Automated Method for Airfield Pavement Condition Index Determination

by

Randall A. Pietersen

B.S., Civil and Environmental Engineering
United States Air Force Academy (2020)

Submitted to the Department of Civil and Environmental Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Civil and Environmental Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2022

Author . . . . . .
Department of Civil and Environmental Engineering
May 6, 2022

Certified by . . . . . .
Herbert H. Einstein
Professor of Civil and Environmental Engineering
Thesis Supervisor

Accepted by . . . . . .
Colette L. Heald
Professor of Civil and Environmental Engineering
Chair, Graduate Program Committee

Automated Method for Airfield Pavement Condition Index Determination

by

Randall A. Pietersen

Submitted to the Department of Civil and Environmental Engineering
on May 6, 2022, in partial fulfillment of the
requirements for the degree of
Master of Science in Civil and Environmental Engineering

Abstract

Infrastructure inspection and maintenance is a necessary, and often costly, process required for civil engineering structures throughout a project's life-cycle to ensure continued safety and serviceability. While many of these procedures have seen the introduction of technologies to assist, augment, or automate traditional methods of inspection, current practices for assessing airfield pavement serviceability remain predominately manual. Though roadway inspection has benefited from automation with the introduction of various types of sensor arrays attached to automobiles, the characteristics of airfields and their pavements have prompted research into the use of drones as a flexible, and low cost solution for automating aspects of the inspection process. As one of the largest owners and operators of airfield pavement across the globe, the United States Air Force has a unique interest in implementing such a process in a way that is both compliant and compatible with current institutional guidelines. Funded by the US Air Force Civil Engineering Center, this research proposes a novel method for conducting an automated airfield pavement condition index (PCI) survey on Air Force owned airfields using drone mounted imaging technology. Intermediate results from different stages of field testing over an auxiliary airfield located at the Air Force Academy in Colorado Springs, CO are presented and discussed in detail. Ultimately, the automated data collection and analysis developed by this study produced a PCI value of 56.5, which strongly agrees with manual inspection results that calculated a PCI value of 54 for the same runway. Also presented is a fiscal analysis of the autonomous method being proposed. Using uncertainty analysis and Monte Carlo simulation, cost estimates are given for replacing manual PCI inspections with an autonomous solution across a large number of airfield pavement assets. These estimates provide economic insights into factors that affect technological development and implementation and suggest that replacing manual methods with an autonomous system could reduce inspection costs roughly 25%.

Thesis Supervisor: Herbert H. Einstein
Title: Professor of Civil and Environmental Engineering

Acknowledgments

I would first like to thank Professor Herbert H. Einstein, whom I spoke to for the first time on January 31st of 2020 when he called to inform me that I had been admitted into MIT. Professor Einstein has been my biggest advocate over the last two years and is a mentor in my life whom I hold in the highest regard. I cannot adequately express my gratitude. He has championed my research and given me the freedom to pursue my interests, while also providing me with the resources, funding, and wisdom to transform those ideas into tangible results. As a student in his classes and a researcher in his lab, he has taught me to think critically, organize, write, and communicate at a level far beyond my previous abilities. He has read hundreds of pages of my work and given thoughtful, detailed, feedback in every paragraph. And while all these professional contributions are deeply appreciated, what I will remember most is the kindness and care he has shown me, not just as a researcher or student, but as a person. Professor Einstein has always been intentional about making the lab group feel like family and from my first meeting I felt like I was welcomed and belonged. Thank you.

I am also forever grateful to Professor Melissa Beauregard at the United States Air Force Academy. As a sophomore in college I was first introduced to geotechnical engineering in her class and was also introduced to the exciting world of academic research through an independent study course with her later that year. Somehow we parlayed that one semester of independent study into a 2.5 year research project. During this time she gave me the guidance, skills, and opportunities to develop as an academic, to write and speak at conferences, to apply for funding/grants, and to apply to grad schools. She had an open door policy on helping students and I appreciate her for letting me abuse that generosity by constantly bombarding her with geotechnical questions and personal conversations in her office. Without her I would not have been inspired to pursue a master's, and I most certainly would not have been able to do that at MIT. After graduating the Air Force Academy she has remained a great mentor in my life, and an even better friend. Thank you for the phone calls, life updates, dinners, drinks, puzzles, Billy Joel jam sessions, and rock climbing adventures. And thank you to Chaz, Teddy, and V for letting me be a part of your wonderful family.

My gratitude also extends to all the institutions that have supported my lifelong academic journey. I want to acknowledge all my teachers at Elitha Donner Elementary School, Joseph Kerr Middle School, and Elk Grove High School and all of my professors at the United States Air Force Academy and MIT. I thank the USAFA Civil Engineering department for providing me with a passion for Civil Engineering and the facilities to conduct my research. I thank the USAF Civil Engineering Center for financially supporting the equipment needs of this research.

A very special thanks extends to my friends who brought me countless moments of joy and laughter and helped me through the personal challenges of the last two years. Miggy, you are and forever will be my best friend. Thanks for the phone calls and for listening to my long rants about life. Thanks for getting me through breakups, long weeks at work, and the moments I questioned all of my life choices. Thanks for always being ready for a new adventure. You bring more excitement and joy into my life than I could ever imagine.

I know I can always count on you for anything and I can't thank you enough for always being there for me. Spencer, thank you for being my brother, through thick and thin. I always look forward to our weekly phone calls and Bible study and although grad school has been a uniquely difficult experience I am glad to have gotten the chance to experience it with you. You are one of the toughest and most joyful people I know and your passion to excel in all aspects of your life continues to inspire me and push me to be better. You continue to hold me accountable to the person I want to become with a grace, acceptance, and love that is truly unique to you. Christy, you are an astute and caring friend who is able to empathize and understand me, even better than I am able to understand myself at times. Thank you for your constant reassurance and support over the years and a lifetime of unforgettable adventures. You have always had faith in my abilities, even when I doubted them in myself. Andrew S., you are closer than a brother to me and will understand me in a way few people ever will. Thank you for your unconditional friendship. Thank you for deep conversations. Thank you for making me a part of your family and for several incredible thanksgiving celebrations during my time here. Phil, thank you for being my friend and roommate the last two years. You are one of the few people who understand the unique struggles of being a grad student at MIT and while some moments of COVID quarantine may have tested the depths of our friendship I am immensely grateful for the memories and deep conversations we've gotten to share (not to mention all the exceptional glasses of whiskey). Hoff, thanks for being one of the most solid friends a guy could ask for. I am immensely impressed by your work ethic and your desire to improve in all areas of your life. I look up to you more than you may realize and you inspire me to constantly be pushing myself and growing. Thank you for holding me to the highest standard. Scott Alsid and Ben Paulk, thank you for your fellowship, guidance, and deep theological discussions.

