μ	76.2	34.3	3.8	4.6
	σ	(45.8)	(7.3)	(6.1)	(7.2)
Dipper - $\boldsymbol{\theta_b}$	μ	28.7	14.7	5.3	15.6
	σ	(20.3)	(16.5)	(12.3)	(9.4)

6.3.3 Triangulating 2D Cameras Image Coordinates to 3D Coordinates

At time t, there were two 2D skeletons for the right and left cameras. Each skeleton had its corresponding points in the other camera image. Therefore, three pairs of points were triangulated using the stereo camera parameters estimated in Section 5.4.1. The outputs after using stereo parameters were three points representing the pose of the excavator in the 3D environment. In order to evaluate the performance of estimated poses, three metrics used in (Lundeen et al., 2015; Lundeen, et al., 2016; Feng et al., 2015; and Yuan et al., 2016) are considered. The first metric compares the actual lengths of the boom and the dipper (i.e. $\overline{ab}$ and $\overline{bc}$ in Figure 6-18), which are known for the excavator used in this experiment, with the estimated lengths after the triangulation of the 2D skeletons using the proposed method.

The pose of the excavator can be presented by the coordinates of one point (i.e. point a in Figure 6-18) and the angles of each part (θ_a and θ_b). The ground truth for this step was achieved by extracting the three interest points (i.e. *a*, *b*, and *c*) from each frame manually and triangulating between the left and the right views to find the 3D coordinates of the interest points. The second metric is the error of the location of point a and the calculated point using the proposed method compared with the location based on identifying this point manually. The third metric shows the difference between manually extracted angles of each part and the estimated angles of each part using the proposed method. For 100 frames, the average error and the standard deviation for each part is calculated.

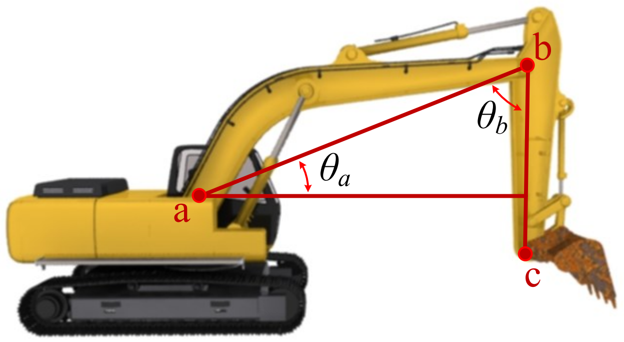


Figure 6-18. Pose representation notation

Table 6-4 shows the abovementioned averaged errors (μ) with standard deviations (σ) in parentheses. In Table 6-4, there are two groups of results (i.e. before and after). The first group includes the frames with falsely segmented parts (22 frames out of total of 100 frames) from the previous step and the second group were achieved by excluding the frames with false segmentation in either the right or left view. Therefore, the error of the second group are smaller than the first group. Focusing on the second group, the average error of estimated length of the dipper and the boom are 76 *cm* and 41 *cm*, respectively. The coordinates of point *a* is estimated with the average error of 60 *cm* while 2.9° and 4.2° are the average error of the estimated θ_a and θ_b.

Table 6-4. Pose estimation errors in the second test

		Absolute Length Error (*cm*)			**Location Error (*cm*)**			**Part Angle Error (degrees)**	
		Before	After		Before	After		Before	After
Dipper	μ	188	76				$\boldsymbol{\theta_a}$	4.7	2.9
	σ	(124)	(75)					(6.1)	(4.3)
Boom	μ	157	41	**Point *a***	1022	60	$\boldsymbol{\theta_b}$	6.2	4.2
	σ	(67)	(50)		(94)	(58)		(7.5)	(6.8)

6.3.4 Transferring Pose Data into Game Environment

As mentioned in Section 6.2.3, when the process of estimating the 3D pose of the excavators should be transferred into the 3D environment of a game engine for further processing related to safety. In the research, the methods leading to the estimation of 3D pose of the excavator were developed in Matlab. Therefore, it is necessary to integrate Matlab as the processing platform with the game engine. Due to free access and ease of use of Unity (Unity, 2017), it is used as the game engine in this research. This integration is developed through TCP/IP communication model as shown in Figure 6-19. As mentioned in Section 6.2.3, a communication channel is created between the data processing unit (in this research is Matlab) and Unity. A specific IP with free socket is defined in Matlab.

Matlab is defined as the server and Unity is considered as the client. There is one-time connection establishment between the server and the client. Afterward, data is sent to the game engine until there is no more data for transmission. At that point, the communication is terminated.

After the ready-to-use pose data of the excavators are transferred to Unity, the 3D model of the excavator (Figure 6-20 (a)) is imported into the game environment. In order to make the 3D environment developed in the game engine as close as possible to the reality, the 3D models of the static objects and the temporary equipment can be modeled in Autodesk 3Ds Maxs or Google SketchUp or can be purchased from online virtual stores such as Asset Store of Unity (Figure 6-20 (b)).

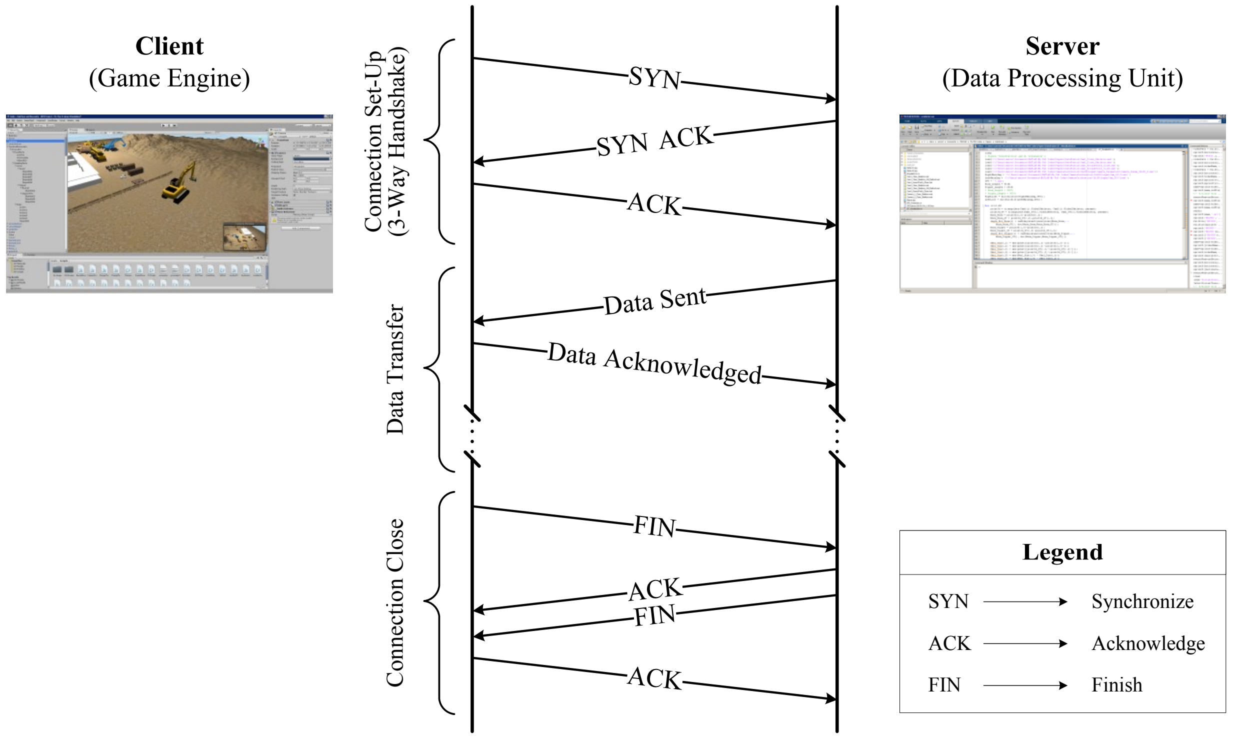


Figure 6-19. Communication model used in this research (adapted from Fall & Stevens, 2011)

Before starting to process the data, the socket is opened and it waits for Unity to send a starting signal. On the other side, a script in C# is developed in Unity and the address of the server and its open socket is stored in the script. By running the script, Unity sends a communication request with a synchronization code to Matlab and the communication is established and synchronized when Unity receives the acknowledgement of the accepted communication from Matlab. At this moment, Matlab can start processing the cameras and GPS data. Whenever the 3D pose at time *t* is ready, it will be sent to Unity for visualization and processing. This process continues untill there is no more data for transferring from Matlab to the Unity and Matlab sends a connection termination request to Unity.

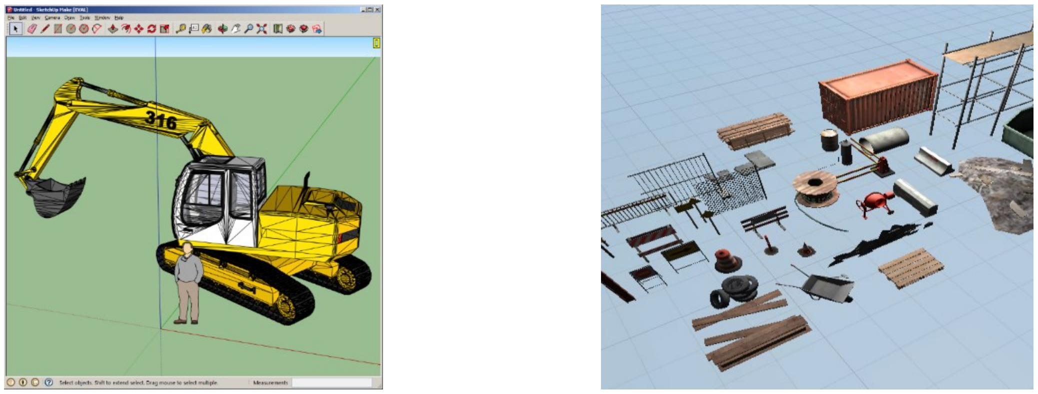
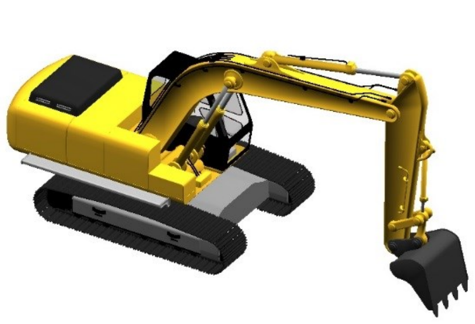

(a) Equipment model in Google Sketchup (b) Temporary site equipment

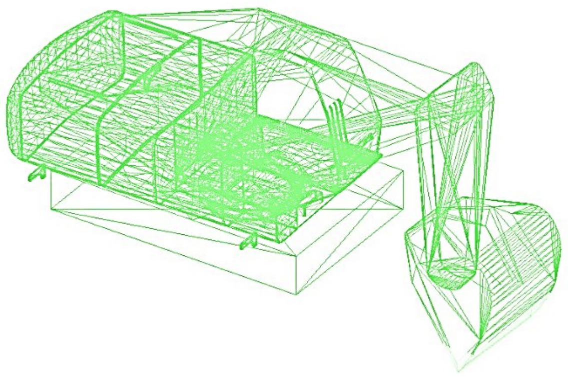
Figure 6-20. 3D models used in the game environment

By receiving the termination signal, the server acknowledges the request and returns it termination signal to the client. The communication channel is closed the moment that the server receives the termination acknowledgement from the client. There was no communication issue or packet lost during the data transmission since both the server and client were on the same computer. However, when each side of the communication are far from each other and use a wireless connection, there are chances for packet lost or data corruptions.

As mentioned in Section 6.2.3, the safety buffers should be created around each object. These buffers are known as colliders in Unity. They can be defined using primitive shapes (e.g. boxes, cones, cylinders, etc.), off-set meshes of the object, or finite cells generated based on the equipment risk maps proposed by Vahdatikhaki & Hammad (2015b).

This research uses the off-set meshes of the static and moving objects except for the workers. The pose of the workers can change very fast; therefore a capsule-shaped collider is considered as the safety buffer of the workers. A diameter of 1 *m* for the buffer is considered for off-setting the meshes and capsules colliders. It should be mentioned that selecting the required size of the buffer is not the interested of this research and 1 *m* is only used as an example. The example of the generated collider for the excavator shown in Figure 6-21(a) can be found in Figure 6-21(b). The integrated collider around the excavator is shown in Figure 6-21(c).


(a) 3D model


(b) Safety collider

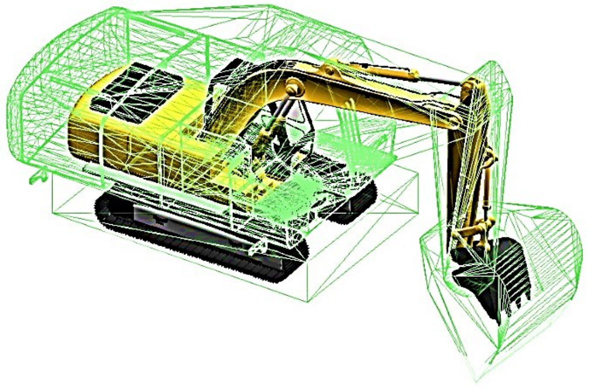
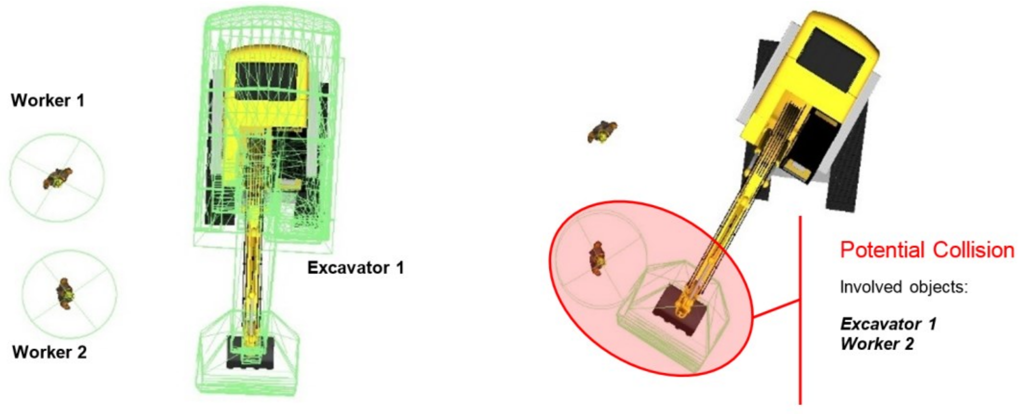

(c) Integrated safety collider around the excavator

Figure 6-21. Example of mesh collider used as safety buffer

In the following case study, the 3D pose data estimated in Section 6.3.3 are transferred from Matlab into Unity. The pose data is mapped to the three corresponding joints of the excavator model in Unity. Since it is not possible to create a real scene of accident, a hypothetical scenario is created in Unity to evaluate the performance of the system in detecting potential collisions.

In order to identify the potential collisions, a rule is defined within the game engine to continuously check if the exterior surface of two colliders from two different objects are touching each other. In this case, the game engine generates a warning for a potential danger. In the example shown in

Figure 6-22(a), two virtual workers are standing at a safe distance from the excavator and they are not in danger at the beginning of the excavator operation.



(a) No potential accident (b) Potential accident

Figure 6-22. Example scenario for detecting potential collision

While the excavator is rotating clockwise and approaching Worker 1, the collider of the bucket hits the collider of worker 1. As shown in Figure 6-22(b), the game catches the event and considers it as a potential accident. Additionally, the game is able to provide the objects involved in the potential accident. Some possibilities for notifying the involved workers and operators are: using the audio alarm, the vibration alarm (e.g. wristband vibrating alarm or smart phone), and/or the visual alarm (e.g. on the display installed in the cabin of the excavator).

On the tests for detecting the potential collisions, false collisions and false states were detected and identified, respectively, whenever there was a large error on the estimated pose. As expected, this shows the direct influence of the pose estimation errors on the collision detection and state identification results.

6.4 **Discussion**

Regarding the skeleton detection phase, it was found that the proposed method is reliable most of the time; however, the results are dependent to some factors. A vital factor for estimation the skeleton with high accuracy is a correct part detection result. In other words, when the true positive rate of a detector is low, it is not possible to achieve a good estimation of the skeleton. Another critical point is the problem of overlaps or gaps between the detected bounding boxes. Minimizing the overlap or gap may cause removing a portion of a part incorrectly. Focusing on estimating the 2D skeleton of the excavator includes a high error when the yaw angle of the excavator is large. Therefore, better accuracy can be reached when the boom's plane is parallel to the image plane.

Depending on the size of the excavator the errors provided in Figure 6-16 can have different interpretations. For instance, assuming that the reach length of the boom is $7m$ (hypotenuse of a right angle triangle), when θ_a has 2° error, the impact of this error on the opposite side of the triangle is approximately 24 *cm* (vertical displacement of the node c in Figure 6-12) while 10° error results in 122 *cm* vertical displacement.

Federal Standard 1037C (1996) defines real-time as "pertaining to the performance of a computation during the actual time that the related physical process occurs, in order that results of the computation can be used in guiding the physical process" while it defines near real-time as "pertaining to a delay introduced, by automated data processing, between the occurrence of an event and the use of the processed data". Knowing the aforementioned definitions, the process of estimating the skeleton of the excavator, which is computationally extensive, is considered as a NRT event; however, the recognition step is the most time-consuming step and it takes between one to two seconds. The remaining steps including background subtraction and skeleton estimation take approximately two seconds altogether. In order to use the proposed method for NRT purposes, the computation time should be reduced. In this research, the efficiency in developing the computer codes was not considered since the interest of this paper was to present a robust method. In the future studies, it is necessary to not only to make the increase the efficiency of the codes but also to divide the computation load between several GPUs to accelerate the speed of computation and save time.

The collision detection phase evaluated in this research was limited to the collisions between the workers and the excavators. In that scenario, the workers were not required to work within the safety buffer of the excavator. However, if the scenario includes the interaction between an excavator and a truck (e.g. process of loading a truck by an excavator), then the truck may enter into the safety buffer of the excavator and the system generates a false alarm. In such scenarios, it might be better to use LAEW (Vahdatikhaki & Hammad, 2015b) using the equipment risk maps. As mentioned in Section 2.2, the flexibility for increasing the accepted risk level in that method can be helpful in reducing the false alarms. Moreover, any error on the estimated pose can result in false state identification and collision detection.

6.5 Summary and Conclusions

In this chapter, the detected bounding box for each part was passed through the *k*-mean clustering algorithm to subtract the background and segment the part as a single foreground. Initially, *k* was set to two, which means there were two clusters, one for the background pixels and the other one for the foreground pixels. Later on, the background subtraction was improved by assigning a larger number of clusters (e.g. five in this research) instead of two clusters to overcome the false foreground selection in case that multiple shiny objects appeared within the detected bounding box. The average accuracies for subtracting the background of the dipper and boom For Camera 1 were 96% and 80%, respectively, while there was no false subtraction from Camera 2. The segmented foregrounds for each view were fitted to two curves for finding the intersecting point between the dipper and the boom. The skeleton of the excavator from each camera was represented by three points including the calculated intersecting point. Knowing the stereo cameras parameters and the pair of the excavator skeletons from both cameras, the 3D pose of the excavator was triangulated. The absolute error of the estimated length of the boom was 41 *cm* and for the dipper was 76 *cm*. The origin of the pose was estimated with 60 *cm* error and the relative angles errors for the boom and the dipper were 4.2˚ and 2.9˚, respectively. The estimated data were transferred into game environment using TCP/IP connection between Matlab and Unity. The 3D pose data in the game engine was used for detecting a potential collision, artificially created in the game.

The contributions of this chapter are: (1) A clustering technique (*k*-mean) was applied on the detected parts' boxes to subtract their background, and the skeleton of each part was extracted from the background using morphological operations; (2) The 2D location of each joint of the arm of the excavator from each view were calculated to provide the skeleton of the excavator; (3) The 3D pose of the excavator was estimated having the 2D skeletons of the excavator from each camera's views, then it was sent into game engine for further safety analysis; and (4) The skeleton extraction and 3D pose estimation methods were tested in a case study and the results proved the applicability of the proposed methods. Additionally, the 3D pose data was used for detecting a potential collision in the game engine.