Lastly I thank my family. To my big brother, Andrew, thank you for always setting a good example for me. You are a renaissance man who has pushed me to grow my intellectual weaknesses and learn more about everything, not just science and engineering. Thank you for the rich theological discussion and the book recommendations. You make me proud to be your brother. To my Mom and my Dad, I cherish your unconditional love and support. Thank you for giving me every opportunity in life to pursue my dreams and encouraging me every step of the way. Thank you for the phone calls and for your continued wisdom and advice. I love you both dearly. And thank you to my fiancé, Hannah, and my future daughter Daphne. You two are the greatest joys of my life and I thank God for the blessing of meeting you two during my time here. I look forward to spending the rest of our lives together.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context: Airfield Pavements and Infrastructure Maintenance

Aviation is a $3.5 trillion dollar/year industry (ATAG, 2020) that is a cornerstone of the modern way of life. Supporting this industry are the billions of square feet of airfield pavement that enable aircraft to take-off, land, taxi, and park. The U.S. alone spends nearly $4 billion dollars on building or maintaining airfield pavement runways, taxiways, and aprons each year (NAPA, 2014). Broken into two broad categories of material: (1) flexible pavement (typically asphalt concrete), and (2) rigid pavement (typically Portland cement concrete), the runway pavement has to accommodate planes with different requirements for loading and wheel configuration to operate safely and repeatedly (Mallick and El-Korchi, 2018). Throughout the pavement's lifespan, damages and distresses requiring maintenance develop within the pavement's structure and on its surface and are identified, categorized,

and addressed using on information obtained from inspection. While natural surface runways exist (dirt, gravel, ice, grass, etc.), they are not included in the discussion presented in this paper.

1.1.1 Infrastructure Inspection and Maintenance

Infrastructure inspection involves evaluating a structure to asses its safety and serviceability, thereby providing a system's operator with valuable information required for maintenance and upkeep (Fenves, 1984). Inspection and maintenance constitute a sizable, but necessary, portion of life-cycle costs for infrastructure projects (Das et al.). To conduct on-site inspections requires trained professionals, increasing the costs associated with this process, while introducing human error, and posing a safety risk for the inspector (Lattanzi and Miller, 2017; Ellenberg et al., 2016; Henrickson et al., 2016). This added cost and degree of possible error is pushing many different industries toward integrating technologies that can assist or completely automate the inspection process (Agnisarman et al., 2019). Common structures like pavements, tunnels, dams, and bridges all require inspection at regular intervals to find distress indicative of possible or impending failure (Lattanzi and Miller, 2017). Performing both reactive and preventative maintenance in response to these inspections oftentimes reduces the overall cost of major rehabilitative expenditures, while also ensuring continued safety and serviceability (Symons, 2010).

As an organization, the US Air Force has a critical interest in maintaining nearly 2.2 billion square feet of airfield pavement assets (Ford, 2020) valued at roughly $20 billion (Kemeny, 2018). The current regulations for ensuring the serviceability and mission ready

status of airfield pavements are mostly conducted by the Air Force Airfield Pavement Evaluation (APE) Team. This team is responsible for evaluating the Air Force's airfield pavements for over 200 installations, enabling 1.2 million hours of aircraft flight missions annually (Ford, 2020). Under their current operating model, the organization conducts roughly 20 comprehensive evaluations of airfields a year (Ford, 2020). Consequently, the APE evaluation cycle for any given airfield is approximately once every 10 to 12 years (Ford, 2020).The tests conducted by the APE team include Airfield Pavement Structural Evaluation, Runway Friction Characteristics Evaluation, Pavement Condition Index (PCI) Survey, Power Check Pad Anchor Test, and Foreign Object Debris (FOD) index (USAFCE and Cooper, 2017). These tests are then analyzed and summarized in reports and engineering assessments that outline the current condition of the runway and provide recommendations for operation and maintenance projects (USAFCE and Cooper, 2017). While this model for pavement inspection has proven functional, there is benefit to having more current information on a pavement's condition readily available. While engineers stationed locally at bases can help supplement APE team efforts with additional inspection, as well as preventative and reactive maintenance, there are limitations regarding manning, cost, operational disruption, and subjectivity. Ideally, a new technologically augmented inspection system could address these shortcomings to improve work efficiency, extend the overall lifespan of the pavement, and provide increased confidence in continued mission serviceability. To address this problem, the Air Force has begun investigating the use of drone mounted imaging to more frequently inspect runways.

1.1.2 Recent Technological Advances

The development of semi-autonomous pavement evaluation systems has grown significantly in recent decades, employing a wide variety of data acquisition devices mounted on a diverse range of platforms. The published research on pavement evaluation using passive remote sensors involves image analysis of pavements obtained in highly controlled or strictly experimental settings (Manzo et al., 2014; Herold et al., 2008), from satellites (Ozden et al., 2016; Mohammed, 2017), from surface vehicle mounted systems (Rowe et al., 2002; Chan et al., 2016; Wang et al., 2015), and from unmanned aerial vehicles (Henrickson et al., 2016; Inzerillo et al., 2018). Many of these early efforts, while showing positive results, were not focused on detecting specific damages, but rather broad categorical classifications of entire images and road segments. While this may be sufficient for highways and roads, airfield pavement management could benefit significantly from greater specificity. More recent efforts, both in general infrastructure maintenance, and pavements in particular, have focused on manually overseen and partially autonomous crack segmentation from high resolution images in an attempt to identify which specific pixels within an image represent a crack. These investigations still struggle to overcome challenges inherent to pavement distress features like crack heterogeneity, feature sharing with non-crack objects (like joints and chipped paint), and highly variable surficial texture (Gopalakrishnan et al., 2017).

Unlike highways, which have less stringent failure criteria, airport pavements are held to a comparatively higher standard due to their loading patterns and failure tolerance. As a result, the nature of airport maintenance is more preventative than that of highways,

making detailed and frequent monitoring and intervention far more critical. Identifying and quantifying specific pavement distresses and their locations is therefore needed to facilitate pothole or spalling repair (Figure 1-1), foreign object debris removal, slab replacements, crack sealing, joint resealing, surface treatments, localized skin patching, or any other form of maintenance (Symons, 2010).

Figure 1-1: Air Force personnel work on repairing a section of runway. Image courtesy of: https://www.defense.gov/News/News-Stories/Article/Article/982954/air-force-engineers-repair-runway-in-iraq/

1.2 Research Objectives

The purpose of this research is to develop and assess a novel, partially autonomous, drone mounted imaging system and detection algorithm that can conduct segmentation based distress detection on an Air Force operated asphalt concrete runway at the United States Air Force Academy in Colorado Springs, CO. The output of this system should be a Pavement Condition Index (PCI) approximately equivalent to manually conducted PCI results.

Achieving this objective involves the use of supervised machine learning algorithms that are able to learn how to perform detection tasks from training data. Because the data required to perform this pavement distress detection task do not yet exist for airfield pavements, a major contribution of this study is providing an initial machine learning data set and the methodological framework required to expand algorithm performance to other runways in different conditions.

This research focuses detection objectives around cracking and patching distresses only and excludes all other damages normally included in a PCI evaluation for asphalt concrete pavements. The method proposed was not intended to meet the inspection criteria for rigid pavements, which slightly differ from flexible pavements; however, small modifications to the system's operations and code could theoretically accommodate these differences. This study presents initial validation of the proposed method and insight into future developments, which is a path toward expanding this approach to include additional distresses, more airfield locations, and full automation. Finally a fiscal analysis of Air Force pavement inspection that models replacing existing practices with an automated system is included to provide insight into the economic desirability of a fully functioning autonomous system. This research primarily focuses on adapting existing technologies to a novel application. Including a fiscally focused system analysis provides a complete discussion of automated airfield pavement evaluation, from technological development to real world implementation.

1.3 Thesis Outline

To begin approaching the research objectives detailed in section 1.2, Chapter 2 starts with an understanding of the physical problem by reviewing the basic structure of pavements and how they deteriorate and eventually fail. Then a summary of current inspection and maintenance practices will contextualize the problem and a detailed review of the PCI process will provide the guiding framework for technological design and performance requirements. A literature review of technologies used for similar structural inspection applications will provide inspiration for potential approaches, while a review of drone systems and computer vision algorithms will provide a strong understanding of the tools available for best accomplishing this task.

Chapter 3 introduces Machine Learning (ML) algorithms and how they are taught to perform a task. It also reviews the state of the art models available for image detection tasks and provides a discussion on which options best suit the needs of this project.

Chapter 4 uses this background to develop the methodology used to complete the inspection objective of this research. An overview of system hardware, flight path optimization, and data acquisition methodology is presented, followed by a detailed analysis of the Convolutional Neural Network (CNN) used and how the algorithm was trained. The performance of the trained Neural Network is presented and the process for extracting PCI values from the Neural Network's outputs is summarized. This chapter concludes by comparing the results of manually conducted PCI calculations with those derived by the partially autonomous method being proposed.

Chapter 5 takes a step back from the technological details and analyzes the autonomous

method being proposed through a fiscal lens. Using uncertainty analysis and Monte Carlo simulation, this chapter constructs a model that estimates the cost of replacing manual PCI inspections with a competent autonomous solution across a large number of airfield pavement assets. The technological system may still be in early stages of development; however, this hypothetical implementation analysis gives insight into some key parameters that influence future design, while roughly estimating potential economic benefits of continuing this research to a robust implementation.

Chapter 6 concludes this discussion by providing a commentary on the results and limitations of this body of work. The future work required to improve the initial design concept validated by this research is also proposed.

Chapter 2

Background

2.1 Airfield Pavement

2.1.1 Structure

Though designed to different standards based on use, pavements are structures that meet a specific transportation need. Pavements are generally classified as either flexible or rigid (Figure 2-1) depending on their material composition. Their purpose is to withstand loads without excessive deformation to provide vehicles (or people) with a smooth functional surface to travel over under different environmental conditions.

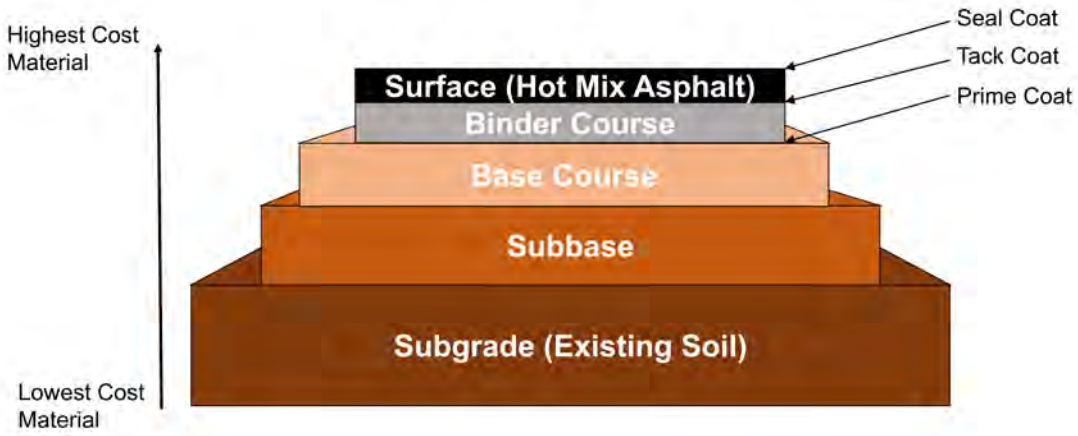
Flexible and rigid pavements are comprised of a series of asphalt, aggregate, and soil layers that vary in thickness and material to provide a cost efficient manner of meeting loading demands for the duration of a pavement's design life. These layers, sometimes referred to as "courses", are designated as either subgrade, subbase, base, binder, or surface and are typically arranged in the configurations similar to those shown in Figures 2-2 and