There are some limitations of the current chapter that need further studies: (1) The current work focused on estimating the pose of a single excavator. However, there are many cases where multiple excavators work close to each other. Processing the data in congested areas brings new challenges for estimating the pose of the equipment, especially in case of partial or full occlusions. (2) Moreover, the proposed method performs at its best only when the pairs of skeletons are available from both cameras. In other words, when the data of one or more joints of the skeleton are not available temporarily (whether they were not detected or they were occluded in the specific frame(s)), the proposed framework failed to provide the pose. (3) Additionally, no dependency is defined for relating the pose of the same excavator in the subsequent frames. It means that the current framework detects the excavator as a new equipment in each frame. Applying tracking algorithms (e.g. Kalman Filter) can address the aforementioned issues by tracking each joint, which will help in case of temporary missing joints. Also, the estimated path of the joints will be smoother with fewer sudden jumps from one frame to the next due to the calculation errors. (4) More than two cameras with an overlapping view can be applied and the accuracy of the 3D pose estimation can be investigated using multiple cameras. (5) The 3D pose data of the excavator was used to evaluate a potential collision scenario. Further research is needed to distinguish the scenarios where close interaction between the excavators and other equipment is necessary (e.g. a truck loaded by an excavator).

CHAPTER 7 SUMMARY, CONTRIBUTIONS, AND FUTURE WORK

7.1 Summary of Research

This research covered the review of the related literature, the current research gaps, the overview of the proposed framework, and detailed explanation of the proposed methods followed by the case studies to validate and evaluate the applicability of the proposed framework.

In the proposed framework of this research, the method of generating the synthetic images of the construction equipment and their parts, and the auto-annotation method were introduced. These methods were able to reduce the required time and efforts for the positive sample preparation. Moreover, the methods improved the performance and accuracy of the object detectors trained by the synthetic images. Furthermore, the developed negative sample generator helped the detectors to decrease the number of false detections. The on-site cameras with the overlapping views were calibrated using a large checker board. The videos collected from the site combined with the RTLS data could narrow down the search scope for the excavator's parts detectors. The background of the detected parts were subtracted using *k*-mean clustering technique and the foreground images were segmented. The proposed skeleton extraction method was applied on the segmented foregrounds and the 2D skeleton of the excavator was derived from each camera's view knowing the kinematic relationship between all parts of the excavator. The 3D pose of the excavator was estimated using the stereo cameras parameters and a triangulation algorithm. The near real-time pose and location of the equipment were sent to the game environment using TCP/IP connection between the computation unit and the game engine. The 3D pose data in the game engine was used for detecting a potential collision, artificially created in the game. Additionally, the pose data was used for identifying the states of the excavator using a rule-based system.

7.2 Research Contributions and Conclusions

By doing this research, the following contributions were achieved:

(1) Developing a method for creating and annotating the synthetic images of the construction equipment and their parts using the equipment 3D models and the real images of the construction sites instead of taking the images of the equipment and annotating them manually, which is a very time consuming task. With regard to this contribution the following conclusions can be drawn:

- ✓ The rendered 3D model of the construction equipment were used to produce images with a single-color background that these images were annotated automatically using the proposed method.
- ✓ Synthetic images were produced by integrating the images of the 3D model with the desired background from different construction sites, different size of the objects, and different illumination conditions for the training phase.
- ✓ Synthetic images were annotated automatically without user interference based on the annotation results of the images with the single-color background.
- ✓ A large number of negative samples were produced from the image of different construction sites automatically while the target object is cut from these images.

Moreover, the synthetic images of the parts were partially used as negative sample by removing the target part from these images.

(2) Creating and training the HOG-based excavator’s parts detectors using the database of the synthetic images developed earlier and automatically produced negative samples from the other excavator parts in addition to the real images of different construction sites while the target object is cut from these. The following conclusions are achieved:

- ✓ The results of the HOG detectors using the proposed method were better than those obtained from a detector based on manual annotation of real images of construction equipment.
- ✓ The results showed that the proposed method is able to reduce the annotation time by more than 90% while the accuracy of the object recognition is improved by training more synthetic images.
- ✓ Part detectors were tested in three case studies and the results proved the applicability of the proposed methods.

(3) Developing a data fusion framework after calibration two regular surveillance cameras with the long baseline to integrate the RTLS data received from GPS with the video data from the cameras to decrease the processing efforts for detecting excavator parts while increasing the detection accuracy by limiting the search scope for the detectors. The following conclusions are drawn:

- ✓ Two regular surveillance cameras were successfully used as a stereo vision system on a large construction site and a guideline was developed for calibrating multiple cameras with long baseline using sensitivity analysis.

- ✓ Data fusion was effectively used to integrate RTLS and video images to decrease the processing efforts for detecting excavator parts while increasing the detection accuracy by limiting the search scope for the detectors.

- ✓ The data fusion method was tested in a case study and the results proved the applicability of the proposed method.

(4) Developing a clustering technique to subtract their background and extracting the 2D skeleton of the excavator in each camera's view and estimating the 3D pose of the excavator. With regard to this contribution the following conclusions can be drawn:

- ✓ *k*-mean clustering technique was applied on the detected parts' boxes to subtract their background and the skeleton of each part was extracted using morphological operations.

- ✓ The 2D location of each joint of the arm of the excavator from each view were calculated to provide the skeleton of the excavator.

- ✓ The 3D pose of the excavator was estimated by triangulating between the 2D skeletons of the excavator from each camera's views.

(5) Transferring the 3D pose data of the excavator to the game environment using TCP/IP connection and visualizing the near real-time pose of the excavator in the game engine for detecting the potential collisions. The following conclusions are achieved:

- ✓ The 3D pose data was sent into game engine for further safety analysis.
- ✓ The 3D pose data was used for detecting an artificial collision in the game engine and the states of the excavator was identified using a rule-based system.

7.3 Limitations and Future Work

While this research has successfully achieved its objectives, the following limitations still remains to be addressed in the future:

(1) The synthetic images created using the backgrounds with the random views of the cameras may sometimes look far from the reality. For instance, the background image could be taken at the street level but the image of the 3D model could be taken from the height of two meters above the street level; therefore a mismatch appears in the synthetic image. This limitation can be addressed in the future by:

- ✓ Using AR techniques can help by matching the view of the 3D model and the background (Furht, 2011).

(2) Weather conditions are another difficulty for creating this kind of synthetic images. Foggy and rainy weathers will cause the scene to look different than when the weather is clear. For the future studies, it is recommended to

- ✓ Take the advantages of the game engine for simulating the weather conditions while generating the synthetic images.

(3) Detecting the bucket of the excavator and estimating it pose are other challenges since the appearance of the buckets may change considerably by contacting with the soil. One of the potential solutions can be:

- ✓ The use of moving object detector as a preliminary step while searching for the bucket. This would help to limit the search space for the bucket detector.

(4) Although the part detectors achieved an average accuracy of 95% in Section 5.4.3.2, but the results of the test in Section 5.4.3.1 show that the detectors may perform lower than their best performances in some construction sites. It is highly recommended in the future to:

- ✓ Investigate the potential of the emerging object detection algorithms (e.g. Convolutional Neural Network based methods (Soltani et al., 2017)) to achieve more accurate detections.

(5) The scope of this research for estimating the 3D pose was limited to the excavators. The future studies should aim to:

- ✓ Expand the proposed framework to estimate the 3D pose of other construction equipment (e.g. loaders, dozers, etc.).

(6) The proposed framework performs at its best only when the pairs the skeletons are available from both cameras. In other words, when the data of one or more joints of the skeleton are not available temporarily (whether they were not detected or they were occluded in the specific frame(s)), the method fails to provide the pose. Processing the data in congested areas brings new challenges for estimating the pose of the equipment, especially in case of partial or full occlusions. Moreover, this research focused on estimating the 3D pose of one excavator. There are many scenarios where multiple excavators are working close to each other. Moreover, no dependency is defined for relating the pose of the same excavator in the consequent frames. It means that the current method detects the excavator as a new equipment in each frame. The future studies should focus on:

- ✓ Applying object trackers (e.g. Kalman Filter, Particle Filter, etc.) to predict the location of each missing joint of the excavator skeleton. Also, the estimated path of the joints will be smoother with fewer sudden jumps from one frame to the next due to the calculation errors.

- ✓ Using multiple object trackers to follow and record the motion of each part. Considering proximity-based and feature matching rules can be the solution for assigning the parts to their corresponding trackers.

  Investigating simulation-based optimization methods (Albahri & Hammad, 2017) for calculating camera coverage and planning the cameras' locations considering the configuration of the site depending on the schedule of the project.

(7) Another limitation of this research is that it assumes the location of the excavator on a flat ground with no slope and it defines the virtual cylinder based on this assumption. However, in reality, the excavator may be on a slope; therefore the cylinder has to have the same angle as the slope of the ground. It is highly recommended in the future to:

- ✓ Include IMU as an additional source of data, which not only can address the aforementioned problem related to the slope but also can provide information about the direction of the excavator, which can help the CV algorithm to select the detector matching the actual relative direction of the equipment out of multiple detectors.

(8) The accuracy of the proposed framework is limited to about a meter when the excavator is at a maximum distance of 50 *m* from the cameras. In other words, this accuracy may not be achieved when the excavator is further than 50 *m* from one camera. More accurate results can be achieved in the future by:

- ✓ Deploying cameras with higher resolution and using the checker boards with smaller square sizes.

- ✓ Evaluating the use of the checker boards with the larger square sizes, when the HD cameras similar to those that has been used in this research are deployed. In this scenario, accuracy can also be evaluated for the distances longer than 50 *m*.

- ✓ The accuracy of the 3D pose estimation can be investigated using more than two cameras with an overlapping view.

(9) The 3D pose data of the excavator was evaluated in an artificial collision scenario. In the future it is necessary to:

- ✓ Evaluate the effectiveness of detecting the potential collisions in real scenarios.

  Evaluate different scenarios where the close interaction between the excavators and other equipment are mandatory (e.g. a truck loaded by an excavator).

(10) The process of estimating the 3D pose of the excavator was utilized in the off-line mode. It means that the RTLS and cameras data were collected during the operation of the excavator but they were processed later on. Knowing that the achieved processing time of about four seconds is not enough for collision avoidance, it is highly recommended to consider:

- ✓ Applying HPC (e.g. using GPUs) to increase the processing speed of data for the near real-time applications.

REFERENCES

AEIOHS. (2011). *Construction Foreman Killed in Motor Vehicle Collision.* Alberta Employment and Immigration, Occupational Health and Safety.

Akinci, B., Fischer, M., Kunz, J., & Levitt, R. (2002). Representing work spaces generically in construction method models. *Journal of Construction Engineering and Management, 128*(4), 296-305.

Albahri, A. H., & Hammad, A. (2017). Simulation-Based Optimization of Surveillance Camera Types, Number, and Placement in Buildings Using BIM. *Journal of Computing in Civil Engineering, 31*(6).

Al-Jibouri, S., Mawdesley, M., Scott, D., & Gribble, S. J. (2005). The application of a simulation model and its effectiveness in teaching construction planning and control. *Computing in Civil Engineering, 179*(7).

Amr, A., & Mohamed, Y. (2011). Application of gaming engines in simulation driven visualization of construction operations. *The Electronic Journal of Information Technology in Construction, 16*, 23-38.

Andreopoulos, A., & Tsotsos, J. K. (2013). 50 Years of object recognition: Directions forward. *Computer Vision and Image Understanding, 117*(8), 827-891.

Atmel. (2013). *ATR0635 (ATMEL).* Retrieved April 23, 2013, from http://doc.chipfind.ru/atmel/atr0635.htm

Autodesk. (2015). Autodesk 3Ds Max. Retrieved from Autodesk.

AWCBC. (2014). *Number of Accepted Time-loss Injuries, by Industry and Jurisdiction.* Association of Workers' Compensation Boards of Canada (AWCBC), National Work Injury/Disease Statistics Program (NWISP).

Aybar, E. (2006). *Sobel edge detection method for matlab.* Anadolu University, Porsuk Vocational School, 26410.

Azar, E. R. (2013). Computer Vision-based Solution to Monitor Earth Material Loading Activities. *Doctoral dissertation*. Toronto, Ontario: University of Toronto.

Azar, E. R., & McCabe, B. (2011). Automated visual recognition of dump trucks in construction videos. *Journal of Computing in Civil Engineering, 26*(6), 769-781.

Azar, E. R., & McCabe, B. (2012a). Vision-based Recognition of Dirt Loading Cycles in Construction Sites. *Construction Research Congress* (pp. 1042-1051). West Lafayette, IN, USA: Purdue University.

Barnes, J., Rizos, C., Wang, J., Small, D., Voigt, G., & Gambale, N. (2003). Locata: the positioning technology of the future. *In Proceedings of the 6th International Symposium on Satellite Navigation Technology Including Mobile Positioning & Location Services.* Melbourne.

Barney, B. (2015). *Introduction to Parallel Computing.* Retrieved 05 25, 2015, from Lawrence Livermore National Laboratory: https://computing.llnl.gov/tutorials/parallel_comp/

Bay, H., Tuytelaars, T., & Van Gool, L. (2006). Surf: Speeded up robust features. *In 9th European Conference on Computer Vision* (pp. 404–417). Graz, Austria: Springer Berlin Heidelberg.

Bender, M. (2009). *MCTS Guide to Microsoft Windows Server 2008 Network Infrastructure Configuration.* Cengage Learning.

Blanz, V., Romdhani, S., & Vetter, T. (2002). Face identification across different poses and illuminations with a 3d morphable model. *In Proceeding of Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, (pp. 192-197).

Blanz, V., Schölkopf, B., Bülthoff, H. C., Burges, C., Vapnik, V., & Vetter, T. (1996). Comparison of view-based object recognition algorithms using realistic 3D models. *In Artificial Neural Networks—ICANN 96* (pp. 251-256). Springer Berlin Heidel.

Blough, R. (1983). *More construction for the money.* Construction Industry Cost Effectiveness Project, Summary Report, the Business Round Table.

Bouguet, J. Y. (2004). Camera Calibration Toolbox for Matlab. Intel Corporation.

Bradski., G. (2000). The opencv library. *Dr. Dobb's Journal of Software Tools, 25*(11), 120-126.

Brilakis, I., Park, M. W., & Jog, G. (2011). Automated vision tracking of project related entities. *Advanced Engineering Informatics, 25*(4), 713-724.

Brown, P. E. (1997). *Total Integration of Safety Professional into the Project Management Team.* CIB REPORT.

Bulusu, N., Heidemann, J., & Estrin, D. (2000). *GPS-less low cost outdoor localization for very small devices.* Technical, University of Southern California, Computer science department, LosAngles, CA.

Calfayan Construction. (2010). *Hazel House Excavation*. Retrieved March 01, 2017, from Calfayan Blog: https://calfayanconstruction.wordpress.com/2010/10/18/hazel-house-excavation/

Carr, P., Sheikh, Y., & Matthews, I. (2012). Monocular object detection using 3d geometric primitives. *In 2012 European Conference on Computer Vision* (pp. 864-878). Florence, Italy: Springer Berlin Heidelberg .

Charsky, D. (2010). From edutainment to serious games: A change in the use of game characteristics. *Games and Culture, 5*, 177-198.

Chavada, R. D., Kassem, M., Dawood, N. N., & Naji, K. K. (2012). A framework for construction workspace management: a serious game engine approach. *International Conference on Computing in Civil Engineering*, (pp. 57-64).

Chi, S., & Caldas, C. H. (2011). Image-based safety assessment: automated spatial safety risk identification of earthmoving and surface mining activities. *Journal of Construction Engineering and Management, 138*(3), 341-351.

Chiang, A. T., & Wang, Y. (2014). Human detection in fish-eye images using HOG-based detectors over rotated windows. *IEEE International Conference on Multimedia and Expo Workshops (ICMEW).*

Cho, Y. K., & Gai, M. (2013). Projection-recognition-projection method for automatic object recognition and registration for dynamic heavy equipment operations. *Journal of Computing in Civil Engineering, 28*(5), A4014002.

Churcher, D. W., & Alewani-Starr, G. M. (1997). *Incorporating construction health and safety into the design process.* CIB REPORT.

Cootes, T. F., Edwards, G. J., & Taylor, C. J. (2001). Active appearance models. *IEEE Transactions on pattern analysis and machine intelligence, 23*(6), 681-685.

Cootes, T. F., Taylor, C. J., Cooper, D. H., & Graham, J. (1995). Active shape models-their training and application. *Computer vision and image understanding, 61*(1), 38-59.

Cordova, F., & Brilakis, I. (2008). On-site 3D vision tracking of construction personnel. *Conference of the International Group for Lean Construction Management*, (pp. 809-820).

CPWR. (2013). *The Construction Chart Book: The US Construction Industry and Its Workers.* CPWR – The Center for Construction Research and Training.

Csurka, G., Dance, C., Fan, L., Willamowski, J., & Bray, C. (2004). Visual categorization with bags of keypoints. *In Workshop on statistical learning in computer vision* (pp. 1-2). ECCV.

Dalal, N., & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. *IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 886-893). San Diego, CA, USA.

Deshpande, A. A., & Huang, S. H. (2011). Simulation games in engineering education: A state-of-the-art review. *Computer Applications in Engineering Education, 19*(3), 399-410.

Dibitonto, M., Buonaiuto, A., Marcialis, G., Muntoni, D., Medaglia, C., & Roli, F. (2011). Fusion of radio and video localization for people tracking. *International Joint Conference on Ambient Intelligence* (pp. 258-263). Springer Berlin Heidelberg.

Djaouti, D., Alvarez, J., & Jessel, J. P. (2011). Classifying serious games: the G/P/S model. *Handbook of research on improving learning and motivation through educational games: Multidisciplinary approaches, 2*, 118-136.

Eagan, B. (2012). High Performance Computer Vision. *Master Thesis*. The University of Edinburgh.

E-digging. (2016). Excavator Loading Dump Truck - Excavator digging Deep Hole, Pond, Trench - Video #4. Youtube. Retrieved February 14, 2017, from https://www.youtube.com/watch?v=MDQdcUVi0XU

El-Omari, S., & Moselhi, O. (2011). Integrating automated data acquisition technologies for progress reporting of construction projects. *Automation in Construction, 20*(6), 699-705.

Fall, K. R., & Stevens, W. R. (2011). Chapter 13. TCP Connection Management. In *TCP/IP illustrated, volume 1: The protocols.* Addison-Wesley. Retrieved April 04, 2017, from Shichao's notes: https://notes.shichao.io/tcpv1/ch13/

Fan, H., Kim, H., AbouRizk, S., & Han, S. H. (2008). Decision support in construction equipment management using a nonparametric outlier mining algorithm. *Expert Systems with Applications, 34*(3), 1974-1982.

Farris, A. (2016). Reads (x,y,z) from a GoogleEarth kml file.

Federal Standard - 1037c. (1996). Telecommunications: Glossary of telecommunication terms. Institute for Telecommunications Sciences.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database.* Cambridge, Massachusetts: MIT Press.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 32*(9), 1627-1645.

Felzenszwalb, P., McAllester, D., & Ramanan, D. (2008). A discriminatively trained, multiscale, deformable part model. *IEEE Conference on Computer Vision and Pattern Recognition CVPR 2008* (pp. 1-8). IEEE.

Feng, C., Dong, S., Lundeen, K. M., Xiao, Y., & Kamat, V. R. (2015). Vision-based articulated machine pose estimation for excavation monitoring and guidance. *In Proceedings of International Symposium on Automation and Robotics in Construction (ISARC).*

Ferrer, B., Pomares, J. C., Irles, R., Espinosa, J., & Mas, D. (2013). Image processing for safety assessment in civil engineering. *Applied optics, 52*(18), 4385-4390.

Fraden, J. (2011). *Handbook of modern sensors.* Springer Science+ Business Media.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences, 55*(1), 119-139.

Friedlos, D. (2008). *RFID improves safety efficiency of Brisbane tunnel construction.* Retrieved April 30, 2013, from RFIDJournal: http://www.rfidjournal.com/articles/view?4137

Fullerton, C. E., Allread, B. S., & Teizer, J. (2009). Pro-active-real-time personnel warning system. *In Proceedings of the Construction Research Congress.* Seattle, Washington.