(a) Image courtesy of: https://www.airport-technology.com/contractors/apron_clean/asi-solutions/

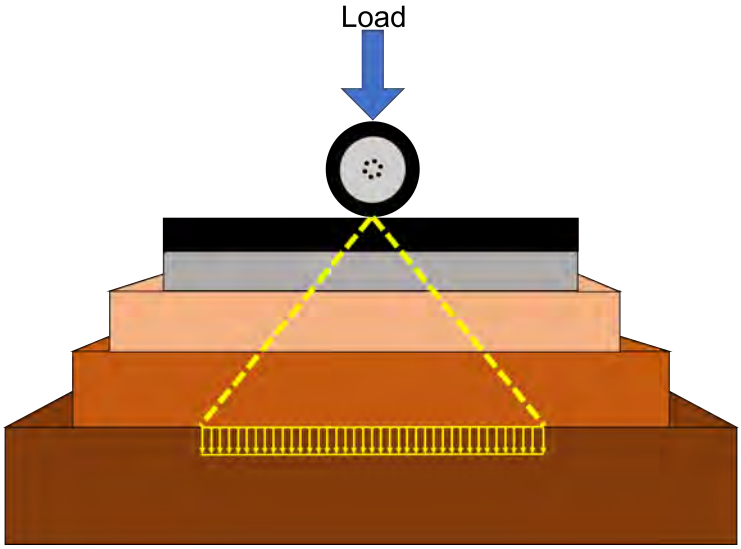
(b) Image courtesy of: https://qph.fs.quoracdn.net/main-qimg-e6334f118c1d34c3bf54bf1ebe362377

Figure 2-1: (a) Flexible pavement runway constructed from Asphalt Concrete (AC) (b) Rigid pavement runway constructed from Portland Cement Concrete (PCC).

2-3. In these configurations, the cost of a material generally increases the closer it is to the surface and the best designs specify the thickness and material properties of each layer to satisfying loading, drainage, workability, and serviceability requirements in the most cost-efficient manner possible.

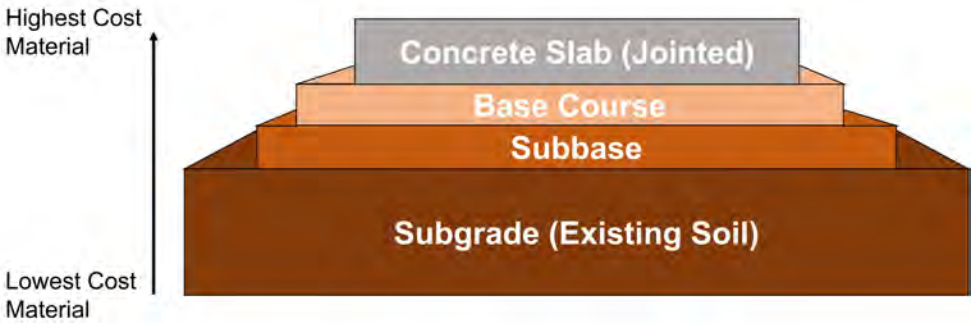


(a) Typical flexible pavement structure

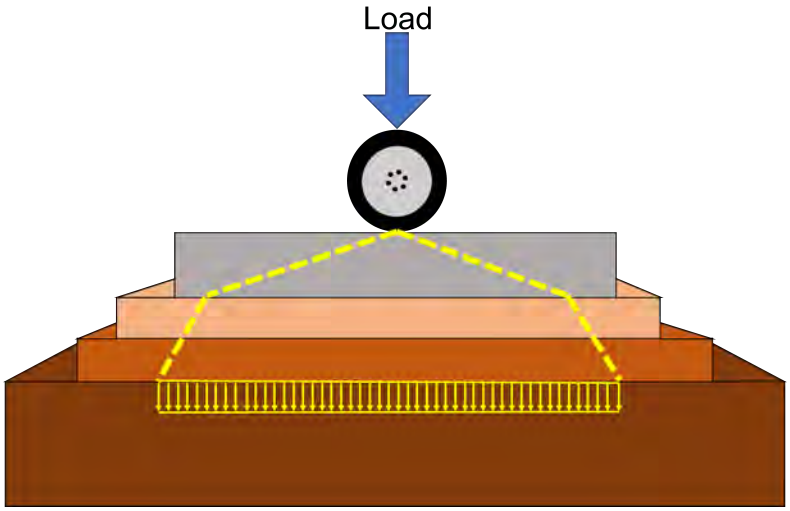


(b) Structural load distribution in a flexible pavement

Figure 2-2: (a) Flexible pavement pavements typically have multiple thick layers over subgrade with special coatings in between the layers to help with layer bonding or water infiltration. The surface coat is usually hot mix asphalt. (b) In flexible pavement, the surface layer of asphalt concrete does not distribute a significant portion of the load and therefore relies heavily on the thickness and composition of the subsequent layers to adequately distribute loading to the subgrade.



(a) Typical rigid pavement structure



(b) Structural load distribution in a rigid pavement

Figure 2-3: (a) Rigid pavements normally have a thicker layer of Portland cement concrete (PCC) at the surface, but generally require thinner subbase and base course layers than flexible pavements. (b) Load distribution in a rigid pavement is primarily accomplished by the PCC surface layer before being transferred into the base course, subbase, and ultimately the subgrade.

Ultimately, when discussing pavement design and management, regardless of the use, Mallick and El-Korchi (Mallick and El-Korchi, 2018) summarize the most important issues for an engineer to be aware of:

1. Drainage is needed to drain water quickly and effectively away from the pavement.
2. The materials must be evaluated and selected properly so that they can withstand the effects of the traffic and the environment.
3. The mix must be designed properly such that it can withstand traffic and environmental factors.
4. The structure should be designed properly such that it has adequate thickness to resist excessive deformation under traffic and under different environmental conditions.
5. The pavement must be constructed properly such that it has desirable qualities.
6. The pavement must be maintained/managed properly through periodic work, regular testing, and timely rehabilitation.
7. Sustainable technologies must be continuously incorporated in road construction process.
8. Generation of knowledge through research is critical for ensuring good pavements in the future.

2.1.2 Airfield Pavement Design

Airfield pavements are designed for three different use areas: (1) runways, (2) taxiways, and (3) ramps/aprons (Figure 2-4). A runway is used for aircraft landing and takeoff, taxiways are used for connecting runways to aprons, hangers, terminals, and other facilities, and aprons are used for aircraft loading, unloading, and parking. Design differs based on use because the structural loading requirements, airplane paths, and frequency of use changes based on area type. For example, on an apron, planes will be moving at low speeds along many different paths to a choice of terminals or parking spots; however, on a runways airplanes are traveling fast or braking along a single designated path repeatedly. Over

a 20-year design period, the runway pavement will undergo more loading cycles and be much more heavily worn along a singular path, therefore has different design requirements. The Federal Aviation Administration (FAA) imposes strict standards unique to airfields to

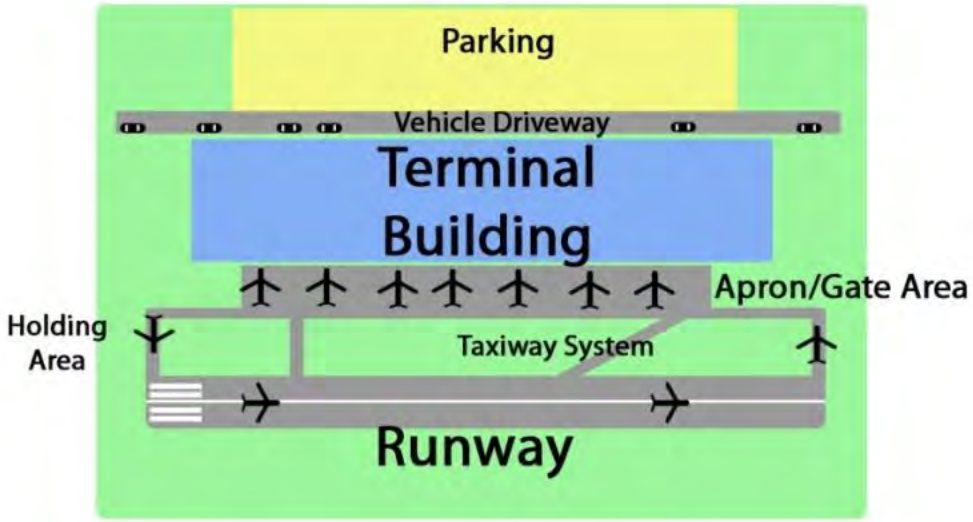


Figure 2-4: Airfield pavements have three main classification based on use: (1) runways, (2) taxiways, and (3) aprons. Runways are used for landing and takeoff, taxiways for connecting runways with other facilities, and aprons for aircraft parking and loading/unloading. Image courtesy of: `https://www.aboutcivil.org/airport-components`.

guide concrete mix and pavement layer design to avoid damage to aircraft and passengers. The climate, the characteristics of applied loads, the subgrade (natural soil on site), and the paving materials (subbase, base, and surface layers), are the four major components considered in the design process to achieve the following results:

> Airport pavements are designed and constructed to provide adequate support for the loads imposed by airplanes and to produce a firm, stable, smooth, skid resistant, year-round, all-weather surface free of debris or other particles that can be blown or picked up by propeller wash or jet blast. To fulfill these requirements, the quality and thickness of the pavement must not fail under the imposed loads. The pavement [mix design] must also possess sufficient inherent stability to withstand, without damage, the abrasive action of traffic, adverse weather conditions, and other deteriorating influences. This requires coordination of many design factors, construction, and inspection to assure the best combination of available materials and workmanship (FAA, 2016)

For flexible and rigid pavements, the FAA method for design is a layered elastic and finite element (FE) based process specified in AC 150/5320-6F (FAA, 2016), which in 2009 replaced AC 150/5320-6E (FAA, 2009), an empirically-based spreadsheet design method previously governed by California Bearing Ratio (CBR) values.

The current method for flexible pavements uses the software FAARFIELD 1.41 to design pavement layers that fulfill the failure criteria of rutting (caused by compressive strain responses at the top of the subgrade - Figure 2-7) and bottom-up fatigue cracking (caused by tensile strain at the bottom of the asphalt surface layer - Figure 2-6). This software requires aircraft traffic and an initial pavement structure as inputs. The aircraft traffic inputs specify how many loading cycles each type of aircraft is predicted to have on the pavement during its lifespan. Each type of aircraft is uniquely considered so the model is able to consider the gear configuration and load distributions specific to each aircraft. The requirements of the initial pavement structure input are shown in Figure 2-5. Then the model adjusts pavement layer thickness until the pavement structure is able to withstand loading requirements for the entirety of the structure's design life.