Furht, B. (2011). *Handbook of augmented reality* (Vol. 71). New York: Springer.

Gelb, A. (1974). *Applied optimal estimation.* MIT press.

Goedert, J., Cho, Y., Subramaniam, M., Guo, H., & Xiao, L. (2011). A framework for virtual interactive construction education (VICE). *Automation in Construction, 20*(1), 76-87.

Government of Alberta. (2011). *Occupational Fatalities in Alberta - 2001 to 2010.* Retrieved June 15, 2016, from https://work.alberta.ca/documents/Occupational-Fatalities-in-Alberta.pdf

Grau, D., Caldas, C. H., Haas, C. T., Goodrum, P. M., & Gong, J. (2009). Assessing the impact of materials tracking technologies on construction craft productivity. *Automation in Construction, 18*(7), (pp. 903-911).

Grauman, K., & Leibe, B. (2011). Visual object recognition (Synthesis Lectures on Artificial Intelligence and Machine Learning). San Rafael, CA: Morgan & Claypool Publishers.

Guo, S. (2002). Identification and resolution of workspace conflicts in building construction. *Journal of Construction engineering and management, 128*(4), 287-295.

Hall, D. L. (1992). *Mathematical Techniques in Multisensor Data Fusion.* Artech House.

Hammad, A., El Ammari, K., Langari, S. M., Vahdatikhaki, F., Soltani, M. .., AlBahnassi, H., & Paes, B. (2014). Simulating macro and micro path planning of excavation operations using game engine. *In Proceedings of the 2014 Winter Simulation Conference.* Savannah, GA.

Han, S., Lee, S., & Peña-Mora, F. (2011). Application of dimension reduction techniques for motion recognition: Construction worker behavior monitoring. *In 2011 ASCE International Workshop on Computing in Civil Engineering*, (pp. 19-22).

Han, S., Lee, S., & Peña-Mora, F. (2012). A Machine-Learning Classification Approach to Automatic Detection of Workers' Actions for Behavior-Based Safety Analysis. *In 2012 ASCE International Workshop on Computing in Civil Engineering*, (pp. 65-72).

Han, S., Lee, S., & Peña-Mora, F. (2014). Comparative Study of Motion Features for Similarity-Based Modeling and Classification of Unsafe Actions in Construction. *Journal of Computing in Civil Engineering, 28*(5), A4014005.

Hariharan, S. (2006). Improving Dynamic Decision Making Through RFID: A Partially Observable Markov Decision Process (POMDP) for RFID-enhanced warehouse search operations. *Master Thesis*. Oklahoma State University.

Harter, A., Hopper, A., Steggles, P., Ward, A., & Webster, P. (1999). The anatomy of a context-aware application. *The 5th Annual Joint ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM'99)* (pp. 59-68). New York, NY, USA: ACM.

Hartley, R. I. (1993). Euclidean reconstruction from uncalibrated views. *Joint European-US Workshop on Applications of Invariance in Computer Vision* (pp. 235-256). Springer Berlin Heidelberg.

Hartley, R., & Zisserman, A. (2003). Epipolar Geometry and the Fundamental Matrix. In *Multiple view geometry in computer vision* (pp. 239-261). Cambridge university press.

Hartley, R., Gupta, R., & Chang, T. (1992). Stereo from uncalibrated cameras. *In Proceeding 1992 IEEE international conference on image processing* (pp. 761–764). Champaign: IEEE .

Haupt, T. C. (2001). The performance approach to construction worker safety and health. *(Doctoral dissertation, University of Florida)*.

Heisele, B., Kim, G., & Meyer, A. J. (2009). Object recognition with 3D models. *In British Machine Vision Conference.*

Heisele, B., Kim, G., & Meyer, A. J. (2013). *Washington, DC: U.S. Patent and Trademark Office Patent No. 8,422,797.*

Helmus, M. (2007). Application fields of RFID in health safety and environment management. *RFID Eurasia, 2007 1st Annual* (pp. 1-3). IEEE.

Helmus, M., Kelm, A., Khazaee, M. J., & Laussat, L. (2011). Using Auto-ID Systems for Life Cycle Data Management of Personal Protective Equipment to Improve Occupational Health and Safety. *Modern Methods and Advances in Structural Engineering and Construction*, (pp. 455-460). Zürich.

Hightower, J., & Borriello, G. (2001). Location systems for ubiquitous computing. *IEEE Computer, 34*(8), (pp. 57-66).

Howarth, T., & Watson, P. (2009). *Construction Safety Management.* Wiley-Blackwell.

Howe, N. (2006). MATLAB Implementation of Skeletonization.

Huang, C. N., & Chan, C. T. (2011). ZigBee-based indoor location system by k-nearest neighbor algorithm with weighted RSSI. *Procedia Computer Science, 5*, 58-65.

Hunter, R. S. (1958). Photoelectric color difference meter. *Journal of the Optical Society of America (JOSA), 48*(12), 985-995. doi:http://dx.doi.org/10.1364/JOSA.48.000985

indus.ai. (2016). *Optimize your Cycle-Time using Intelligent Video Metrics*. (indus.ai) Retrieved March 20, 2017, from https://www.indus.ai/#home-section

Ji, Y., Biaz, S., Pandey, S., & Agrawal, P. (2006). ARIADNE: a dynamic indoor signal map construction and localization system. *The 4th international conference on Mobile systems, applications and services* (pp. 151-164). Uppsala: ACM.

Jiménez Ruiz, A. R., Seco Granja, F., Prieto Honorato, J. C., & Guevara Rosas, J. I. (2012). Accurate pedestrian indoor navigation by tightly coupling foot-mounted IMU and RFID measurements. *Instrumentation and Measurement, IEEE Transactions, 61*(1), (pp. 178-189).

John Deere. (2016). *290G LC*. Retrieved February 05, 2017, from http://www.deere.com/en_US/docs/construction/excavators/290g_lc/290G_LC_Web_02_21_11_final.pdf

Julesz, B. (1981). Textons, the elements of texture perception, and their interactions. *Nature, 290*(5802), 91-97.

Kandil, A., Hastak, M., & Dunston, P. S. (2012a). Vision-based motion detection for safety behavior analysis in construction. *Construction Research Congress 2012*, (pp. 1032-1041).

Kandil, A., Hastak, M., & Dunston, P. S. (2012b). Automated vision-based recognition of construction worker actions for building interior construction operations using RGBD cameras. *Construction Research Congress 2012*, (pp. 879-888).

Kashani, A., Owen, W. S., Himmelman, N., Lawrence, P. D., & Hall, R. A. (2010). Laser scanner-based end-effector tracking and joint variable extraction for heavy machinery. *The International Journal of Robotics Research*.

Kelm, A., & Laussat, L. (2010). Erweiterte digitale Zutritts- und PSA Kontrolle auf Baustellen. *Tagungsband des , Eigenverlag TU Wien.* Wien: Eigenverlag TU Wien.

Khoo, B. (2010). RFID from Tracking to the Internet of Things: A Review of Developments. *Proceedings of the IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing.*

Kittipanya-ngam, P., & Lung, E. H. (2010). HOG-based descriptors on rotation invariant human detection. *Asian Conference on Computer Vision.* Springer Berlin Heidelberg.

Kiviniemi, M., Sulankivi, K., Kahkonen, K., Makele, T., & Merivirta, M. (2011). *BIM-based Safety Management and Communication for Building Construction.* VTT research notes 2597. Retrieved March 07, 2012, from www.vtt.fi/inf/pdf/tiedotteet/2011/T2597.pdf

Kläser, A. (2007). Image annotation tool with bounding boxes. LEAR. Retrieved September 16, 2014, from http://lear.inrialpes.fr/people/klaeser/software_bbox_image_annotation

Kohlmeyer, A. (2012). *Introduction to High-Performance Computing.* Retrieved May 25, 2015, from International Centre for Theoritical Physics: http://portal.ictp.it/icts/hpc-appointments/HPC-Appointment-3.pdf

Kor, F., & Schneider, D. (2007). Annotation Tool. Dept. of Photogrammetry, University of Bonn. Retrieved September 16, 2014, from http://www.ipb.uni-bonn.de/html_pages_software/annotation-tool/publ/Korc-TR-IGG-P-2007-01.pdf

Krumm, J., Harris, S., Meyers, B., Brumitt, B., Hale, M., & Shafer, S. (2000). Multi-camera multi-person tracking for easyliving. *In Visual Surveillance, 2000. Proceedings. Third IEEE International Workshop*, (pp. 3-10).

Ku, K., & Mahabaleshwarkar, P. S. (2011). Building interactive modeling for construction education in virtual worlds. *Journal of Information Technology in Construction, 16*, 189-208.

KUBO-SK. (2015). New excavator Doosan DX380LC-5 loading jaw crusher. Youtube. Retrieved February 14, 2017, from https://www.youtube.com/watch?v=k5ehUV7nSSg

Kukelova, Z., Bujnak, M., & Pajdla, T. (2008). Polynomial Eigenvalue Solutions to the 5-pt and 6-pt Relative Pose Problems. In M. Everingham, & C. Needham (Ed.), *Proceedings of the British Machine Vision Conference. 2*, pp. 56.1-56.10. BMVA Press.

Kunkel, S., Bieber, R., Huang, M. S., & Vossiek, M. (2009). A concept for infrastructure independent localization and augmented reality visualization of RFID tags. *In IEEE MTT-S International Microwave Workshop on Wireless Sensing, Local Positioning, and RFID, IMWS 2009.* , (pp. 1-4).

Lam, L., Lee, S. W., & Suen, C. Y. (1992). Thinning methodologies-a comprehensive survey. *IEEE Transactions on pattern analysis and machine intelligence, 14*(9), 869-885.

Langari, S., & Hammad, A. (2015). Embedding Heuristic Rules in RRT Path Planning of Excavators. *In Proceedings of the International Symposium on Automation and Robotics in Construction (ISARC). 32*, pp. 1-8. Vilnius: Vilnius Gediminas Technical University, Department of Construction Economics & Property.

Lee, S., Ha, K. N., & Lee, K. C. (2006). A pyroelectric infrared sensor-based indoor location-aware system for the smart home. *Consumer Electronics, IEEE Transactions, 52*(4), 1311-1317.

Li, C., & Lee, S. (2011). Computer vision techniques for worker motion analysis to reduce musculoskeletal disorders in construction. *In 2011 ASCE International Workshop on Computing in Civil Engineering*, (pp. 19-22).

Li, L. H., W., G. I., & Tian, Q. (2003). Foreground object detection from videos containing complex background. *In Proceedings of the eleventh ACM international conference on Multimedia* (pp. 2-10). ACM.

Li, N., & Becerik-Gerber, B. (2011). Performance-based evaluation of RFID-based indoor location sensing solutions for the built environment. *Advanced Engineering Informatics, 25*(3), 535–546.

Liebelt, J., & Schmid, C. (2010). Multi-view object class detection with a 3d geometric model. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1688-1695). IEEE.

Lin, K. Y., Son, J. W., & Rojas, E. M. (2011). A pilot study of a 3D game environment for construction safety education. *Journal of Information Technology in Construction, 16*, 69-83.

Liu, H., Darabi, H., Banerjee, P., & Liu, J. (2007). Survey of wireless indoor positioning techniques and systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 37*(6), 1067-1080.

Liu, K., Skibbe, H., Schmidt, T., Blein, T., P. K., Brox, T., & Ronneberger, O. (2014). Rotation-invariant HOG descriptors using fourier analysis in polar and spherical coordinates. *International Journal of Computer Vision, 106*(3), 342-364.

Lloyd, S. P. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory, 28*(2), pp. 129-137.

Lowe, D. (1999). Object recognition from local scale-invariant features. *7th International Conference on Computer Vision*, *2*, pp. 1150–1157. Corfu, Greece.

Lundeen, K. M., Dong, S., Fredricks, N., Akula, M., Seo, J., & Kamat, V. R. (2016). Optical marker-based end effector pose estimation for articulated excavators. *Automation in Construction, 65*, 51-64.

Lundeen, K. M., Donga, S., Fredricksa, N., Akulaa, M., & Kamata, V. R. (2015). Electromechanical Development of a Low Cost End Effector Pose Estimation System for Articulated Excavators. *Proceedings of the International Symposium on Automation and Robotics in Construction (ISARC).* 32.

Luo, R. C., Yih, C.-C., & Su, K. L. (2002). Multisensor Fusion and Integration: Approaches, Applications, and Future Research Directions. *IEEE Sensors Journal, 2*(2), 107-119.

Luo, X., O'Brien, W. J., & Leite, F. (2013). Evaluating the Impact of Location-Aware Sensor Data Imperfections on Autonomous Jobsite Safety Monitoring. *Computing in Civil Engineering*, 573-580.

Maintz, T. (2005). Mathematical morphology. In *Digital and medical image processing.* Universiteit Utrecht.

Matas, J., & Obdrzalek, S. (2004). Object Recognition Methods Based on Transformation Covariant Features. *12th European Signal Processing Conference (EUSIPCO 2004).* Vienna, Austria.

Mathworks. (2014a). *Label Images for Classification Model Training*. Retrieved September 16, 2014, from http://www.mathworks.com/help/vision/ug/label-images-for-classification-model-training.html?refresh=true

Mathworks. (2014b). Dilate image. Retrieved September 16, 2014, from http://www.mathworks.com/help/images/ref/imdilate.html?searchHighlight=imdilate

Mathworks. (2016). *MATLAB*. Retrieved December 21, 2016, from Mathworks: http://www.mathworks.com/

Matthews, I., & Baker, S. (2004). Active appearance models revisited. *International Journal of Computer Vision, 2*, 135-164.

Memarzadeh, M., Golparvar-Fard, M., & Niebles, J. C. (2013). Automated 2D detection of construction equipment and workers from site video streams using histograms of oriented gradients and colors. *Automation in Construction, 32*, 24-37.

Meneses, E. (2015). *Introduction to High Performance Computing Systems (CS1645).* Retrieved May 25, 2015, from Univerity of Pittsburgh: http://coco.sam.pitt.edu/~emeneses/wp-content/uploads/2013/11/introduction6.pdf

Moeglein, M., & Krasner, N. (1998). An introduction to SnapTrack server-aided GPS technology. *In Proceedings of the 11th International Technical Meeting of the Satellite Division of The Institute of Navigation*, (pp. 333-342).

Monwar, M., Rezaei, S., & Prkachin, K. (2007). Eigenimage based pain expression recognition. *IAENG International Journal of Applied Mathematics, 2*(36), 1-6.

Motwani, J., Kumar, A., & Novakoski, M. (1995). Measuring construction productivity: a practical approach. *Work study, 44*(8), 18-20.

Mundy, J. L. (2006). Object recognition in the geometric era: A retrospective. *Toward category-level object recognition*, 3-28. Heidelberg: Springer Berlin.

Nehemiah, A. (2016). *Camera Calibration with MATLAB*. (Mathworks) Retrieved December 13, 2016, from https://www.mathworks.com/videos/camera-calibration-with-matlab-81233.html

NICS. (2011). *What is HPC?* Retrieved May 21, 2015, from University of Tennessee: https://www.nics.tennessee.edu/computing-resources/what-is-hpc

Nistér, D. (2004). An efficient solution to the five-point relative pose problem. *IEEE transactions on pattern analysis and machine intelligence, 26*(6), 756-770.

Oerter, J., Suddarth, W., Morhardt, M., Gehringer, J., McGinnis, M. L., Shockley, J., & Baysa, A. (2013). A System Architecture and Simulation Environment for Building Information Modeling in Virtual Worlds. *The Journal of Defense Modeling and Simulation, 11*(3), 205-210.

Okutomi, M., & Kanade, T. (1993). A multiple-baseline stereo. *IEEE Transactions on pattern analysis and machine intelligence, 15*(4), 353-363.

Oloufa, A. A., Ikeda, M., & Oda, H. (2003). Situational awareness of construction equipment using GPS, wireless and web technologies. *Automation in Construction, 12*(6), 737-748.

Pal, N. R., & Pal, S. K. (1993). A review on image segmentation techniques. *Pattern recognition,, 26*(9), 1277-1294.

Papapostolou, A., & Chaouchi, H. (2011). RFID-assisted indoor localization and the impact of interference on its performance. *Journal of Network and Computer Applications, 34*(3), 902-913.

Park, M. W., & Brilakis, I. (2012a). Enhancement of construction equipment detection in video frames by combining with tracking. *In International Workshop on Computing in Civil Engineering*, (pp. 17-20).

Park, M. W., & Brilakis, I. (2012b). Construction Worker Detection in Video Frames for Initializing Vision Trackers. *Automation in Construction, 28*, 15-25.

Park, M. W., Koch, C., & Brilakis, I. (2011). Three-dimensional tracking of construction resources using an on-site camera system. *Journal of Computing in Civil Engineering, 26*(4), 541-549.

Payet, N., & Todorovic, S. (2011). From contours to 3D object detection and pose estimation. *In 2011 IEEE International Conference on (ICCV)* (pp. 983-990). IEEE.

Postel, J. (1981). *Transmission control protocol.* Information Sciences Institute - University of Southern California.

Pradhan, A., Akinci, B., & Garrett Jr., J. H. (2009). Development and testing of inertial measurement system for indoor localization. *Proceedings of the 2009 ASCE International Workshop on Computing in Civil Engineering, 346.*

Pradhan, A., Akinci, B., & Haas, C. T. (2011). Formalisms for query capture and data source identification to support data fusion for construction productivity monitoring. *Automation in Construction, 20*(4), 389-398.

Pradhan, A., Ergen, E., & Akinci, B. (2009b). Technological assessment of radio frequency identification technology for indoor localization. *Journal of Computing in Civil Engineering, 23*(4), 230-238.

Pradhananga, N., & Teizer, J. (2013). Automatic spatio-temporal analysis of construction site equipment operations using GPS data. *Automation in Construction, 29*, 107-122.

Pratt, W. K. (1991). Image segmentation. *Digital Image Processing: PIKS Inside, Third Edition*, 551-587.

Priyantha, N. B., Chakraborty, A., & Balakrishnan, H. (2000). The cricket location-support system. *In Proceedings of the 6th annual international conference on Mobile computing and networking*, (pp. 32-43).

Pulli, K., Baksheev, A., Kornyakov, K., & Eruhimov, V. (2012). Real-time computer vision with OpenCV. *Communications of the ACM, 55*(6), 61-69.

QSTARZ. (2016). *GPS Lap Timer, BT-Q1000eX.* (QSTARZ) Retrieved January 30, 2017, from http://www.qstarz.com/Products/GPS%20Products/BT-Q1000EX-10HZ-F.html

Rafiee, M., Siddiqui, H., & Hammad, A. (2013). Improving Indoor Security Surveillance By Fusing Data From BIM, UWB And Video. *In Proceedings of the 30th International Symposium on Automation and Robotics in Construction.* Montreal.

Ramadevi, Y., Sridevi, T., Poornima, B., & Kalyani, B. (2010). Segmentation and object recognition using edge detection techniques. *International Journal of Computer Science & Information Technology, 2*(6), 153-161.

RASPBERRY PI FOUNDATION. (2017). *RASPBERRY PI PRODUCTS*. (RASPBERRY PI FOUNDATION) Retrieved January 30, 2017, from https://www.raspberrypi.org/

Razavi, S. N., & Haas, C. T. (2010). Multisensor data fusion for on-site materials tracking in construction. *Automation in Construction, 19*, 1037-1046.

Razavi, S. N., & Haas, C. T. (2011). Using reference RFID tags for calibrating the estimated locations of construction materials. *Automation in Construction, 20*(6), (pp. 677-685).

Rebolj, D., Babic, N. C., Magdic, A., Podbreznik, P., & Pšunder, M. (2008). Automated construction activity monitoring system. *Advanced Engineering Informatics, 22*, 493-503.

Reese, j., & Zaranek, S. (2011). *GPU Programming in MATLAB.* Mathworks.

Rodriguez, S. (2010). Experimental study on location tracking of construction resources using UWB for better productivity and safety. *Thesis (Masters)*. (A. Hammad, Ed.) Concordia University.