Rigid pavements are poured in slabs, rather than being continuous like flexible pavements. Therefore horizontal stresses at the edge of each slab are considered in addition to internal stresses in the middle of the slabs. Like flexible pavements, these stresses are determined using the aircraft traffic information as input. The model determines pavement layer thickness by using a finite element approach and selecting the working stress for design as the highest value of the following three computations: (1) the interior stresses, (2) 75% of the free edge stress obtained with gear configurations oriented parallel to the slab edge, and (3) 75% of the free edge stress obtained with gear configurations oriented parallel

Thickness	Pavement Structure
4 inches	P-401 Asphalt Surface Course
5 inches	P-401/P-403 Stabilized Base Course
6 inches	P-209 Crushed Aggregate Base Course
12 inches	P-154 Aggregate Base Course
	Subgrade, CBR=5 (E = 7500 psi)

Figure 2-5: The FAARFIELD pavement design software is based on an iterative approach, requiring this information as an initial input for the pavement structure. Then, using the aircraft traffic and pavement structure information provided, layer thicknesses is adjusted so the pavement satisfies a specified design life (FAA, 2021a).

to the slab edge. More detailed explanations, as well as examples of the rigid and flexible pavement design process, can be found in AC 150/5320-6G (FAA, 2021a).

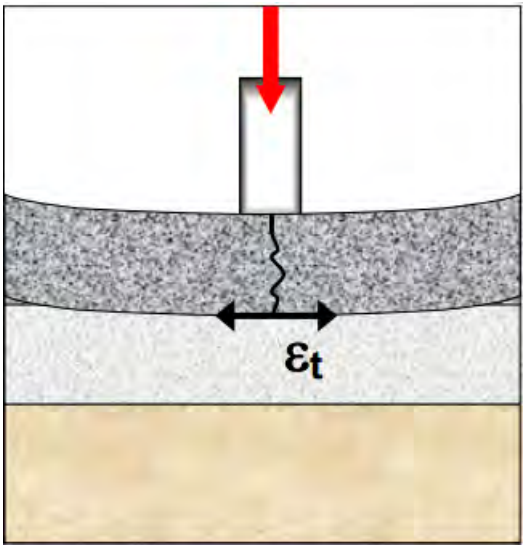


Figure 2-6: Flexible pavements are designed to resist fatigue cracking, which occurs when cyclic loading causes excess tensile strain on the bottom of the asphalt layer. Image courtesy of: https://www.ukessays.com/essays/engineering/fatigue-cracking-pavement-2611.php#citethis.

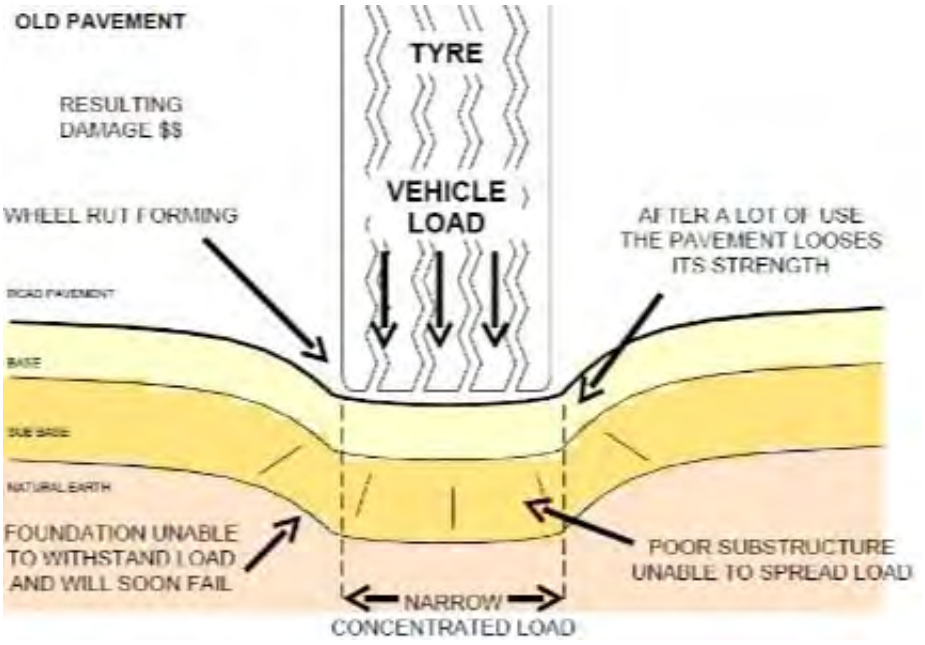


Figure 2-7: When the load on a pavement is insufficiently distributed by the preceding layers, after enough cycles of loading the subgrade experiences noticeable compressive strain, which is identified on the surface as rutting. Image courtesy of: https://theconstructor.org/transportation/types-failures-in-flexible-pavements-repair/16124/.

2.2 Maintenance Practices

While dependent on size, the initial cost of an airfield pavement structure typically ranges on the order of millions to tens of millions of dollars (Gibson et al., 2011). Due to repeated loading and environmental factors like freeze-thaw cycles and water infiltration, pavements all degrade with time and begin to exhibit signs of distress. After construction, pavement tends to perform well for much of its service life, until "critical condition" is reached, and deterioration occurs rapidly as shown in Figure 2-8.

The FAA reports that preventative maintenance (summarized in Table 2.1) and rehabilitation costs for good pavement prior to critical condition greatly extend pavement life (Fig. 2-9) and are four to five times less expensive than efforts for pavements that have already reached "fair" and "poor" conditions (FAA, 2014b). While the details of different models for optimal preventative maintenance vary, the literature supports the conclusion that the long-term life cycle cost of regular maintenance and preventative upkeep is significantly less than that of full rehabilitation after deterioration (Irfan et al., 2015); however even the most sophisticated models fail to predict the optimal rehabilitation point exactly. Consequently, continued inspection and current pavement condition data is vital to developing and updating the most economic and informed pavement maintenance plan. Specialized data management systems, like "PAVER", are sometimes used to assist in organizing and analyzing PCI data across a large network of pavement assets to provide information and insight guiding maintenance practices.

Table 2.1: Different maintenance techniques can be employed to extend the life and serviceability of an airfield pavement before full rehabilitation in required (Hajek, 2011).