Roduit, N. (2006). JMicroVision: Image analysis toolbox for measuring and quantifying components of high-definition images. Retrieved July 2, 2006, from http://www.jmicrovision.com

Rüppel, U., & Schatz, K. (2011). Designing a BIM-based serious game for fire safety evacuation simulations. *Advanced Engineering Informatics, 25*(4), 600-611.

Russell, B. C., Torralba, A., Murphy, K. P., & Freeman, W. T. (2008). LabelMe: a database and web-based tool for image annotation. International journal of computer vision. *77*(1-3), 157-173.

Rybski, P. E., Huber, D., Morris, D. D., & Hoffman, R. (2010). Visual classification of coarse vehicle orientation using histogram of oriented gradients features. *In Intelligent Vehicles Symposium (IV)* (pp. 921-928). IEEE.

Safety Shield Systems. (2012). *Personnel Shield*. Retrieved January 01, 2012, from www.utilityshield.co.uk/personnelshield.html

Salvi, J., Armangué, X., & Batlle, J. (2002). A comparative review of camera calibrating methods with accuracy evaluation. *Pattern recognition, 35*(7), 1617-1635.

Schels, J., Liebelt, J., Schertler, K., & Lienhart, R. (2011). Building a semantic part-based object class detector from synthetic 3d models. *IEEE International Conference on Multimedia and Expo (ICME)* (pp. 1-6). IEEE.

Schon, S., & Bielenberg, O. (2008). On the capability of high sensitivity GPS for precise indoor positioning. *In 5th Workshop on Positioning, Navigation and Communication, WPNC 2008.* (pp. 121-127). IEEE.

Seo, J., Han, S., Lee, S., & Kim, H. (2015). Computer vision techniques for construction safety and health monitoring. *Advanced Engineering Informatics, 29*(2), 239-251.

Shahi, A., Cardona, J., Haas, C., West, J., & Caldwell, G. (2012). Activity-Based Data Fusion for the Automated Progress Tracking of Construction Projects. *Construction Research Congress.* ASCE.

Shahi, A., Safa, M., Haas, C. T., & West, J. S. (2014). Data Fusion Process Management for Automated Construction Progress Estimation. *Journal of Computing in Civil Engineering.*

Shoelson, A. (2012). SegmentTool: An Interactive GUI for Segmenting Images. Retrieved April 3, 2013, from http://www.mathworks.com/matlabcentral/fileexchange/38484-segmenttool--an-interactive-gui-for-segmenting-images

Shotton, J., Johnson, M., & Cipolla, R. (2008). Semantic texton forests for image categorization and segmentation. *In IEEE Conference on Computer vision and pattern recognition, CVPR* (pp. 1-8). London: IEEE.

Sim, R., & Dudek, G. (2003). Comparing image-based localization methods. *International Joint Conference on Artificial Intelligence* (pp. 1560-1562). LAWRENCE ERLBAUM ASSOCIATES LTD.

Simic, S. N., & Sastry, S. (2002). *Distributed localization in wireless ad hoc networks.* Technical , University of California, Department of Electrical Engineering and Computer Sciences, Berkeley, CA.

Skibniewski, M. J., & Jang, W. S. (2007). Localization technique for automated tracking of construction materials utilizing combined RF and ultrasound sensor interfaces. *In Proceedings of the 2007 International Workshop on Computing in Civil Engineering.*

Smith, D., & Singh, S. (2006). Approaches to Multisensor Data Fusion in Target Tracking: A Survey. *Knowledge and Data Engineering, IEEE Transactions on, 18*(12), 1696-1710.

Soille, P. (1999). Morphological Image Analysis: Principles and Applications. *Springer-Verlag*, 173-174.

Soille, P. (2013). Morphological image analysis: principles and applications. Springer Science & Business Media.

Soltani, M. M. (2010). Entwicklung eines Modells zur automatisierten Unfallanzeige in der Bauwirtschaft. *Bachelor Thesis*. Wuppertal, NRW, Germany: Bergische Universität Wuppertal.

Soltani, M., Karandish, S., Aly, W., Zhu, Z., & Hammad, A. (2017). Evaluating the Performance of Convolutional Neural Network for Classifying Equipment on Construction Sites. *the 34th International Symposium on Automation and Robotics in Construction.* Taipei, Taiwan.

Son, H., & Kim, C. (2012). Multiimaging sensor data fusion-based enhancement for 3d workspace representation for remote machine operation. *Journal of Construction Engineering and Management, 139*(4), 434-444.

Song, J., Haas, C., Caldas, C., Ergen, E., & Akinci, B. (2006). Automating the task of tracking the delivery and receipt of fabricated pipe spools in industrial projects. *Automation in Construction, 15*(2), (pp. 166-177).

Sorokin, A. (2009). Generic web-based annotation tool for Mechanical Turk (v0.1). University of Illinois at Urbana-Champaign. Retrieved September 16, 2014, from http://vision.cs.uiuc.edu/annotation/tools/annotation_instructions.html

Sorokin, A., & Forsyth, D. (2008). Utility data annotation with amazon mechanical turk. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops* (p. 820). Urbana: IEEE.

Statistic Brain. (2016). *Construction Industry Statistics.* Retrieved January 05, 2016, from http://www.statisticbrain.com/construction-industry-statistics/

Statistics Canada. (2016). *Construction (NAICS 23) : Gross domestic product (GDP).* Retrieved January 05, 2016, from https://www.ic.gc.ca/app/scr/sbms/sbb/cis/gdp.html?code=23&lang=eng#gdp2a

SUMITOMO. (2017). *Specifications*. (SUMITOMO (S.H.I.) Construction Machinery Co., Ltd) Retrieved February 14, 2017, from http://www.sumitomokenki.com/excavator/sh800lhd-5.html

Swain, M. J., & Ballard, D. H. (1991). Color indexing. *International journal of computer vision, 7*(1), 11-32.

Swedberg, C. (2007). *Tracking construction cranes in real-time.* Retrieved April 30, 2013, from RFIDJournal: http://www.rfidjournal.com/articles/view?3195

Tajeen, H., & Zhu, Z. (2014). Image dataset development for measuring construction equipment recognition performance. *Automation in Construction, 48*, 1-10.

Taneja, S., Akcamete, A., Akinci, B., Garrett, J., Soibelman, L., & East, E. W. (2010b). Analysis of three indoor localization technologies to support facility management field activities. *In Proceedings of the International Conference on Computing in Civil and Building Engineering.* Nottingham, UK.

Taneja, S., Akinci, B., Garrett, J. H., Soibelman, L., Ergen, E., Pradhan, A., & Anil, E. B. (2010a). Sensing and field data capture for construction and facility operations. *Journal of construction engineering and management, 137*(10), 870-881.

Tao, H., Sawhney, H. S., & Kumar, R. (2002). Object tracking with bayesian estimation of dynamic layer representations. *In IEEE Transactions on Pattern Analysis and Machine Intelligence, 24*(1), 75-89.

Taylor, J. R. (1997). *An introduction to error analysis: the study of uncertainties in physical measurements.* Univ. Science, Sausalito, CA.

Teizer, J., Caldas, C. H., & Haas, C. T. (2007). Real-time three-dimensional occupancy grid modeling for the detection and tracking of construction resources. *Journal of Construction Engineering and Management, 133*(11), 880-888.

Thiel, A., Uerikon, H., & Favey, E. (2007). *USA Patent No. US 7,170,445 B2.*

Tong, H., & Zekavat, S. (2007). A novel wireless local positioning system via a merger of DS-CDMA and beamforming: probability-of-detection performance analysis under array perturbations. *IEEE Transactions on Vehicular Technology, 56*(3), (pp. 1307-132).

Torr, P. H., & Zisserman, A. (2000). MLESAC: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding, 1*(78), 138-156.

Torralba, A., Murphy, K. P., & Freeman, W. T. (2004). Sharing features: efficient boosting procedures for multiclass object detection. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 762–769). IEEE.

Torralba, A., Murphy, K. P., & Freeman, W. T. (2007). Sharing visual features for multiclass and multiview object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 29*(5), 854-869.

Trimble. (2017). *AS450 Angle Sensor.* Retrieved April 09, 2017, from http://www.trimble.com/support_trl.aspx?Nav=Collection-58688&pt=AS450%20Angle%20Sensor

Trimble Navigation. (2013). SketchUp Pro 2013.

Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. *Journal of cognitive neuroscience, 3*(1), 71-86.

Unity. (2017). *Unity Game Engine 5.6*. Retrieved April 19, 2017, from https://unity3d.com/

Vahdatikhaki, F. (2015). Towards Smart Earthwork Sites Using Location-based Guidance and Multi-agent Systems. *Doctoral dissertation*. Concordia University.

Vahdatikhaki, F., & Hammad, A. (2014). Framework for near real-time simulation of earthmoving projects using location tracking technologies. *Automation in Construction, 42*, 50-67.

Vahdatikhaki, F., & Hammad, A. (2015a). Dynamic equipment workspace generation for improving earthwork safety using real-time location system. *Advanced Engineering Informatics, 29*(3), 459-471.

Vahdatikhaki, F., & Hammad, A. (2015b). Risk-based look-ahead workspace generation for earthwork equipment using near real-time simulation. *Automation in Construction, 58*, 207-220. doi:http://doi.org/10.1016/j.autcon.2015.07.019

Vahdatikhaki, F., Langari, S. M., Taher, A., El Ammari, K., & Hammad, A. (2017). Enhancing coordination and safety of earthwork equipment operations using Multi-Agent System. *Automation in Construction*. doi:10.1016/j.autcon.2017.04.008

van Diggelen, F. (2002). Indoor GPS theory & implementation. *Position Location and Navigation Symposium, 2002 IEEE* (pp. 240-247). IEEE.

Villamizar, M., Moreno-Noguer, F., Andrade-Cetto, J., & Sanfeliu, A. (2010). Efficient rotation invariant object detection using boosted random ferns. *Computer Vision and Pattern Recognition (CVPR)* (pp. 1038-1045). IEEE.

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *In Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2001* (p. 511). IEEE.

Von Ahn, L., & Dabbish, L. (2004). Labeling images with a computer game. *In Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 319-326). ACM.

Von Ahn, L., Liu, R., & Blum, M. (2006). Peekaboom: a game for locating objects in images. *In Proceedings of the SIGCHI conference on Human Factors in computing systems* (pp. 55-64). ACM.

Vossiek, M. W., Gulden, P., Weighardt, J., Hoffmann, C., & Heide, P. (2003). Wireless local positioning. *IEEE Microwave Magazine*, (pp. 77-86).

Wang, J., Pradhananga, N., & Teizer, J. (2014). Automatic Cave-in Safety Risk Identification in Construction Excavation. *In Construction Research Congress 2014* (pp. 130-139). Atlanta, Georgia: ASCE.

Want, R., Hopper, A., Falcao, V., & Gibbons, J. (1992). The active badge location system. *ACM Transactions on Information Systems (TOIS), 10*(1), (pp. 91-102).

WCB – Alberta. (2014). *Occupational Health and Safety Results 2013 - WCB-Alberta data.* OHS Policy and Program Development, Job, Skills, Training and Labour.

Weerasinghe, I. T., & Ruwanpura, J. Y. (2009). Automated data acquisition system to assess construction worker performance. *In Building a Sustainable Future, Construction Research Congress*, *1*, pp. 61-70. Seattle, WA, USA.

Wu, W., Yang, H., Chew, D. A., Yang, S. H., Gibb, A. G., & Li, Q. (2010). Towards an autonomous real-time tracking system of near-miss accidents on construction sites. *Automation in Construction, 19*(2), 134-141.

Xie, H., Tudoreanu, E., & Shi, W. (2006). Development of a Virtual Reality Safety-Training System for Construction Workers. *Digital library of construction informatics and information technology in civil engineering and construction*.

Yagi, J., Arai, E., & Arai, T. (2005). Parts and packets unification radio frequency identification (RFID) application for construction. *Automation in Construction, 14*(4), 477-490.

Yang, M. (2009). Object Recognition. *Encyclopedia of Database Systems*, 1936-1939. (L. L., & O. M. T., Eds.) Retrieved 15 August, 2013, from http://faculty.ucmerced.edu/mhyang/papers/object-recognition-chapter.pdf

Yuan, C., Li, S., & Cai, H. (2016). Vision-Based Excavator Detection and Tracking Using Hybrid Kinematic Shapes and Key Nodes. *Journal of Computing in Civil Engineering*, 04016038.

Zhang, C., & Hammad, A. (2011). Multiagent approach for real-time collision avoidance and path replanning for cranes. *Journal of Computing in Civil Engineering, 26*(6), 782-794.

Zhang, C., Hammad, A., & Rodriguez, S. (2012b). Crane pose estimation using UWB real-time location system. *Journal of Computing in Civil Engineering, 26*(5), 625-637.

Zhang, C., Hammad, A., Soltani, M., Setayeshgar, S., & Motamedi, A. (2012a). Dynamic virtual fences for improving workers safety using BIM and RTLS. *In 14th International Conference on Computing in Civil and Building Engineering.*

Zhang, S., & Yuan, F. (2006). RFID technique and its application in safety management system for people and vehicle in mine shafts. *Southern Metals, 1*, (pp. 12-14).

Zhang, S., Teizer, J., Lee, J. K., Eastman, C. M., & Venugopal, M. (2013). Building information modeling (BIM) and safety: Automatic safety checking of construction models and schedules. *Automation in Construction, 29*, 183-195.

Zhang, Z. (1999). Flexible camera calibration by viewing a plane from unknown orientations. *The Proceedings of the Seventh IEEE International Conference on Computer Vision. 1*, pp. 666-673. IEEE.

Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence, 22, 11*, 1330-1334.

Zhao, D., Lucas, J., & Thabet, W. (2009). Using virtual environments to support electrical safety awareness in construction. *In Proceedings of the Winter Simulation Conference (WSC).* IEEE.

Zhou, J., & Shi , J. (2009). Performance evaluation of object localization based on active radio frequency identification technology. *Computers in Industry, 60*(9), (pp. 669-676).

Zhu, Z., Park, M. W., Koch, C., Soltani, M., Hammad, A., & Davari, K. (2016). Predicting movements of onsite workers and mobile equipment for enhancing construction site safety. *Automation in Construction, 68*, 95-101.

Zou, J., & Kim, H. (2007). Using hue, saturation, and value color space for hydraulic excavator idle time analysis. *Journal of computing in civil engineering, 21*(4), 238-246.

APPENDICES

Appendix A. Matlab Code of Auto Annotation

```
clc
clear
% Margine
m = 20;
% Number of Conditions
CD = 3;
tic
MainBackImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\Conference\Excavator3DModel\BackGround\';
JPG = '*.jpg';
BackImList = dir(fullfile(MainBackImg,JPG));
% Number of Backgrounds
NoG = length(BackImList);

MainImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\Conference\SensivityAnalysis\8of16\+45\';
JPG = '*.jpg';
ImList = dir(fullfile(MainImg,JPG));
% Number of Images
NoI = length(ImList);

NewImgBG = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\Conference\SensivityAnalysis\8of16\+45\NewGenImg\';

parfor i=1:NoI
    imgfile = fullfile(MainImg,ImList(i).name);
    I1 = imread(imgfile);
    I = rgb2gray(I1);
    DipperMat(i).imageFilename = strrep (imgfile, '/', '\');
    figure, imshow(I1), title('original image');
    I = rgb2gray(imread('Dipper.jpg'));
    figure, imshow(I), title('gray image');

    se90 = strel('line', 3, 90);
    se0 = strel('line', 3, 0);

    [~, threshold] = edge(I, 'sobel');
    fudgeFactor = 0.1;
    BWs = edge(I,'sobel', threshold * fudgeFactor);
    figure, imshow(BWs), title('binary gradient mask');

    BWsdil = imdilate(BWs, [se90 se0]);
    figure, imshow(BWsdil), title('dilated gradient mask');

    BWdfill = imfill(BWsdil, 'holes');
    figure, imshow(BWdfill);title('binary image with filled holes');

    BWnobord = imclearborder(BWdfill, 4);
    figure, imshow(BWnobord), title('cleared border image');

    seD = strel('diamond',1);
    BWfinal = imerode(BWnobord,seD);
    BWfinal = imerode(BWfinal,seD);
    figure, imshow(BWfinal), title('segmented image');

    BWoutline = bwperim(BWfinal,8);
    Segout = I;
    Segout(BWoutline) = 100;
    figure, imshow(BWoutline), title(     );

    regdata = regionprops(BWfinal,'BoundingBox','Area');
    Area = cat(1, regdata.Area);
    BoundingBox = cat(1, regdata.BoundingBox);
    Best = find(Area(:) == max(Area(:)));
    bbox = BoundingBox(Best,:);
    DipperMat(i).objectBoundingBoxes = [bbox(1,1)-m bbox(1,2)-m bbox(1,3)+2*m
      bbox(1,4)+2*m];
    x = bbox(1, 1); y = bbox(1, 2); w = bbox(1, 3); h = bbox(1, 4);
    bboxPolygon = [x-m, y-m, x+w+m, y-m, x+w+m, y+h+m, x-m, y+h+m];
    videoFrame = insertShape(I1, 'Polygon', bboxPolygon,'Color','black');
    figure, imshow(videoFrame), title(' Bounding Box');
end

parfor i=1:NoG
    BG = [];
    BG = imread(fullfile(MainBackImg,BackImList(i).name));
    figure, imshow(LogZero)
    for j=1:NoI
        LogOne = [];
        LogOne = imread(fullfile(MainImg,ImList(j).name));
        figure, imshow(LogOne)
        I = rgb2gray(LogOne);
        figure, imshow(I)
        I2 = [];
        I2 = ~im2bw(I,0.95);
        I2 = repmat(I2,[1 1 3]);
        LogOne(I2==0)=0;
        figure, imshow(LogOne)
        I3 = [];
        I3 = ~I2;
        LogZero = BG;
        LogZero(I3==0)=0;
        figure, imshow(LogZero)
        for k=1:CD
            LogOne = imadjust(LogOne,[0; 1],[(k-1)*0.1; (1.1-(k*0.1))],(1.1-
(k*0.1)));
            figure, imshow(LogOne)
            LogOne(I2==0)=0;
            figure, imshow(LogOne)
            K=imadd(LogZero,LogOne);
            figure, imshow(K)
            New_k = k + (CD*(j-1)) + (NoI*CD*(i-1));
            imwrite(K,fullfile(NewImgBG,sprintf('IMAGE%04d.jpg',New_k)));
            NewTemp(i,j,k).imageFilename =
                  fullfile(NewImgBG,sprintf('IMAGE%04d.jpg',New_k));
            NewTemp(i,j,k).objectBoundingBoxes =
            DipperMat(j).objectBoundingBoxes
        end
    end
end
New_k=0;
for i=1:NoG
    for j=1:NoI
        for k=1:CD
            New_k = New_k+1;
            Temp(New_k).imageFilename = NewTemp(i,j,k).imageFilename;
            Temp(New_k).objectBoundingBoxes =
                  NewTemp(i,j,k).objectBoundingBoxes;
        end
    end
end
toc
DipperMat = [DipperMat, Temp];
save(fullfile(MainImg,sprintf('ROI.mat')),'DipperMat');
% 0 on background 1 on foreground
% I gray original image
% I2 Background with logical One on foreground
% I3 Background with logical Zero on foreground
```

Appendix B. Matlab Code of Multiple Scales Generator

```
clear
clc

% Minimum number of scale reduction
FD = 3;

MainBackImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\Conference\SensivityAnalysis\8of16\-45\Original\';
JPG = '*.jpg';
ImList = dir(fullfile(MainBackImg,JPG));
% Number of Backgrounds
NoG = length(ImList);

NewImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\Conference\SensivityAnalysis\8of16\-45\';
NB = 0;
for i=1:NoG
    Res_Sample = [];
    Res_Sample = imread(fullfile(MainBackImg,ImList(i).name));
    ImgData = imfinfo(fullfile(MainBackImg,ImList(i).name));
    W = ImgData.Width;
    H = ImgData.Height;
    NB = NB + 1;
    imwrite(Res_Sample,fullfile(NewImg,sprintf('Pyramid %04d.jpg',NB)));
    for j=1:FD
       Res_Sample = impyramid(Res_Sample, 'reduce');
       [M,N] = size(Res_Sample(:,:,1));
       New_Sample = padarray(Res_Sample,[floor((H-M)/2), ((W-N)/2)]);
       [M,N] = size(New_Sample(:,:,1));
       if H ~= M
           New_Sample = padarray(New_Sample,[H-M, 0],'post');
       elseif W~=N
           New_Sample = padarray(New_Sample,[0, W-N],'post');
       end
       New_Sample(New_Sample==0) = 255;
       NB = NB + 1;
       imwrite(New_Sample,fullfile(NewImg,sprintf('Pyramid %04d.jpg',NB)));
       New_Sample = [];
    end
end
```