Maintenance Techniques for Airport Pavement		
Flexible Pavements	**Rigid Pavements**	**Flexible and Rigid Pavements**
1) Sealing and filling of cracks	1) Joint and crack sealing	1) Shot Blast Texturization
2) Small area patching	2) Partial-depth repairs	2) Diamond grinding
3) Spray patching	3) Full-depth repairs	3) Microsurfacing
4) Machine patching with AC material	4) Machine patching using hot mix	
5) Rejuvenators and seals	5) Slab stabilization and slabjacking	
6) Texturization using fine milling	6) Load transfer	
7) Surface treatment	7) Crack and joint stitching	
8) Slurry seal	8) Hot-mix overlays	
9) Hot-mix overlay	9) Bonded PCC overlay	
10) Hot in-place recycling		
11) Cold in-place recycling		
12) Ultra-thin whitetopping		

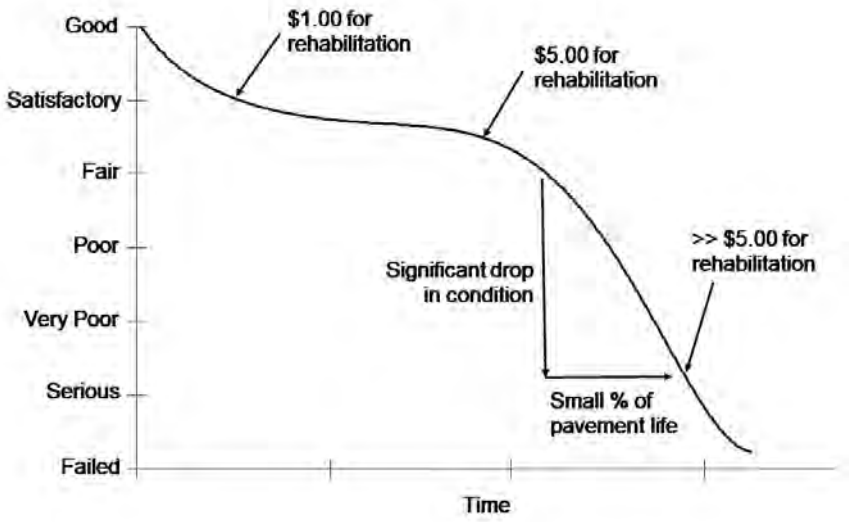


Figure 2-8: Standard life cycle of airfield pavements with relative rehabilitation costs. If regular maintenance is conducted early, under good pavement conditions, studies show that the same rehabilitative outcomes can cost more than five times less than if they are delayed (FAA, 2014b).

2.2.1 PAVER

Developed in the late 1970s for the Department of Defense by Colorado State University, PAVER™ is a software for organizing, managing, and analyzing pavement inspection and Pavement Condition Index (PCI) data across a network of pavement related assets. It is capable of managing airfield, road, and parking lot pavement information and performs multiple levels of analysis to guide the fiscal allocation process for maintenance and repair resources across local and remote databases. The software can assist and augment traditional inspection through the use of integrated global information systems (GIS) and global positioning system (GPS) integration that provide tools like geo-referenced pavement network maps. The US Air Force, US Army, US Navy, Federal Aviation Administration (FAA), and Federal Highway Administration (FHA) are all major supporters of this

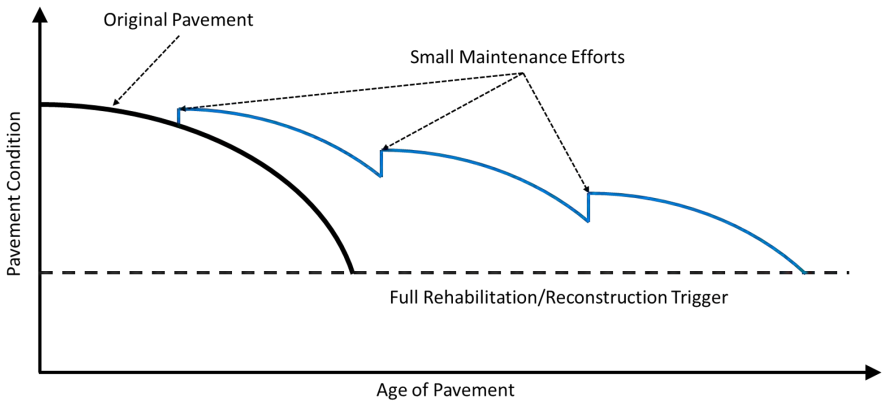


Figure 2-9: Lifecycle effect of pavement preservation maintenance and upkeep. If small, routine maintenance efforts are consistently implemented the pavement structure lasts longer and total costs are reduced (FAA, 2014b).

platform (CSU).

2.3 Inspection Practices and Airfield Pavement Condition Index

The Airfield Pavement Condition Index (APCI), originally developed by the Army Corps of Engineers for use by the US Air Force (Shahin et al., 1976; Kohn and Shahin, 1984), has been adopted and verified by the FAA (FAA, 2014a) and the U.S. Naval Facilities Engineering Command (USN, 1988), and is one of the most important metrics for quantifying rigid and flexible airfield pavement condition. This visual evaluation method is indirectly related to, but not intended to replace, direct roughness, structural capacity, texture, or friction measurements. In addition years of empirical data collected made it possible to relate the PCI score to maintenance and repair requirements (ASTM, 2020).

On asphalt pavements this process works by dividing a runway into branches based on function and sections based on construction, maintenance, usage history, traffic volume, load intensity, and condition. Then using probability distribution requirements of a 95% confidence interval, the number of sample units to survey from each section is determined. Figure 2-10 displays the relationship between airfield branches, sections, and sample units in the PCI process. Each sample unit is inspected visually, including measurements of asphalt pavement distresses categorized by type (Table 2.2) and severity: either low, medium, or high. All pavements start with an initial score of 100. Then a tabulated metric known as a ”deduct value” is determined for each distress type and severity based on a measurement of the distress. Deduct values are taken through a series of calculations before ultimately subtracting from the initial score of 100, thereby yielding the PCI value. This is done for each inspected sample unit, and then averaged for the entire section. A PCI of 100 in-