Appendix C. Matlab Code of Negative Samples Generator

```
clc

% Minimum number of pixels for negative samples
FD = 100;
% Number of samples
NS = 100;
MainBackImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Hassan\LabTest09162014\Snapshots-Sep16-DataFusion\ClearShop\';
JPG = '*.jpg';
BackImList = dir(fullfile(MainBackImg,JPG));
% Number of Backgrounds
NoG = length(BackImList);

NegImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Hassan\LabTest09162014\Snapshots-Sep16-DataFusion\Negative\';

for i=1:NoG
    BG_Org = [];
    BG_Org = imread(fullfile(MainBackImg,BackImList(i).name));
    ImgData = imfinfo(fullfile(MainBackImg,BackImList(i).name));
    W = ImgData.Width;
    H = ImgData.Height;
    for j=1:NS
       New_Neg = [];
       X1 = randi([1 (W-(FD+1))],1,1);
       X2 = randi([(X1+FD) (W-1)],1,1);
       Y1 = randi([1 (H-(FD+1))],1,1);
       Y2 = randi([(Y1+FD) (H-1)],1,1);
       rect = [X1 Y1 (X2-X1) (Y2-Y1)];
       New_Neg = imcrop(BG_Org,rect);
       imwrite(New_Neg,fullfile(NegImg,sprintf('Negative_IMAGE%04d.jpg',(((i-
            1)*NS)+j))));
    end
end
```

Appendix D. Matlab Code of Auto Cropping

```
clc
clear

tic
load('C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\PartRecognition\ExcavatorCompleteModel\Dipper\E\OnlyDipper\E-5-
55D45-135\ROI.mat');

MainImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\PartRecognition\ExcavatorCompleteModel\Dipper\E\OnlyDipper\E-5-
55D45-135\NewGenImg\';
JPG = '*.jpg';
ImList = dir(fullfile(MainImg,JPG));
% Number of Images
NoI = length(ImList);

PreImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\PartRecognition\ExcavatorCompleteModel\Dipper\E\OnlyDipper\E-5-
55D45-135\';
JPG = '*.jpg';
ImPreList = dir(fullfile(PreImg,JPG));
% Number of Images
NoPI = length(ImPreList);

NewImgBG = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\PartRecognition\ExcavatorCompleteModel\Dipper\E\OnlyDipper\E-5-
55D45-135\Cropped\';


parfor i=1:NoI+NoPI
        imgfile = DipperMat(i).imageFilename;
        I1 = imread(imgfile);
        bbox = DipperMat(i).objectBoundingBoxes;
        Icrop = imcrop(I1, bbox);
        imwrite(Icrop,fullfile(NewImgBG,sprintf('Cropped%04d.jpg',i)));
end
toc
```

Appendix E. Matlab Code of Parts Detection

```
clc
clear

%% Step 1: Read Image
% MainImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\PartRecognition\DiggingFrames\';
MainImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\CameraCalibration\CALIB\Logan\Cam_01\FarCropped\';
JPG = '*.jpg';
ImList = dir(fullfile(MainImg,JPG));
% Number of Images
NoI = length(ImList);

XMLDipper = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\PartRecognition\ExcavatorCompleteModel\Dipper\S\S5-25D135-
225\S5-25D135-225.xml';
DipperDetector = vision.CascadeObjectDetector(XMLDipper);
%DipperDetector = vision.CascadeObjectDetector(XMLDipper);
XMLBoom = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\PartRecognition\ExcavatorCompleteModel\Boom\E\E5-45D45-135\E-5-
55D45-135.xml';
BoomDetector = vision.CascadeObjectDetector(XMLBoom);
scl = 1;
scr = 1;
q= 1;
figure('OuterPosition',get(0,'screensize'))

for i=1:896
    clf
    i
    Estimated(q).Name = i;
    imgfile = fullfile(MainImg,ImList(i).name);
    I = imread(imgfile);
    Dipperbbox = step(DipperDetector, I);
    DipperArea = Dipperbbox(:,3).*Dipperbbox(:,4);
    DipperBest = find(DipperArea(:) == max(DipperArea(:)));
    DipperMaxBox = Dipperbbox(DipperBest,:);
    DipperImg = imcrop(I, DipperMaxBox(1,:));
    DipperX = DipperMaxBox(1, 1); DipperY = DipperMaxBox(1, 2);
    DipperW = DipperMaxBox(1, 3); DipperH = DipperMaxBox(1, 4);
    DipperbboxPolygon = [DipperX, DipperY, DipperX+DipperW, DipperY,...
        DipperX+DipperW, DipperY+DipperH, DipperX, DipperY+DipperH];

    Boombbox = step(BoomDetector, I);
    BoomArea = Boombbox(:,3).*Boombbox(:,4);
    BoomBest = find(BoomArea(:) == max(BoomArea(:)));
    BoomMaxBox = Boombbox(BoomBest,:);
    BoomImg = imcrop(I, BoomMaxBox);
    BoomX = BoomMaxBox(1, 1); BoomY = BoomMaxBox(1, 2);
    BoomW = BoomMaxBox(1, 3); BoomH = BoomMaxBox(1, 4);
    BoombboxPolygon = [BoomX, BoomY, BoomX+BoomW, BoomY, BoomX+BoomW,...
        BoomY+BoomH, BoomX, BoomY+BoomH];
    DipperI = insertShape(I, 'Polygon', DipperbboxPolygon,'Color','blue');
    imshow(DipperI); title('Detected Dipper');

    BoomDipperI = insertShape(DipperI, 'Polygon',
BoombboxPolygon,'Color','blue');
    imshow(BoomDipperI); title('Detected Dipper');

    BoundingBoxes = [DipperMaxBox; BoomMaxBox];
    overlapRatio = bboxOverlapRatio(DipperMaxBox,BoomMaxBox,'min');
    if overlapRatio == 0
        [xLeft, yLeft] = find(BoundingBoxes(:,1) == min(BoundingBoxes(:,1)));
        [xRight, yRight] = find(BoundingBoxes(:,1) ==
max(BoundingBoxes(:,1)));
        dist = BoundingBoxes(xRight,1) - (BoundingBoxes(xLeft,1) ...
            + BoundingBoxes(xLeft,3));
        if xLeft == 1
            % Dipper Left-side
            BoomMaxBox = [BoomX-(scl*dist),BoomY,BoomW+(2*scr*dist),BoomH];
            BoomX = BoomMaxBox(1, 1); BoomY = BoomMaxBox(1, 2);
            BoomW = BoomMaxBox(1, 3); BoomH = BoomMaxBox(1, 4);
            BoombboxPolygon = [BoomX, BoomY, BoomX+BoomW, BoomY,
BoomX+BoomW,...
                BoomY+BoomH, BoomX, BoomY+BoomH];
            BoomImg = imcrop(I, BoomMaxBox);
        else
            % Boom Left-side
            BoomMaxBox = [BoomX-(scr*dist),BoomY-
(scr*dist),BoomW+(2*scl*dist),BoomH+(1*scl*dist)];
            BoomX = BoomMaxBox(1, 1); BoomY = BoomMaxBox(1, 2);
            BoomW = BoomMaxBox(1, 3); BoomH = BoomMaxBox(1, 4);
            BoombboxPolygon = [BoomX, BoomY, BoomX+BoomW, BoomY,
BoomX+BoomW,...
                BoomY+BoomH, BoomX, BoomY+BoomH];
            BoomImg = imcrop(I, BoomMaxBox);
        end
    elseif overlapRatio > 0.2
        [xLeft, yLeft] = find(BoundingBoxes(:,1) == min(BoundingBoxes(:,1)));
        [xRight, yRight] = find(BoundingBoxes(:,1) ==
max(BoundingBoxes(:,1)));
        dist = BoundingBoxes(xRight,1) - (BoundingBoxes(xLeft,1) ...
            + BoundingBoxes(xLeft,3));
        if xLeft == 1
            % Dipper Left-side
            BoomMaxBox = [BoomX+dist,BoomY,BoomW,BoomH];
            BoomX = BoomMaxBox(1, 1); BoomY = BoomMaxBox(1, 2);
            BoomW = BoomMaxBox(1, 3); BoomH = BoomMaxBox(1, 4);
            BoombboxPolygon = [BoomX, BoomY, BoomX+BoomW, BoomY,
BoomX+BoomW,...
                BoomY+BoomH, BoomX, BoomY+BoomH];
            BoomImg = imcrop(I, BoomMaxBox);
        else
            % Boom Left-side
            BoomMaxBox = [BoomX,BoomY,BoomW+dist,BoomH];
            BoomX = BoomMaxBox(1, 1); BoomY = BoomMaxBox(1, 2);
            BoomW = BoomMaxBox(1, 3); BoomH = BoomMaxBox(1, 4);
            BoombboxPolygon = [BoomX, BoomY, BoomX+BoomW, BoomY,
BoomX+BoomW,...
                BoomY+BoomH, BoomX, BoomY+BoomH];
            BoomImg = imcrop(I, BoomMaxBox);
        end
    end

    overlapRatio = bboxOverlapRatio(DipperMaxBox,BoomMaxBox);
    if overlapRatio > 0
        BoomMaxBox(1, 3)= BoomMaxBox(1, 3) - ((BoomMaxBox(1, 1) + ...
            BoomMaxBox(1, 3)) - DipperMaxBox(1, 1));
    end

    % Draw the returned bounding box around the detected face.
    DipperI = insertShape(I, 'Polygon', DipperbboxPolygon,'Color','blue');
    imshow(DipperI); title('Detected Dipper');

    BoomDipperI = insertShape(DipperI, 'Polygon',
BoombboxPolygon,'Color','blue');
    imshow(BoomDipperI); title('Detected Dipper');

    %% Step 2: Convert Image from RGB Color Space to L*a*b* Color Space
    % Convert the image to L*a*b* color space using |makecform| and
|applycform|.

    cform = makecform('srgb2lab');
%     lab_Dipper = applycform(DipperImg,cform);
    lab_Dipper = rgb2lab(DipperImg);
    lab_Boom = applycform(BoomImg,cform);
    lab_Merge = applycform(I,cform);

    %% Step 3: Classify the Colors in 'a*b*' Space Using K-Means Clustering
    Dipper_ab = double(lab_Dipper(:,:,2:3));
    Dipper_nrows = size(Dipper_ab,1);
    Dipper_ncols = size(Dipper_ab,2);
    Dipper_ab = reshape(Dipper_ab,Dipper_nrows*Dipper_ncols,2);

    Boom_ab = double(lab_Boom(:,:,2:3));
    Boom_nrows = size(Boom_ab,1);
    Boom_ncols = size(Boom_ab,2);
    Boom_ab = reshape(Boom_ab,Boom_nrows*Boom_ncols,2);

    Merge_ab = double(lab_Merge(:,:,2:3));
    Merge_nrows = size(Merge_ab,1);
    Merge_ncols = size(Merge_ab,2);
    Merge_ab = reshape(Merge_ab,Merge_nrows*Merge_ncols,2);

    nColors = 3;
    % repeat the clustering 3 times to avoid local minima
    [Dipper_cluster_idx Dipper_cluster_center] = kmeans(Dipper_ab,...
        nColors,'distance','sqEuclidean','Replicates',3);
    [Boom_cluster_idx Boom_cluster_center] = kmeans(Boom_ab,...
        nColors,'distance','sqEuclidean','Replicates',3);

    [Merge_cluster_idx Merge_cluster_center] = kmeans(Merge_ab,...
        nColors,'distance','sqEuclidean','Replicates',3);

    %% Step 4: Label Every Pixel in the Image Using the Results from KMEANS
    Dipper_pixel_labels =
reshape(Dipper_cluster_idx,Dipper_nrows,Dipper_ncols);
    imshow(Dipper_pixel_labels,[]), title('image labeled by cluster index');

    Boom_pixel_labels = reshape(Boom_cluster_idx,Boom_nrows,Boom_ncols);
    imshow(Boom_pixel_labels,[]), title('image labeled by cluster index');

    Merge_pixel_labels = reshape(Merge_cluster_idx,Merge_nrows,Merge_ncols);
    imshow(Merge_pixel_labels,[]), title('image labeled by cluster index');

    %% Step 5: Create Images that Segment the H&E Image by Color.
    Merge_segmented_images = cell(1,3);
    Merge_rgb_label = repmat(Merge_pixel_labels,[1 1 3]);

    for k = 1:nColors
        Merge_color = I;
        Merge_color(Merge_rgb_label ~= k) = 0;
        Merge_segmented_images{k} = Merge_color;
    end

    subplot(3,5,11)
    imshow(Merge_segmented_images{1}), title('Cluster 1');
    subplot(3,5,12)
    imshow(Merge_segmented_images{2}), title('Cluster 2');
    Merge_Temp_1 = rgb2gray(Merge_segmented_images{1});
    Merge_Temp_1(2:end-1,2:end-1)=0;
    MC_1 = length(find(Merge_Temp_1 ~= 0));
    Merge_Temp_2 = rgb2gray(Merge_segmented_images{2});
    Merge_Temp_2(2:end-1,2:end-1)=0;
    MC_2 = length(find(Merge_Temp_2 ~= 0));
    if MC_1 < MC_2
        subplot(3,5,13)
        imshow(Merge_segmented_images{1}), title('Total');
        Merge_img = Merge_segmented_images{1};
        MC = 1;
    else
        subplot(3,5,13)
        imshow(Merge_segmented_images{2}), title('Total');
        Merge_img = Merge_segmented_images{2};
        MC = 2;
    end

    [counts_Merge_img,x_Merge_img] = imhist(rgb2gray(Merge_img));
    counts_Merge_img(1) = 0;
    pd_Merge_img = fitdist(x_Merge_img, 'extreme
value','freq',counts_Merge_img);
```

```
    Dipper_segmented_images = cell(1,3);
    Dipper_rgb_label = repmat(Dipper_pixel_labels,[1 1 3]);

    for k = 1:nColors
        Dipper_color = DipperImg;
        Dipper_color(Dipper_rgb_label ~= k) = 0;
        Dipper_segmented_images{k} = Dipper_color;
    end

%     figure,
    subplot(3,5,1)
    imshow(Dipper_segmented_images{1}), title('Cluster 1');
    subplot(3,5,2)
    imshow(Dipper_segmented_images{2}), title('Cluster 2');
    Dipper_Temp_1 = rgb2gray(Dipper_segmented_images{1});
    Dipper_Temp_1(2:end-1,2:end-1)=0;
    DC_1 = length(find(Dipper_Temp_1 ~= 0));
    Dip_Temp_2 = rgb2gray(Dipper_segmented_images{2});
    Dip_Temp_2(2:end-1,2:end-1)=0;
    DC_2 = length(find(Dip_Temp_2 ~= 0));
    if DC_1 < DC_2
        subplot(3,5,3)
        imshow(Dipper_segmented_images{1}), title('Dipper');
        Dipper_img = Dipper_segmented_images{1};
        DC = 1;
    else
        subplot(3,5,3)
        imshow(Dipper_segmented_images{2}), title('Dipper');
        Dipper_img = Dipper_segmented_images{2};
        DC = 2;
    end

    [counts_Dipper_img,x_Dipper_img] = imhist(rgb2gray(Dipper_img));
    counts_Dipper_img(1) = 0;
    pd_Dipper_img = fitdist(x_Dipper_img, 'extreme
value','freq',counts_Dipper_img);

    Boom_segmented_images = cell(1,3);
    Boom_rgb_label = repmat(Boom_pixel_labels,[1 1 3]);

    for k = 1:nColors
        Boom_color = BoomImg;
        Boom_color(Boom_rgb_label ~= k) = 0;
        Boom_segmented_images{k} = Boom_color;
    end

    subplot(3,5,6)
    imshow(Boom_segmented_images{1}), title('Cluster 1');
    subplot(3,5,7)
    imshow(Boom_segmented_images{2}), title('Cluster 2');
    Boom_Temp_1 = rgb2gray(Boom_segmented_images{1});
    Boom_Temp_1(2:end-1,2:end-1)=0;
    BC_1 = length(find(Boom_Temp_1 ~= 0));
    Boom_Temp_2 = rgb2gray(Boom_segmented_images{2});
    Boom_Temp_2(2:end-1,2:end-1)=0;
    BC_2 = length(find(Boom_Temp_2 ~= 0));
    if BC_1 < BC_2
        subplot(3,5,8)
        imshow(Boom_segmented_images{1}), title('Boom');
        Boom_img = Boom_segmented_images{1};
        BC = 1;
    else
        subplot(3,5,8)
        imshow(Boom_segmented_images{2}), title('Boom');
        Boom_img = Boom_segmented_images{2};
        BC = 2;
    end

    [counts_Boom_img,x_Boom_img] = imhist(rgb2gray(Boom_img));
    counts_Boom_img(1) = 0;
    pd_Boom_img = fitdist(x_Boom_img, 'extreme
value','freq',counts_Boom_img);

    % Second Time
    se90 = strel('line', 3, 90);
    se0 = strel('line', 3, 0);

    [~, Dipper_threshold] = edge(rgb2gray(Dipper_img), 'sobel');
    fudgeFactor = 0.5;
    Dipper_BWs = edge(rgb2gray(Dipper_img),'sobel', Dipper_threshold *
fudgeFactor);
    Dipper_Base = ones(size(Dipper_BWs));
    Dipper_BWsdil = imdilate(Dipper_BWs, [se90 se0]);
    Dipper_BWdfill = imfill(Dipper_BWsdil, 'holes');
    Dipper_New_BW = Dipper_Base - Dipper_BWdfill;
    Dipper_New_img = Dipper_img.*repmat(uint8(Dipper_New_BW), [1 1 3]);
    Dipper_regdata = regionprops(im2bw(Dipper_New_img,0.01),'Area',....
        'FilledImage','PixelList');
    Dipper_Area = cat(1, Dipper_regdata.Area);
    Dipper_Best = find(Dipper_Area(:) == max(Dipper_Area(:)));
    Dipper_PixelList = cat(1, Dipper_regdata(Dipper_Best).PixelList);
    Dipper_BWnobord = zeros(size(Dipper_New_BW));
    for j=1:length(Dipper_PixelList)
        Dipper_BWnobord(Dipper_PixelList(j,2),Dipper_PixelList(j,1)) = 1;
    end

    seD = strel('diamond',1);
    Dipper_BWfinal = imerode(Dipper_BWnobord,seD);
    Dipper_BWfinal = imerode(Dipper_BWfinal,seD);
    subplot(3,5,4)
    [Dipper_x, Dipper_y] = size(Dipper_BWfinal);
    Dipper_q = boundary(Dipper_PixelList(:,1),Dipper_PixelList(:,2),0.99);
    plot(Dipper_PixelList(Dipper_q,1),Dipper_PixelList(Dipper_q,2));
    Dipper_mask = poly2mask(Dipper_PixelList(Dipper_q,1), ...
        Dipper_PixelList(Dipper_q,2),Dipper_x,Dipper_y);
    imshow(Dipper_mask), title('Segmented Dipper');
    %Second Time End
    [~, Boom_threshold] = edge(rgb2gray(Boom_img), 'sobel');
    fudgeFactor = 0.5;
    Boom_BWs = edge(rgb2gray(Boom_img),'sobel', Boom_threshold *
fudgeFactor);
    imshow(Boom_BWs), title('binary gradient mask');
    Boom_Base = ones(size(Boom_BWs));
    Boom_BWsdil = imdilate(Boom_BWs, [se90 se0]);
    imshow(Boom_BWsdil), title('dilated gradient mask');
    Boom_New_BW = Boom_Base - Boom_BWsdil;
    imshow(Boom_New_BW)
    Boom_New_img = Boom_img.*repmat(uint8(Boom_New_BW), [1 1 3]);
    imshow(Boom_New_img)
    Boom_regdata = regionprops(im2bw(Boom_New_img,0.01),'Area',....
        'FilledImage','PixelList');
    Boom_Area = cat(1, Boom_regdata.Area);
    Boom_Best = find(Boom_Area(:) == max(Boom_Area(:)));
    Boom_FilledImage = cat(1, Boom_regdata(Boom_Best).FilledImage);
    Boom_PixelList = cat(1, Boom_regdata(Boom_Best).PixelList);
    Boom_BWnobord = zeros(size(Boom_New_BW));
    for j=1:length(Boom_PixelList)
        Boom_BWnobord(Boom_PixelList(j,2),Boom_PixelList(j,1)) = 1;
    end

    Boom_BWfinal = imerode(Boom_BWnobord,seD);
    Boom_BWfinal = imerode(Boom_BWfinal,seD);
    subplot(3,5,9)
    [Boom_x, Boom_y] = size(Boom_BWfinal);
    Boom_q = boundary(Boom_PixelList(:,1),Boom_PixelList(:,2),0.7);
    plot(Boom_PixelList(Boom_q,1),Boom_PixelList(Boom_q,2));
    Boom_mask = poly2mask(Boom_PixelList(Boom_q,1), ...
        Boom_PixelList(Boom_q,2),Boom_x,Boom_y);
    imshow(Boom_mask), title('Segmented Boom');

    [~, Merge_threshold] = edge(rgb2gray(Merge_img), 'sobel');
    fudgeFactor = 0.5;
    Merge_BWs = edge(rgb2gray(Merge_img),'sobel', Merge_threshold *
fudgeFactor);
    Merge_Base = ones(size(Merge_BWs));
    Merge_BWsdil = imdilate(Merge_BWs, [se90 se0]);
    Merge_BWdfill = imfill(Merge_BWsdil, 'holes');
    Merge_New_BW = Merge_Base - Merge_BWdfill;
    Merge_New_img = Merge_img.*repmat(uint8(Merge_New_BW), [1 1 3]);
    Merge_regdata = regionprops(im2bw(Merge_New_img,0.01),'Area',....
        'FilledImage','PixelList');
    Merge_Area = cat(1, Merge_regdata.Area);
    Merge_Best = find(Merge_Area(:) == max(Merge_Area(:)));
    Merge_PixelList = cat(1, Merge_regdata(Merge_Best).PixelList);
    Merge_BWnobord = zeros(size(Merge_New_BW));
    for j=1:length(Merge_PixelList)
        Merge_BWnobord(Merge_PixelList(j,2),Merge_PixelList(j,1)) = 1;
    end

    seD = strel('diamond',1);
    Merge_BWfinal = imerode(Merge_BWnobord,seD);
    Merge_BWfinal = imerode(Merge_BWfinal,seD);
```

```
subplot(3,5,14)
imshow(Merge_BWfinal), title('Segmented Body');

%% Step 6: Segment the Nuclei into a Separate Image
Dipper_mean_cluster_value = mean(Dipper_cluster_center,2);
[Dipper_tmp, Dipper_idx] = sort(Dipper_mean_cluster_value);
Dipper_blue_cluster_num = Dipper_idx(1);

Dipper_L = lab_Dipper(:,:,1);
Dipper_blue_idx = find(Dipper_pixel_labels == Dipper_blue_cluster_num);
Dipper_L_blue = Dipper_L(Dipper_blue_idx);
Dipper_is_light_blue = im2bw(Dipper_L_blue,graythresh(Dipper_L_blue));
Dipper_nuclei_labels = repmat(uint8(0),[Dipper_nrows Dipper_ncols]);
Dipper_nuclei_labels(Dipper_blue_idx(Dipper_is_light_blue==false)) = 1;
Dipper_nuclei_labels = repmat(Dipper_nuclei_labels,[1 1 3]);
Dipper_blue_nuclei = DipperImg;
Dipper_blue_nuclei(Dipper_nuclei_labels ~= 1) = 0;
Boom_mean_cluster_value = mean(Boom_cluster_center,2);
[Boom_tmp, Boom_idx] = sort(Boom_mean_cluster_value);
Boom_blue_cluster_num = Boom_idx(1);
Boom_L = lab_Boom(:,:,1);
Boom_blue_idx = find(Boom_pixel_labels == Boom_blue_cluster_num);
Boom_L_blue = Boom_L(Boom_blue_idx);
Boom_is_light_blue = im2bw(Boom_L_blue,graythresh(Boom_L_blue));
Boom_nuclei_labels = repmat(uint8(0),[Boom_nrows Boom_ncols]);
Boom_nuclei_labels(Boom_blue_idx(Boom_is_light_blue==false)) = 1;
Boom_nuclei_labels = repmat(Boom_nuclei_labels,[1 1 3]);
Boom_blue_nuclei = BoomImg;
Boom_blue_nuclei(Boom_nuclei_labels ~= 1) = 0;

%% Step 7: Skeleton
[Dipper_skr,Dipper_rad] = skeleton(Dipper_mask); (Howe, 2006)
Dipper_skel = bwmorph(Dipper_skr > 35,'skel',inf);
subplot(3,5,5)
imshow(Dipper_skel), title('Skeleton Dipper');
% try different thresholds besides 35 to see the effects

% anaskel returns the locations of endpoints and junction points
[Dipper_dmap,Dipper_exy,Dipper_jxy] = anaskel(Dipper_skel); (Howe, 2006)
hold on
plot(Dipper_exy(1,:),Dipper_exy(2,:),'go')
plot(Dipper_jxy(1,:),Dipper_jxy(2,:),'ro')

[Boom_skr,Boom_rad] = skeleton(Boom_mask);
Boom_skel = bwmorph(Boom_skr > 35,'skel',inf);
subplot(3,5,10)
imshow(Boom_skel,'InitialMagnification','fit'), title('Skeleton Boom');
[Boom_dmap,Boom_exy,Boom_jxy] = anaskel(Boom_skel); (Howe, 2006)
hold on
plot(Boom_exy(1,:),Boom_exy(2,:),'go')
plot(Boom_jxy(1,:),Boom_jxy(2,:),'ro')

[Merge_skr,Merge_rad] = skeleton(Merge_BWfinal);
Merge_skel = bwmorph(Merge_skr > 35,'skel',inf);
```

```
    subplot(3,5,15)
    imshow(Merge_skel,'InitialMagnification','fit'), title('Segmented
Merge');
    [Merge_dmap,Merge_exy,Merge_jxy] = anaskel(Merge_skel); (Howe, 2006)
    hold on
    plot(Merge_exy(1,:),Merge_exy(2,:),'go')
    plot(Merge_jxy(1,:),Merge_jxy(2,:),'ro')
    D_Min = find(Dipper_exy(2,:)==min(Dipper_exy(2,:)));
    Dipper_Skel = Dipper_exy(:,D_Min);
    if Dipper_jxy ~= 0
        D_Center = Dipper_jxy(:,1);
    else
        D_Center = Dipper_Skel;
    end
    Dipper_Skel = [Dipper_Skel,D_Center];
    D_Max = find(Dipper_exy(2,:)==max(Dipper_exy(2,:)));
    Dipper_Skel = [Dipper_Skel,Dipper_exy(:,D_Max)];
    Dipper_Skel(1,:) = Dipper_Skel (1,:) + DipperMaxBox(1,1);
    Dipper_Skel(2,:) = Dipper_Skel (2,:) + DipperMaxBox(1,2);
    Estimated(q).Dipper = [Dipper_Skel(1,:)',Dipper_Skel(2,:)'];
    B_Min = find(Boom_exy(1,:)==max(Boom_exy(1,:)));
    if B_Min ~= 0
        Boom_Skel = Boom_exy(:,B_Min);
    else
        Boom_Skel = [0;0];
    end
    if Boom_jxy ~= 0
        B_Center = Boom_jxy(:,1);
    else
        B_Center = Boom_Skel;
    end
    Boom_Skel = [Boom_Skel,B_Center];
    B_Max = find(Boom_exy(2,:)==max(Boom_exy(2,:)));
    if length(B_Max) > 1
        B_Max(B_Max == B_Min) = [];
    end
    if B_Max ~= 0
        Boom_Skel = [Boom_Skel,Boom_exy(:,B_Max)];
    else
        Boom_Skel = [Boom_Skel,[0;0]];
    end
    Boom_Skel(1,:) = Boom_Skel (1,:) + BoomMaxBox(1,1);
    Boom_Skel(2,:) = Boom_Skel (2,:) + BoomMaxBox(1,2);
    Boom_Skel(:,1) = Dipper_Skel(:,2);
    Estimated(q).Boom = [Boom_Skel(1,:)',Boom_Skel(2,:)'];
    Estimated(q).Skel = [Estimated(q).Dipper(3,:);...
        Estimated(q).Dipper(1,:);Estimated(q).Boom(2,:);...
        Estimated(q).Boom(3,:)];
    subplot(3,5,14:15)
    imshow(I,'InitialMagnification','fit'), title('Segmented Merge');
    hold on
    plot(Dipper_Skel(1,:),Dipper_Skel(2,:))
    plot(Dipper_Skel(1,1),Dipper_Skel(2,1),'go')
    plot(Dipper_Skel(1,3),Dipper_Skel(2,3),'go')
    plot(Dipper_Skel(1,2),Dipper_Skel(2,2),'ro')
    plot(Boom_Skel(1,:),Boom_Skel(2,:))
    plot(Boom_Skel(1,1),Boom_Skel(2,1),'ro')
    plot(Boom_Skel(1,3),Boom_Skel(2,3),'go')
    plot(Boom_Skel(1,2),Boom_Skel(2,2),'ro')

    q = q + 1;

end
```

Appendix F. Matlab Code of Enhanced Part Segmentation and Skeleton Extraction

```
clc
clear

%% Step 1: Read Image
RootMainImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\CameraCalibration\CALIB\Logan\Cam_02\Cropped\Parts\';
Part = 'Dipper\';
MainImg = fullfile(RootMainImg,Part);
JPG = '*.jpg';
ImList = dir(fullfile(MainImg,JPG));
% Number of Images
NoI = length(ImList);
scl = 1;
scr = 1;
q= 1;
figure('OuterPosition',get(0,'screensize'))
addpath('C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\PartRecognition\ExcavatorCompleteModel\Skeleton\')
for i=21:1:NoI
    clf
    i
    Estimated(q).Name = i;
    imgfile = fullfile(MainImg,ImList(i).name);
    I = imread(imgfile);
    %% Step 2: Convert Image from RGB Color Space to L*a*b* Color Space
    % Convert the image to L*a*b* color space using |makecform| and
|applycform|.

    cform = makecform('srgb2lab');
    lab_I = applycform(I,cform);
    imshow(lab_I);
     %% Step 3: Classify the Colors in 'a*b*' Space Using K-Means Clustering
    I_ab = double(lab_I(:,:,2:3));
    I_nrows = size(I_ab,1);
    I_ncols = size(I_ab,2);
    I_ab = reshape(I_ab,I_nrows*I_ncols,2);
    nColors = 4;
    % repeat the clustering 3 times to avoid local minima
    [I_cluster_idx I_cluster_center] = kmeans(I_ab,...
        nColors,'distance','sqEuclidean','Replicates',3);
     %% Step 4: Label Every Pixel in the Image Using the Results from KMEANS
    I_pixel_labels = reshape(I_cluster_idx,I_nrows,I_ncols);
    imshow(I_pixel_labels,[]), title('image labeled by cluster index');
    %% Step 5: Create Images that Segment the H&E Image by Color.
    I_segmented_images = cell(1,3);
    I_rgb_label = repmat(I_pixel_labels,[1 1 3]);

    for k = 1:nColors
        I_color = I;
        I_color(I_rgb_label ~= k) = 0;
        I_segmented_images{k} = I_color;
        I_Temp = rgb2gray(I_segmented_images{k});
        BW = im2bw(I_Temp, 0.01);
        I_Temp(2:end-1,2:end-1)=0;
        DC(k,1) = length(find(I_Temp ~= 0));
        DC(k,2) = bwarea(BW);
    end
    Temporary_Segment = I_segmented_images;
    temp_DC = DC;
    MaxBorder = find(temp_DC(:,1)==max(temp_DC(:,1)));
    temp_DC(MaxBorder,:) = 0;
    MaxArea = find(temp_DC(:,2)==max(temp_DC(:,2)));
    I_img = I_segmented_images{MaxArea};
    temp_DC(MaxArea,:) = 0;
    Second_MaxArea = find(temp_DC(:,2)==max(temp_DC(:,2)));
    D =
pdist2([I_cluster_center(MaxArea,1),I_cluster_center(Second_MaxArea,1)],...
        [I_cluster_center(MaxArea,2),I_cluster_center(Second_MaxArea,2)]);
    Base_D =
sqrt(I_cluster_center(MaxArea,1)^2+I_cluster_center(MaxArea,2)^2);
    while nColors > 2 && (D/Base_D) < 0.05
        nColors = nColors - 1;
        I_cluster_idx = [];
        I_cluster_center = [];
        I_pixel_labels = [];
        I_segmented_images = [];
        I_rgb_label = [];
        I_color = [];
        I_Temp = [];
        BW = [];
        DC = [];
        Temporary_Segment = [];
        temp_DC = [];
        I_img = [];
        [I_cluster_idx I_cluster_center] = kmeans(I_ab,...
        nColors,'distance','sqEuclidean','Replicates',3);
         %% Step 4: Label Every Pixel in the Image Using the Results from
KMEANS
        I_pixel_labels = reshape(I_cluster_idx,I_nrows,I_ncols);
        imshow(I_pixel_labels,[]), title('image labeled by cluster index');
        %% Step 5: Create Images that Segment the H&E Image by Color.
        I_segmented_images = cell(1,3);
        I_rgb_label = repmat(I_pixel_labels,[1 1 3]);

        for k = 1:nColors
            I_color = I;
            I_color(I_rgb_label ~= k) = 0;
            I_segmented_images{k} = I_color;
            I_Temp = rgb2gray(I_segmented_images{k});
            BW = im2bw(I_Temp, 0.01);
            I_Temp(2:end-1,2:end-1)=0;
            DC(k,1) = length(find(I_Temp ~= 0));
            DC(k,2) = bwarea(BW);
        end
        Temporary_Segment = I_segmented_images;
        temp_DC = DC;
        MaxBorder = find(temp_DC(:,1)==max(temp_DC(:,1)));
        temp_DC(MaxBorder,:) = 0;
        MaxArea = find(temp_DC(:,2)==max(temp_DC(:,2)));
        I_img = I_segmented_images{MaxArea};
        if nColors > 2
            temp_DC(MaxArea,:) = 0;
            Second_MaxArea = find(temp_DC(:,2)==max(temp_DC(:,2)));
            D =
pdist2([I_cluster_center(MaxArea,1),I_cluster_center(Second_MaxArea,1)],...

[I_cluster_center(MaxArea,2),I_cluster_center(Second_MaxArea,2)]);
            Base_D =
sqrt(I_cluster_center(MaxArea,1)^2+I_cluster_center(MaxArea,2)^2);
        end
    end
    subplot(4,5,1)
    imshow(I_segmented_images{MaxBorder}), title('Cluster 1');
    subplot(4,5,2)
    imshow(I_segmented_images{MaxArea}), title('Cluster 2');
    subplot(4,5,3)
    imshow(I_img), title('Winner')
    [counts_I_img,x_I_img] = imhist(rgb2gray(I_img));
    counts_I_img(1) = 0;
    pd_I_img = fitdist(x_I_img, 'extreme value','freq',counts_I_img);

    % Second Time
    se90 = strel('diamond',3);
    se0 = strel('diamond',1);

    [~, I_threshold] = edge(rgb2gray(I_img), 'sobel');
    fudgeFactor = 0.5;
    I_BWs = edge(rgb2gray(I_img),'sobel', I_threshold * fudgeFactor);
    I_Base = ones(size(I_BWs));
    I_BWsdil = imdilate(I_BWs, [se90 se0]);
    I_BWdfill = imfill(I_BWsdil, 'holes');
    I_New_BW = I_BWdfill;
    I_New_img = I_img.*repmat(uint8(I_New_BW), [1 1 3]);
    I_regdata = regionprops(I_New_BW,'Area',....
        'FilledImage','PixelList');
    I_Area = cat(1, I_regdata.Area);
    I_Best = find(I_Area(:) == max(I_Area(:)));
    I_PixelList = cat(1, I_regdata(I_Best).PixelList);
    I_BWnobord = zeros(size(I_New_BW));
    for j=1:length(I_PixelList)
        I_BWnobord(I_PixelList(j,2),I_PixelList(j,1)) = 1;
    end

    seD = strel('diamond',1);
    I_BWfinal = imerode(I_BWnobord,seD);
    I_BWfinal = imerode(I_BWfinal,seD);
    [I_x, I_y] = size(I_BWfinal);

    switch Part
        case 'Dipper\'
            subplot(4,5,4)
            hold on
            imshow(I_BWfinal), title('Segmented I');
            I_closed = imclose(I_BWfinal,strel('disk',15));
            I_thin = bwmorph(I_closed,'thin',100);
            subplot(4,5,5)
            hold on
            imshow(I_thin), title('Skeleton I');
            I_q = boundary(I_PixelList(:,1),I_PixelList(:,2),0.99);
            plot(I_PixelList(I_q,1),I_PixelList(I_q,2));
            I_mask = poly2mask(I_PixelList(I_q,1), ...
                I_PixelList(I_q,2),I_x,I_y);
            imshow(I_mask), title('Segmented I');
            I_mask = bwconvhull(I_mask);
            imshow(I_mask), title('Segmented I');
            I_skel=bwmorph(skeleton(I_mask) > 35,'skel',inf);
            imshow(I_thin), title('Segmented I');
            [dmap,exy,jxy] = anaskel(I_skel); (Howe, 2006)

            D_Min = find(exy(2,:)==min(exy(2,:)));
            P_Skel = exy(:,D_Min);
            if jxy ~= 0
                D_Center = jxy(:,1);
            else
                D_Center = P_Skel;
            end
            D_Max = find(exy(2,:)==max(exy(2,:)));
            P_Skel = [P_Skel,exy(:,D_Max)];
            X = P_Skel(1,:)';
            Y = P_Skel(2,:)';
            [xData, yData] = prepareCurveData( X, Y );

            % Set up fittype and options.
            ft = fittype( 'poly1' );
            opts = fitoptions( 'Method', 'LinearLeastSquares' );
            opts.Robust = 'Bisquare';

            % Fit model to data.
            [fitresult, gof] = fit( xData, yData, ft, opts );
        case 'Boom\'
            subplot(4,5,4)
            hold on
            imshow(I_BWfinal), title('Segmented I');
            I_closed = imclose(I_BWfinal,strel('disk',15));
            I_thin = bwmorph(I_closed,'thin',1);
            lutfun = @(x)(sum(x(:))==4);
            lut = makelut(lutfun,2);
            I_ult = bwulterode(bwlookup(I_thin,lut));
            subplot(4,5,5)
            hold on
            imshow(I_ult), title('Skeleton I');
            [Y X] = find(I_ult==1);
            [xData, yData] = prepareCurveData( X, Y );
            if length(X)<4
                plot(I_PixelList(I_q,1),I_PixelList(I_q,2));
                I_mask = poly2mask(I_PixelList(I_q,1), ...
                    I_PixelList(I_q,2),I_x,I_y);
                imshow(I_mask), title('Segmented I');
                I_mask = bwconvhull(I_mask);
                imshow(I_mask), title('Segmented I');
                I_skel=bwmorph(skeleton(I_BWfinal) > 35,'skel',inf);
                imshow(I_skel), title('Segmented I');
                [dmap,exy,jxy] = anaskel(I_skel); (Howe, 2006)
                X = jxy(1,:)';
                Y = jxy(2,:)';
            End

            % Set up fittype and options.
            ft1 = fittype( 'power2' );

            % Fit model to data.
            [fitresult1, gof1] = fit( xData, yData, ft1 );

            % Set up fittype and options.
            ft2 = fittype( 'poly1' );
            opts = fitoptions( 'Method', 'LinearLeastSquares' );
            opts.Robust = 'Bisquare';

            % Fit model to data.
            [fitresult2, gof2] = fit( xData, yData, ft2, opts );
            gof1.rsquare
            gof2.rsquare
            if gof1.rsquare >= gof2.rsquare || gof1.rsquare >= 0.75
                fitresult = fitresult1;
                gof = gof1;
            else
                fitresult = fitresult2;
                gof = gof2;
            end
    end
    subplot(3,5,[9 10 14 15])
    imshow(I,'InitialMagnification','fit'), title('Segmented Merge');
    hold on
    X_plot = linspace(1,I_y);
    Y_plot = fitresult(X_plot);
    XY_plot = [X_plot',Y_plot];
    out_points = [];
    if XY_plot(XY_plot(:,1)>I_y,:)
        out_points = find(XY_plot(:,1)>I_y);
        XY_plot(out_points,:) = [];
    end
    out_points = [];
    if XY_plot(XY_plot(:,2)>I_x,:)
        out_points = find(XY_plot(:,2)>I_x);
        XY_plot(out_points,:) = [];
    end
    scatter(XY_plot(:,1),XY_plot(:,2));
    scatter(X,Y);
end
```

Appendix G. Matlab Code of GPS Coordinates Projection

```
P2Du=[];
projected=[];
[HQ_G2_X,HQ_G2_Y,HQ_G2_Z]  = read_kml('HQ_G2.kmz'); (Farris, 2016)
HQ_G2 = [];
j = 1;
for i=220:1:size(HQ_G2_X)
    HQ_G2(j,1) = (10e2*(HQ_G2_Z(i)+0)+14000);
    HQ_G2(j,2) = ((10e6*((HQ_G2_Y(i)-45.52)/0.417))-195500);
    HQ_G2(j,3) = (10e6*(HQ_G2_X(i)+73.55))+65000;
    j = j+1;
end
HQ_G2(1,:)=[];
figure
subplot(2,4,[1,2,5,6])
scatter3(HQ_G2(1:5000,1),HQ_G2(1:5000,2),HQ_G2(1:5000,3));
title('Subplot 1: 3D view')
hold on

X=[];
Y=[];
Z=[];
[X,Y,Z] = cylinder(200);
X=X+HQ_G2(1,1);
Y=Y+HQ_G2(1,2);
Z=(20*Z)+(HQ_G2(1,3)-10);
subplot(2,4,[1,2,5,6])
surf(X,Y,Z)
num = '2';
parameter = sprintf('Camera_Param_%c0x%c0_P.mat',num,num);
load(fullfile('C:\Users\umroot\Documents\MATLAB\My PhD
Codes\CameraCalibration\CALIB\Logan\Camera_Parameters\',parameter));
wpConvexHull = params.CameraParameters1.WorldPoints';
wpConvexHull(3,size(wpConvexHull,2)) = 0;
worldBoardCoords = [];
for boardIdx = 1:size(active_images_index,2)
    R = params.CameraParameters1.RotationMatrices(:, :, boardIdx)';
    t = params.CameraParameters1.TranslationVectors(boardIdx, :)';
    worldBoardCoords = [worldBoardCoords;bsxfun(@plus, R * wpConvexHull,
t)'];
end
subplot(2,4,[1,2,5,6])
scatter3(worldBoardCoords(:,1),worldBoardCoords(:,2),worldBoardCoords(:,3))
P3D = [];
P3D(1,:) = 4*(-HQ_G2(5091:5591,1)')+4000;
P3D(2,:) = (HQ_G2(5091:5591,2)')+3000;
P3D(3,:) = (HQ_G2(5091:5591,3)')+2000;
P2Dw=[];
P2Du=[];
transform1 = sprintf('T%c0x%c0P.mat',num,num);
load(fullfile('C:\Users\umroot\Documents\MATLAB\My PhD
Codes\CameraCalibration\CALIB\Logan\Transformations\',transform1));
transform2 = sprintf('Cam2T%c0x%c0P.mat',num,num);
```

```
load(fullfile('C:\Users\umroot\Documents\MATLAB\My PhD
Codes\CameraCalibration\CALIB\Logan\Transformations\',transform2));
P2Dw=A*[P3D(1,:);P3D(2,:);P3D(3,:);ones(1,length(P3D))];
P2Du(1,:)=P2Dw(1,:)./P2Dw(3,:);
P2Du(2,:)=P2Dw(2,:)./P2Dw(3,:);
[projected(:,1),projected(:,2)] =
transformPointsForward(A2,P2Du(1,:)',P2Du(2,:)');
P2Du2 = projected';
v1 = VideoReader('C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\LoganVideos\Camera 1\355-98to891.43.wmv');
v2 = VideoReader('C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\LoganVideos\Camera 2\355-98to891.43.wmv');
sf=1;
for i=sf:1:50
    P2C = [];
    P2Cy = [];
    projected =[];
    video1 = read(v1,(ceil(v1.FrameRate)*((i)-1)+450));
    subplot(2,4,3)
    hold on
    imshow(video1);
    video2 = read(v2,(ceil(v2.FrameRate)*((i)-1)+450));
    subplot(2,4,7)
    hold on
    imshow(video2);
    [X,Y,Z] = cylinder(16000);
    X = X+P3D(1,(10*(i-1)+1));
    Y =Y+P3D(2,(10*(i-1)+1));
    Z = (20000*Z)+(P3D(3,(10*(i-1)+1))-4000);
    X=reshape(X,size(X,1)*size(X,2),1);
    Y=reshape(Y,size(Y,1)*size(Y,2),1);
    Z=reshape(Z,size(Z,1)*size(Z,2),1);
    P2C  =A*[X';Y';Z';ones(1,length(X))];
    P2Cy(1,:)=P2C(1,:)./P2C(3,:);
    P2Cy(2,:)=P2C(2,:)./P2C(3,:);
    subplot(2,4,3)
    scatter((P2Cy(1,:)),P2Cy(2,:),50,'filled')
    if min(P2Cy(1,:)) < 1
        rect_x = 1;
    else
        rect_x = min(P2Cy(1,:));
    end
    if max(P2Cy(1,:)) > 1920
        rect_w = 1920 - rect_x;
    else
        rect_w = max(P2Cy(1,:)) - rect_x;
    end

    if min(P2Cy(2,:)) < 1
        rect_y = 1;
    else
        rect_y = min(P2Cy(2,:));
    end
    if max(P2Cy(2,:)) > 1080
        rect_h = 1080 - rect_y;
    else
        rect_h = max(P2Cy(2,:)) - rect_y;
    end
    subplot(2,4,4)
    imshow(imcrop(video1,[rect_x rect_y rect_w rect_h]));
    P2Cy2 = transformPointsForward(A2,P2Cy(1,:),P2Cy(2,:));
    subplot(2,4,7)
    scatter((P2Cy(1,:)+450),P2Cy(2,:),50,'filled')
    points = bbox2points([rect_x rect_y rect_w rect_h])';
    [projected(1,:),projected(2,:)] =
transformPointsForward(A2,points(1,:),points(2,:));
    if min(projected(1,:)) < 1
        rect_x = 1;
    else
        rect_x = min(projected(1,:));
    end
    if max(projected(1,:)) > 1920
        rect_w = 1920 - rect_x;
    else
        rect_w = max(projected(1,:)) - rect_x;
    end
    if min(projected(2,:)) < 1
        rect_y = 1;
    else
        rect_y = min(projected(2,:));
    end
    if max(projected(2,:)) > 1080
        rect_h = 1080 - rect_y;
    else
        rect_h = max(projected(2,:)) - rect_y;
    end
    figure
    imshow(imcrop(video2,[rect_x+450 rect_y rect_w rect_h]));
end
```

Appendix H. Matlab Code of Merging the Outputs of Parts' Detection

```
clc
clear

%% Step 1: Read Image
RootMainImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\CameraCalibration\CALIB\Logan\Cam_02\Cropped\';
PartsMainImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\CameraCalibration\CALIB\Logan\Cam_02\Cropped\Parts\';
JPG = '*.jpg';

% Cropped Images
CroppedList = dir(fullfile(RootMainImg,JPG));
Cropped_Data=xlsread('E:\MATLAB\My PhD
Codes\PartDetection\CropRotation.xlsx','Cropped');

% Dipper Images
Dipper = 'Dipper\';
DipperImg = fullfile(PartsMainImg,Dipper);
DipperList = dir(fullfile(DipperImg,JPG));
Dipper_Data=xlsread('E:\MATLAB\My PhD
Codes\PartDetection\CropRotation.xlsx','Dipper');

% Boom Images
Boom = 'Boom\';
BoomImg = fullfile(PartsMainImg,Boom);
BoomList = dir(fullfile(BoomImg,JPG));
Boom_Data=xlsread('E:\MATLAB\My PhD
Codes\PartDetection\CropRotation.xlsx','Boom');

% Number of Images
NoI = length(CroppedList);

scl = 1;
scr = 1;
q= 1;
figure('OuterPosition',get(0,'screensize'))
addpath('C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\PartRecognition\ExcavatorCompleteModel\Skeleton\')
for i=1:1:NoI
    clf
    i
    imgfile = fullfile(RootMainImg,CroppedList(i).name);
    Root_I = imread(imgfile);
    [Rt_y, Rt_x, Rt_z] = size(Root_I);
    subplot(3,5,[11 12 13 14 15])
    imshow(Root_I,'InitialMagnification','fit'), title('Segmented Merge');
    hold on
    XY_plot_boom = [];
    XY_plot_dipper = [];
    fitresult_dipper = [];
    fitresult_boom = [];

    %% Dipper Proccesing
    if ~isnan(Dipper_Data(i,2))
        Dipper_imgfile = fullfile(DipperImg,DipperList(i).name);
        I = imread(Dipper_imgfile);

        % Step 2: Convert Image from RGB Color Space to L*a*b* Color Space
        % Convert the image to L*a*b* color space using |makecform| and
|applycform|.
        cform = makecform('srgb2lab');
        lab_I = applycform(I,cform);
        subplot(4,5,1)
        hold on
        imshow(lab_I);

        % Step 3: Classify the Colors in 'a*b*' Space Using K-Means
Clustering
        I_ab = double(lab_I(:,:,2:3));
        I_nrows = size(I_ab,1);
        I_ncols = size(I_ab,2);
        I_ab = reshape(I_ab,I_nrows*I_ncols,2);
        nColors = 4;
        % repeat the clustering 3 times to avoid local minima
        [I_cluster_idx I_cluster_center] = kmeans(I_ab,...
            nColors,'distance','sqEuclidean','Replicates',3);

        % Step 4: Label Every Pixel in the Image Using the Results from
KMEANS
        I_pixel_labels = reshape(I_cluster_idx,I_nrows,I_ncols);
        imshow(I_pixel_labels,[]), title('image labeled by cluster index');

        % Step 5: Create Images that Segment the H&E Image by Color.
        I_segmented_images = cell(1,3);
        I_rgb_label = repmat(I_pixel_labels,[1 1 3]);

        for k = 1:nColors
            I_color = I;
            I_color(I_rgb_label ~= k) = 0;
            I_segmented_images{k} = I_color;
            I_Temp = rgb2gray(I_segmented_images{k});
            BW = im2bw(I_Temp, 0.01);
            I_Temp(2:end-1,2:end-1)=0;
            DC(k,1) = length(find(I_Temp ~= 0));
            DC(k,2) = bwarea(BW);
        end
        Temporary_Segment = I_segmented_images;
        temp_DC = DC;
        MaxBorder = find(temp_DC(:,1)==max(temp_DC(:,1)));
        temp_DC(MaxBorder,:) = 0;
        MaxArea = find(temp_DC(:,2)==max(temp_DC(:,2)));
        I_img = I_segmented_images{MaxArea};
        temp_DC(MaxArea,:) = 0;
        Second_MaxArea = find(temp_DC(:,2)==max(temp_DC(:,2)));
        D =
pdist2([I_cluster_center(MaxArea,1),I_cluster_center(Second_MaxArea,1)],...

[I_cluster_center(MaxArea,2),I_cluster_center(Second_MaxArea,2)]);
        Base_D =
sqrt(I_cluster_center(MaxArea,1)^2+I_cluster_center(MaxArea,2)^2);
        while nColors > 2 && (D/Base_D) < 0.05
            nColors = nColors - 1;
            I_cluster_idx = [];
            I_cluster_center = [];
            I_pixel_labels = [];
            I_segmented_images = [];
            I_rgb_label = [];
            I_color = [];
            I_Temp = [];
            BW = [];
            DC = [];
            Temporary_Segment = [];
            temp_DC = [];
            I_img = [];
            [I_cluster_idx I_cluster_center] = kmeans(I_ab,...
            nColors,'distance','sqEuclidean','Replicates',3);
            % Step 4: Label Every Pixel in the Image Using the Results from
KMEANS
            I_pixel_labels = reshape(I_cluster_idx,I_nrows,I_ncols);
            imshow(I_pixel_labels,[]), title('image labeled by cluster
index');
            % Step 5: Create Images that Segment the H&E Image by Color.
            I_segmented_images = cell(1,3);
            I_rgb_label = repmat(I_pixel_labels,[1 1 3]);

            for k = 1:nColors
                I_color = I;
                I_color(I_rgb_label ~= k) = 0;
                I_segmented_images{k} = I_color;
                I_Temp = rgb2gray(I_segmented_images{k});
                BW = im2bw(I_Temp, 0.01);
                I_Temp(2:end-1,2:end-1)=0;
                DC(k,1) = length(find(I_Temp ~= 0));
                DC(k,2) = bwarea(BW);
            end
            Temporary_Segment = I_segmented_images;
            temp_DC = DC;
            MaxBorder = find(temp_DC(:,1)==max(temp_DC(:,1)));
            temp_DC(MaxBorder,:) = 0;
            MaxArea = find(temp_DC(:,2)==max(temp_DC(:,2)));
            I_img = I_segmented_images{MaxArea};
            if nColors > 2
                temp_DC(MaxArea,:) = 0;
                Second_MaxArea = find(temp_DC(:,2)==max(temp_DC(:,2)));
                D =
pdist2([I_cluster_center(MaxArea,1),I_cluster_center(Second_MaxArea,1)],...

[I_cluster_center(MaxArea,2),I_cluster_center(Second_MaxArea,2)]);
```

```
                Base_D =
sqrt(I_cluster_center(MaxArea,1)^2+I_cluster_center(MaxArea,2)^2);
            end
        end
        imshow(I_segmented_images{MaxBorder}), title('Cluster 1');
        subplot(4,5,2)
        imshow(I_segmented_images{MaxArea}), title('Cluster 2');
        subplot(4,5,3)
        imshow(I_img), title('Winner')
        [counts_I_img,x_I_img] = imhist(rgb2gray(I_img));
        counts_I_img(1) = 0;
        pd_I_img = fitdist(x_I_img, 'extreme value','freq',counts_I_img);

        % Second Time
        se90 = strel('diamond',3);
        se0 = strel('diamond',1);

        [~, I_threshold] = edge(rgb2gray(I_img), 'sobel');
        fudgeFactor = 0.5;
        I_BWs = edge(rgb2gray(I_img),'sobel', I_threshold * fudgeFactor);
        I_Base = ones(size(I_BWs));
        I_BWsdil = imdilate(I_BWs, [se90 se0]);
        I_BWdfill = imfill(I_BWsdil, 'holes');
        I_New_BW = I_BWdfill;
        I_New_img = I_img.*repmat(uint8(I_New_BW), [1 1 3]);
        I_regdata = regionprops(I_New_BW,'Area',....
            'FilledImage','PixelList');
        I_Area = cat(1, I_regdata.Area);
        I_Best = find(I_Area(:) == max(I_Area(:)));
        I_PixelList = cat(1, I_regdata(I_Best).PixelList);
        I_BWnobord = zeros(size(I_New_BW));
        for j=1:length(I_PixelList)
            I_BWnobord(I_PixelList(j,2),I_PixelList(j,1)) = 1;
        end

        seD = strel('diamond',1);
        I_BWfinal = imerode(I_BWnobord,seD);
        I_BWfinal = imerode(I_BWfinal,seD);
        [I_x, I_y] = size(I_BWfinal);
        subplot(4,5,4)
        hold on
        imshow(I_BWfinal), title('Segmented I');
        I_closed = imclose(I_BWfinal,strel('disk',15));
        I_thin = bwmorph(I_closed,'thin',100);
        subplot(4,5,5)
        hold on
        I_q = boundary(I_PixelList(:,1),I_PixelList(:,2),0.99);
        plot(I_PixelList(I_q,1),I_PixelList(I_q,2));
        I_mask = poly2mask(I_PixelList(I_q,1), ...
            I_PixelList(I_q,2),I_x,I_y);
        imshow(I_mask), title('Segmented I');
        I_mask = bwconvhull(I_mask);
        imshow(I_mask), title('Segmented I');
        I_skel=bwmorph(skeleton(I_mask) > 35,'skel',inf);
        imshow(I_thin), title('Segmented I');
        [dmap,exy,jxy] = anaskel(I_skel); (Howe, 2006)

        D_Min = find(exy(2,:)==min(exy(2,:)));
        P_Skel = exy(:,D_Min);
        if jxy ~= 0
            D_Center = jxy(:,1);
        else
            D_Center = P_Skel;
        end
        D_Max = find(exy(2,:)==max(exy(2,:)));
        P_Skel = [P_Skel,exy(:,D_Max)];

        % Translation and Rotation
        Tr_P_Skel = P_Skel + [Dipper_Data(i,2); Dipper_Data(i,4)];
        if Dipper_Data(i,6) < 0
            Cent_P_Skel(1,:) = Tr_P_Skel(1,:);
            Cent_P_Skel(2,:) =
Tr_P_Skel(2,:)+(Rt_x*sin(deg2rad(Dipper_Data(i,6))));
            [THETA,R] = cart2pol(Cent_P_Skel(1,:),Cent_P_Skel(2,:));
            THETA = THETA+deg2rad(-Dipper_Data(i,6));
            [yr,xr] = pol2cart(THETA,R);
            X = yr;
            Y = xr;
        elseif Dipper_Data(i,6) > 0
            Cent_P_Skel(1,:) = Tr_P_Skel(1,:)-
(Rt_y*sin(deg2rad(Dipper_Data(i,6))));
            Cent_P_Skel(2,:) = Tr_P_Skel(2,:);
            [THETA,R] = cart2pol(Cent_P_Skel(1,:),Cent_P_Skel(2,:));
            THETA = THETA+deg2rad(-Dipper_Data(i,6));
            [yr,xr] = pol2cart(THETA,R);
            X = yr;
            Y = xr;
        else
            X = Tr_P_Skel(1,:)';
            Y = Tr_P_Skel(2,:)';
        end
        [xData, yData] = prepareCurveData( X, Y );

        % Set up fittype and options.
        ft = fittype( 'poly1' );
        opts = fitoptions( 'Method', 'LinearLeastSquares' );
        opts.Robust = 'Bisquare';

        % Fit model to data.
        [fitresult_dipper, gof_dipper] = fit( xData, yData, ft, opts );
        subplot(3,5,[11 12 13 14 15])
        imshow(Root_I,'InitialMagnification','fit'), title('Segmented
Merge');
        hold on
        X_plot = linspace(min(X),max(X));
        Y_plot = fitresult_dipper(X_plot);
        XY_plot_dipper = [X_plot',Y_plot];
        scatter(XY_plot_dipper(:,1),XY_plot_dipper(:,2));
        scatter(X,Y);
    end
    %% Boom Proccesing
    if ~isnan(Boom_Data(i,2))
        Boom_imgfile = fullfile(BoomImg,BoomList(i).name);
        I = imread(Boom_imgfile);

        % Step 2: Convert Image from RGB Color Space to L*a*b* Color Space
        % Convert the image to L*a*b* color space using |makecform| and
|applycform|.
        cform = makecform('srgb2lab');
        lab_I = applycform(I,cform);

        % Step 3: Classify the Colors in 'a*b*' Space Using K-Means
Clustering
        I_ab = double(lab_I(:,:,2:3));
        I_nrows = size(I_ab,1);
        I_ncols = size(I_ab,2);
        I_ab = reshape(I_ab,I_nrows*I_ncols,2);
        nColors = 4;
        % repeat the clustering 3 times to avoid local minima
        [I_cluster_idx I_cluster_center] = kmeans(I_ab,...
            nColors,'distance','sqEuclidean','Replicates',3);

        % Step 4: Label Every Pixel in the Image Using the Results from
KMEANS
        I_pixel_labels = reshape(I_cluster_idx,I_nrows,I_ncols);
%         imshow(I_pixel_labels,[]), title('image labeled by cluster index');

        % Step 5: Create Images that Segment the H&E Image by Color.
        I_segmented_images = cell(1,3);
        I_rgb_label = repmat(I_pixel_labels,[1 1 3]);

        for k = 1:nColors
            I_color = I;
            I_color(I_rgb_label ~= k) = 0;
            I_segmented_images{k} = I_color;
            I_Temp = rgb2gray(I_segmented_images{k});
            BW = im2bw(I_Temp, 0.01);
            I_Temp(2:end-1,2:end-1)=0;
            DC(k,1) = length(find(I_Temp ~= 0));
            DC(k,2) = bwarea(BW);
        end
        Temporary_Segment = I_segmented_images;
        temp_DC = DC;
        MaxBorder = find(temp_DC(:,1)==max(temp_DC(:,1)));
        temp_DC(MaxBorder,:) = 0;
        MaxArea = find(temp_DC(:,2)==max(temp_DC(:,2)));
        I_img = I_segmented_images{MaxArea};
        temp_DC(MaxArea,:) = 0;
        Second_MaxArea = find(temp_DC(:,2)==max(temp_DC(:,2)));
        D =
pdist2([I_cluster_center(MaxArea,1),I_cluster_center(Second_MaxArea,1)],...

[I_cluster_center(MaxArea,2),I_cluster_center(Second_MaxArea,2)]);
```

```
        Base_D = 
sqrt(I_cluster_center(MaxArea,1)^2+I_cluster_center(MaxArea,2)^2);
        while nColors > 2 && (D/Base_D) < 0.05
            nColors = nColors - 1;
            I_cluster_idx = [];
            I_cluster_center = [];
            I_pixel_labels = [];
            I_segmented_images = [];
            I_rgb_label = [];
            I_color = [];
            I_Temp = [];
            BW = [];
            DC = [];
            Temporary_Segment = [];
            temp_DC = [];
            I_img = [];
            [I_cluster_idx I_cluster_center] = kmeans(I_ab,...
            nColors,'distance','sqEuclidean','Replicates',3);
            % Step 4: Label Every Pixel in the Image Using the Results from 
KMEANS
            I_pixel_labels = reshape(I_cluster_idx,I_nrows,I_ncols);
            % Step 5: Create Images that Segment the H&E Image by Color.
            I_segmented_images = cell(1,3);
            I_rgb_label = repmat(I_pixel_labels,[1 1 3]);

            for k = 1:nColors
                I_color = I;
                I_color(I_rgb_label ~= k) = 0;
                I_segmented_images{k} = I_color;
                I_Temp = rgb2gray(I_segmented_images{k});
                BW = im2bw(I_Temp, 0.01);
                I_Temp(2:end-1,2:end-1)=0;
                DC(k,1) = length(find(I_Temp ~= 0));
                DC(k,2) = bwarea(BW);
            end
            Temporary_Segment = I_segmented_images;
            temp_DC = DC;
            MaxBorder = find(temp_DC(:,1)==max(temp_DC(:,1)));
            temp_DC(MaxBorder,:) = 0;
            MaxArea = find(temp_DC(:,2)==max(temp_DC(:,2)));
            I_img = I_segmented_images{MaxArea};
            if nColors > 2
                temp_DC(MaxArea,:) = 0;
                Second_MaxArea = find(temp_DC(:,2)==max(temp_DC(:,2)));
                D = 
pdist2([I_cluster_center(MaxArea,1),I_cluster_center(Second_MaxArea,1)],...

[I_cluster_center(MaxArea,2),I_cluster_center(Second_MaxArea,2)]);
                Base_D = 
sqrt(I_cluster_center(MaxArea,1)^2+I_cluster_center(MaxArea,2)^2);
            end
        end
        subplot(4,5,6)
        imshow(I_segmented_images{MaxBorder}), title('Cluster 1');
        subplot(4,5,7)
```

```
imshow(I_segmented_images{MaxArea}), title('Cluster 2');
subplot(4,5,8)
imshow(I_img), title('Winner')
[counts_I_img,x_I_img] = imhist(rgb2gray(I_img));
counts_I_img(1) = 0;
pd_I_img = fitdist(x_I_img, 'extreme value','freq',counts_I_img);

% Second Time
se90 = strel('diamond',3);
se0 = strel('diamond',1);

[~, I_threshold] = edge(rgb2gray(I_img), 'sobel');
fudgeFactor = 0.5;
I_BWs = edge(rgb2gray(I_img),'sobel', I_threshold * fudgeFactor);
I_Base = ones(size(I_BWs));
I_BWsdil = imdilate(I_BWs, [se90 se0]);
I_BWdfill = imfill(I_BWsdil, 'holes');
I_New_BW = I_BWdfill;
I_New_img = I_img.*repmat(uint8(I_New_BW), [1 1 3]);
I_regdata = regionprops(I_New_BW,'Area',....
    'FilledImage','PixelList');
I_Area = cat(1, I_regdata.Area);
I_Best = find(I_Area(:) == max(I_Area(:)));
I_PixelList = cat(1, I_regdata(I_Best).PixelList);
I_BWnobord = zeros(size(I_New_BW));
for j=1:length(I_PixelList)
    I_BWnobord(I_PixelList(j,2),I_PixelList(j,1)) = 1;
end

seD = strel('diamond',1);
I_BWfinal = imerode(I_BWnobord,seD);
I_BWfinal = imerode(I_BWfinal,seD);
[I_x, I_y] = size(I_BWfinal);

subplot(4,5,9)
hold on
imshow(I_BWfinal), title('Segmented I');
I_closed = imclose(I_BWfinal,strel('disk',15));
I_thin = bwmorph(I_closed,'thin',1);
lutfun = @(x)(sum(x(:))==4);
lut = makelut(lutfun,2);
I_ult = bwulterode(bwlookup(I_thin,lut));
subplot(4,5,10)
hold on
imshow(I_ult), title('Skeleton I');
P_Skel = [];
[P_Skel(2,:), P_Skel(1,:)] = find(I_ult==1);

if length(P_Skel(1,:))<4
    P_Skel = [];
    plot(I_PixelList(I_q,1),I_PixelList(I_q,2));
    I_mask = poly2mask(I_PixelList(I_q,1), ...
        I_PixelList(I_q,2),I_x,I_y);
    imshow(I_mask), title('Segmented I');
```

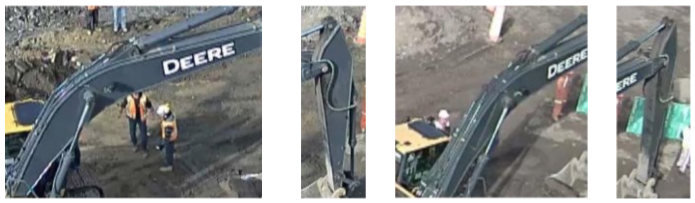
```
            I_mask = bwconvhull(I_mask);
            imshow(I_mask), title('Segmented I');
            I_skel=bwmorph(skeleton(I_BWfinal) > 35,'skel',inf);
            imshow(I_skel), title('Segmented I');
            [dmap,exy,jxy] = anaskel(I_skel); (Howe, 2006)
            P_Skel = jxy;
        end

        % Translation and Rotation
        Tr_P_Skel = P_Skel + [Boom_Data(i,2); Boom_Data(i,4)];
        if Boom_Data(i,6) < 0
            Cent_P_Skel(1,:) = Tr_P_Skel(1,:);
            Cent_P_Skel(2,:) =
Tr_P_Skel(2,:)+(Rt_x*sin(deg2rad(Boom_Data(i,6))));
            [THETA,R] = cart2pol(Cent_P_Skel(1,:),Cent_P_Skel(2,:));
            THETA = THETA+deg2rad(-Boom_Data(i,6));
            [yr,xr] = pol2cart(THETA,R);
            X = yr;
            Y = xr;
        elseif Boom_Data(i,6) > 0
            Cent_P_Skel(1,:) = Tr_P_Skel(1,:)-
(Rt_y*sin(deg2rad(Boom_Data(i,6))));
            Cent_P_Skel(2,:) = Tr_P_Skel(2,:);
            [THETA,R] = cart2pol(Cent_P_Skel(1,:),Cent_P_Skel(2,:));
            THETA = THETA+deg2rad(-Boom_Data(i,6));
            [yr,xr] = pol2cart(THETA,R);
            X = yr;
            Y = xr;
        else
            X = Tr_P_Skel(1,:)';
            Y = Tr_P_Skel(2,:)';
        end
        [xData, yData] = prepareCurveData( X, Y );
        ft1 = fittype( 'power1' );

        % Fit model to data.
        [fitresult1, gof1] = fit( xData, yData, ft1 );

        % Set up fittype and options.
        ft2 = fittype( 'poly1' );
        opts = fitoptions( 'Method', 'LinearLeastSquares' );
        opts.Robust = 'Bisquare';

        % Fit model to data.
        [fitresult2, gof2] = fit( xData, yData, ft2, opts );
        gof1.rsquare
        gof2.rsquare
        if gof1.rsquare >= gof2.rsquare || gof1.rsquare >= 0.75
            fit_type = 'power1';
            fitresult_boom = fitresult1;
            gof_boom = gof1;
        else
            fit_type = 'poly1';
            fitresult_boom = fitresult2;
            gof_boom = gof2;
        end
        subplot(3,5,[11 12 13 14 15])
        hold on
        X_plot = linspace(Boom_Data(i,2),Boom_Data(i,2)+Boom_Data(i,7));
        Y_plot = fitresult_boom(X_plot);
        XY_plot_boom = [X_plot',Y_plot];
        XY_plot_boom(XY_plot_boom(:,2)>(Boom_Data(i,4)+Boom_Data(i,7)),:)=[];
        XY_plot_boom(XY_plot_boom(:,2)<(Boom_Data(i,4)),:)=[];
        scatter(XY_plot_boom(:,1),XY_plot_boom(:,2));
        scatter(X,Y);
    end

    syms x
    if ~isempty(fitresult_dipper) && ~isempty(fitresult_boom)
        switch fit_type
            case 'power1'
                eqn_dipper = fitresult_dipper.p1*x + fitresult_dipper.p2;
                eqn_boom = fitresult_boom.a*x^fitresult_boom.b;
                intersect(1,:) = vpasolve(eqn_dipper-eqn_boom,x,[1, Rt_x]);
                intersect(2,:) = fitresult_boom(double(intersect(1,:)));
            case 'poly1'
                eqn_dipper = fitresult_dipper.p1*x + fitresult_dipper.p2;
                eqn_boom = fitresult_boom.p1*x + fitresult_boom.p2;
                intersect = vpasolve(eqn_dipper-eqn_boom,x,[1, Rt_x]);
                intersect(2,:) = fitresult_boom(double(intersect(1,:)));
        end

        if intersect(2,:) > Rt_y || intersect(2,:) < 0
            intersect = [];
            temp_dist(1) = pdist([XY_plot_boom(1,:);XY_plot_dipper(1,:)]);
            temp_dist(2) = pdist([XY_plot_boom(end,:);XY_plot_dipper(1,:)]);
            temp_dist(3) = pdist([XY_plot_boom(1,:);XY_plot_dipper(end,:)]);
            temp_dist(4) =
pdist([XY_plot_boom(end,:);XY_plot_dipper(end,:)]);
            min_dist = find(temp_dist==min(temp_dist));
            switch min_dist
                case 1
                    P = XY_plot_boom(1,:);
                case 2
                    P = XY_plot_boom(end,:);
                case 3
                    P = XY_plot_boom(1,:);
                case 4
                    P = XY_plot_boom(end,:);
            end

            A = XY_plot_dipper(1,:);
            B = XY_plot_dipper(end,:);
            % Direction vector
            M = B - A;

            % Running parameter t0 defines the intersection point of line
through A and B
            % and the perpendicular through P
            t0  = dot(M, P - A) / dot(M, M);
            % Intersection point of the perpendicular and line through A and
B
            intersect = (A + t0 * M)';
        end
        bd1 = pdist([intersect(:,1)';XY_plot_boom(1,:)])';
        bd2 = pdist([intersect(:,1)';XY_plot_boom(end,:)])';
        if bd1 < bd2
            sklt = [XY_plot_boom(end,:);intersect(:,1)'];
        else
            sklt = [XY_plot_boom(1,:);intersect(:,1)'];
        end
        dd1 = pdist([intersect(:,1)';XY_plot_dipper(1,:)])';
        dd2 = pdist([intersect(:,1)';XY_plot_dipper(end,:)])';
        if dd1 < dd2
            sklt = double([sklt; XY_plot_dipper(end,:)]);
        else
            sklt = double([sklt; XY_plot_dipper(1,:)]);
        end
    elseif isempty(XY_plot_boom)
        upper_pt = find(XY_plot_dipper(:,2)==min(XY_plot_dipper(:,2)));
        lower_pt = find(XY_plot_dipper(:,2)==max(XY_plot_dipper(:,2)));
        sklt = [XY_plot_dipper(lower_pt,:);XY_plot_dipper(upper_pt,:); ...
            XY_plot_dipper(lower_pt,:)];
    elseif isempty(XY_plot_dipper)
        upper_pt = find(XY_plot_boom(:,2)==min(XY_plot_boom(:,2)));
        lower_pt = find(XY_plot_boom(:,2)==max(XY_plot_boom(:,2)));
        sklt = [XY_plot_boom(lower_pt,:);XY_plot_boom(upper_pt,:); ...
            XY_plot_boom(lower_pt,:)];
    end

    SkelMat(i).imageFilename = CroppedList(i).name;
    SkelMat(i).LocalSkeleton = sklt;
    SkelMat(i).GlobalSkeleton = Cropped_Data(i,2:3)+sklt;
    i
end
```

Appendix I. Matlab Code of Coordinates Triangulation and Transferring to Unity

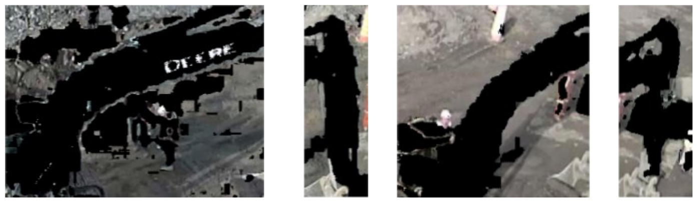
```
clear
figure('OuterPosition',get(0,'screensize'))
load('C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\DataFusion\Cam1_Close_Skeleton.mat');
load('C:\Users\umroot\Documents\MATLAB\My PhD
Codes\Papers\DataFusion\Cam2_Close_Skeleton.mat');
load('C:\Users\umroot\Documents\MATLAB\My PhD
Codes\CameraCalibration\CALIB\Logan\Camera_Parameters\Camera_Param_50x50_P.ma
t');
RightMainImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\CameraCalibration\CALIB\Logan\Cam_01\Close\';
LeftMainImg = 'C:\Users\umroot\Documents\MATLAB\My PhD
Codes\CameraCalibration\CALIB\Logan\Cam_02\Close\';
JPG = '*.jpg';
Boom_Length = 5207;
Dipper_Length = 3033;
RightList = dir(fullfile(RightMainImg,JPG));
LeftList = dir(fullfile(LeftMainImg,JPG));
t = tcpip('127.0.0.1', 55100, 'NetworkRole', 'server');
fopen(t);
for i=1:1:50
    point3d = triangulate(Cam1(i).GlobalSkeleton, Cam2(i).GlobalSkeleton,
params);
    subplot(1,3,2)
    pcshow(point3d, 'VerticalAxis', 'y', 'VerticalAxisDir', 'down',
'MarkerSize', 950);
    Skel_Dim(i,1) = abs(pdist([point3d(:,1)';point3d(:,2)']));
    Skel_Dim(i,2) = abs(pdist([point3d(:,2)';point3d(:,3)']));
    Skel_Dim(i,3) = abs(Boom_Length - Skel_Dim(i,1));
    Skel_Dim(i,4) = abs(Boom_Length - Skel_Dim(i,2));
    hold on;
    plot3(point3d(:,1),point3d(:,2),point3d(:,3));
    hold off;
    Right_imgfile = fullfile(RightMainImg,RightList(i).name);
    Right_I = imread(Right_imgfile);
    subplot(1,3,3)
    imshow(Right_I,'InitialMagnification','fit'), title('Right Camera');
    Left_imgfile = fullfile(LeftMainImg,LeftList(i).name);
    Left_I = imread(Left_imgfile);
    subplot(1,3,1)
    imshow(Left_I,'InitialMagnification','fit'), title('Left Camera');

    if i>1
        Unity.X = point3d(1,1);
        Unity.Y = point3d(2,1);
        Unity.Z = point3d(3,1);
        new_point3d = [point3d(:,2)-point3d(:,1),point3d(:,3)-point3d(:,2)];
        rBoom = vrrotvec(prev_point3d(:,1)',new_point3d(:,1)');
        mBoom = vrrotvec2mat(rBoom);
        Unity.rxBoom = rBoom(1);
        Unity.ryBoom = rBoom(2);
        Unity.rzBoom = rBoom(3);
        rDipper = vrrotvec(prev_point3d(:,2)',new_point3d(:,2)');
        mBoom = vrrotvec2mat(rDipper);
        Unity.rxDipper = rDipper(1);
        Unity.ryDipper = rDipper(2);
        Unity.rzDipper = rDipper(3);
        message = sprintf('%d,%d,%d,%d,%d,%d,%d,%d,%d',Unity.X,Unity.Y,...
            Unity.Z ,Unity.rxBoom,Unity.ryBoom,Unity.rzBoom,...
            Unity.rxDipper,Unity.ryDipper,Unity.rzDipper);
        fwrite(t, message,'char')
        pause(1);
    end
    prev_point3d = [point3d(:,2)-point3d(:,1),point3d(:,3)-point3d(:,2)];
end
mean(Skel_Dim(:,3))
mean(Skel_Dim(:,4))
```

Appendix J. Example of Orientation Estimation Process

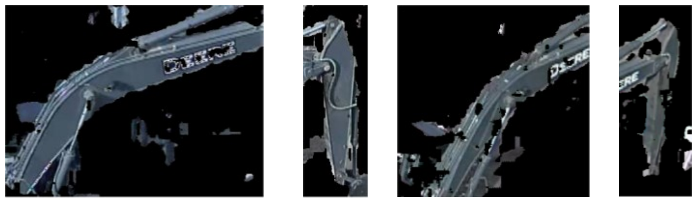
Camera 2 (Left) **Camera 1 (Right)**



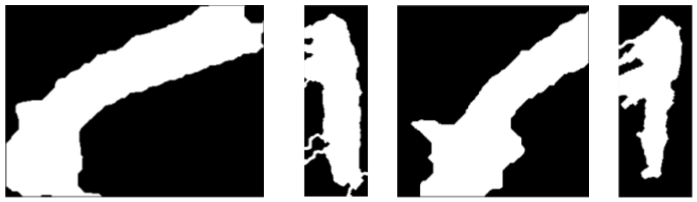
Original images



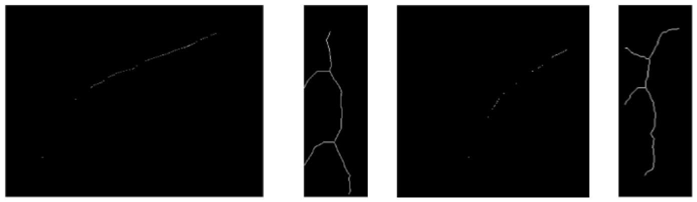
Backgrounds



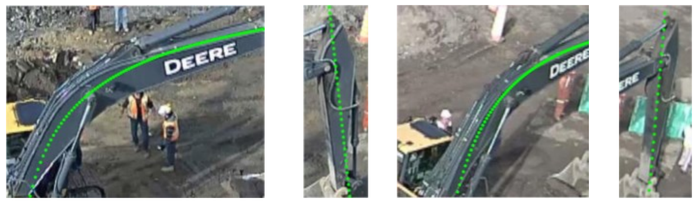
Foregrounds



Segmented parts



Skeletons



Estimated orientations