Table 2.2: All distresses included in an asphalt pavement PCI are listed in the table below, along with the corresponding distress code and unit of measurement. (An airfield PCI for concrete pavement has different distresses, the details of which can be found in ASTM-D5340 (ASTM, 2020))

Distress Code	**Distress Name**	**Unit of Measurement**
41	Alligator or Fatigue Cracking	Square Feet
42	Bleeding	Square Feet
43	Block Cracking	Square Feet
44	Corrugation	Square Feet
45	Depression	Square Feet
46	Jet Blast Erosion	Square Feet
47	Joint Reflection Cracking	Linear Feet
48	Long./Trans. Cracking	Linear Feet
49	Oil Spillage	Square Feet
50	Patching & Utility Cut Patch	Square Feet
51	Polished Aggregate	Square Feet
52	Raveling	Square Feet
53	Rutting	Square Feet
54	Shoving	Square Feet
55	Slippage Cracking	Square Feet
56	Swell	Square Feet
57	Weathering	Square Feet

dicates an airfield pavement in perfect condition, while a score of 0 indicates the worst possible condition. This process is similar, though slightly different for concrete pavement airfields. While the thresholds for rating scales can vary by user, two of the most common are summarized in Figure 2-11 (ASTM, 2020).

Inspector training and certification protocols exist within some organizations to reduce human error; however, it is important to understand the degree of variability in manually conducted PCI test results. ASTM guidelines for distress type and severity help determine classifications; however studies of expert rater variation have concluded that even with experience and formal training, the standard deviation of PCI scores for a section of pavement, surveyed by multiple inspectors, can be as high as 17. (Prakash et al., 1994; Bogus et al., 2010; Andrei and Arabestani). While PCI is reported as a quantitative numerical value, understanding the degree of precision associated with traditional inspection is important when evaluating alternative methods.

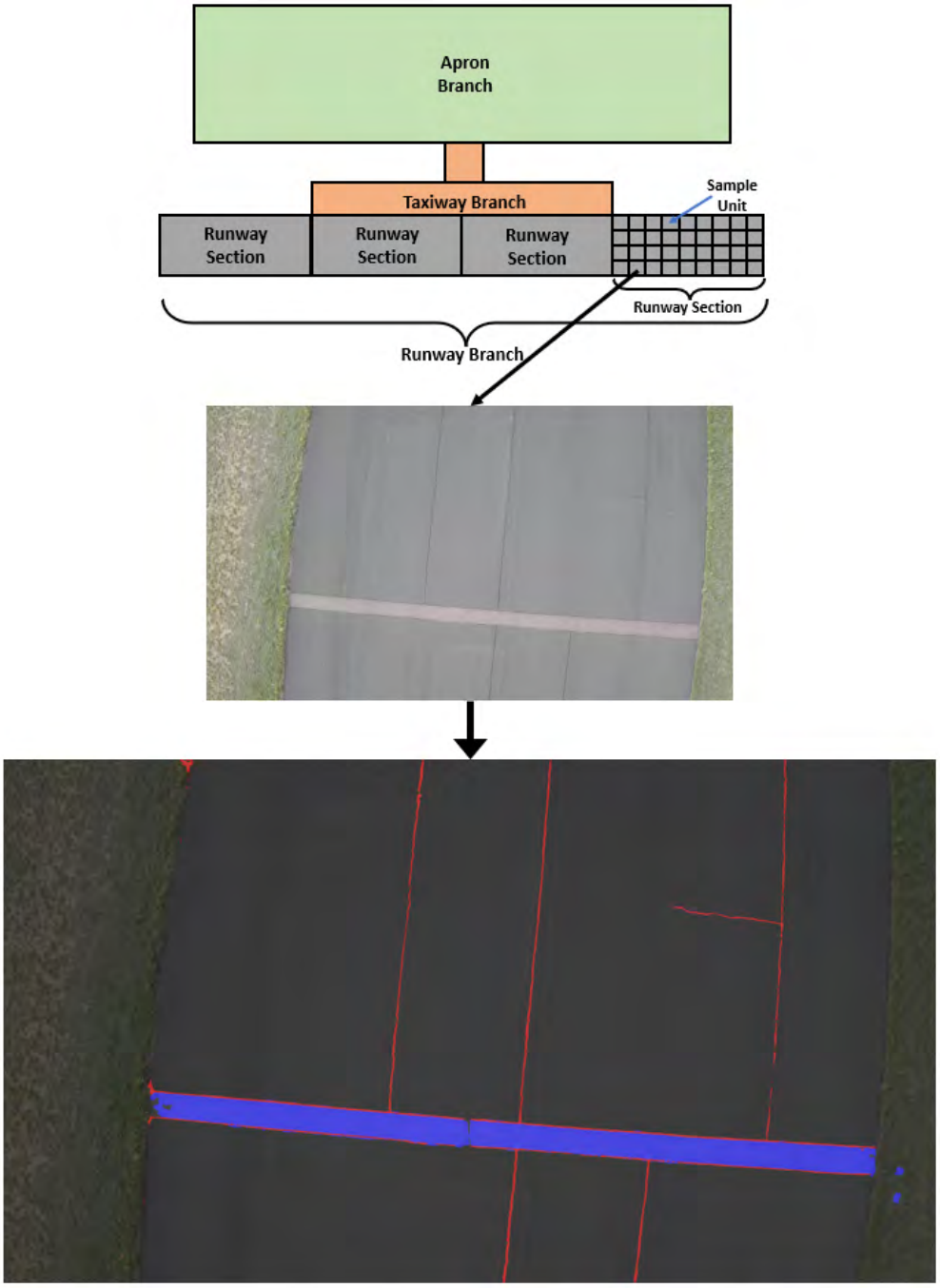


Figure 2-10: PCIs require an airfield pavement be divided into branches based on use. Branches are divided into sections based on construction, maintenance, usage history, traffic volume, load intensity, and condition. Sections are divided into sample units, and these units are statistically sampled to obtain a PCI for the section. The pictures on the right of the figure display a sample unit from a runway section of the Aardvark airfield with longitudinal/transverse cracks highlighted in red and patching in blue.

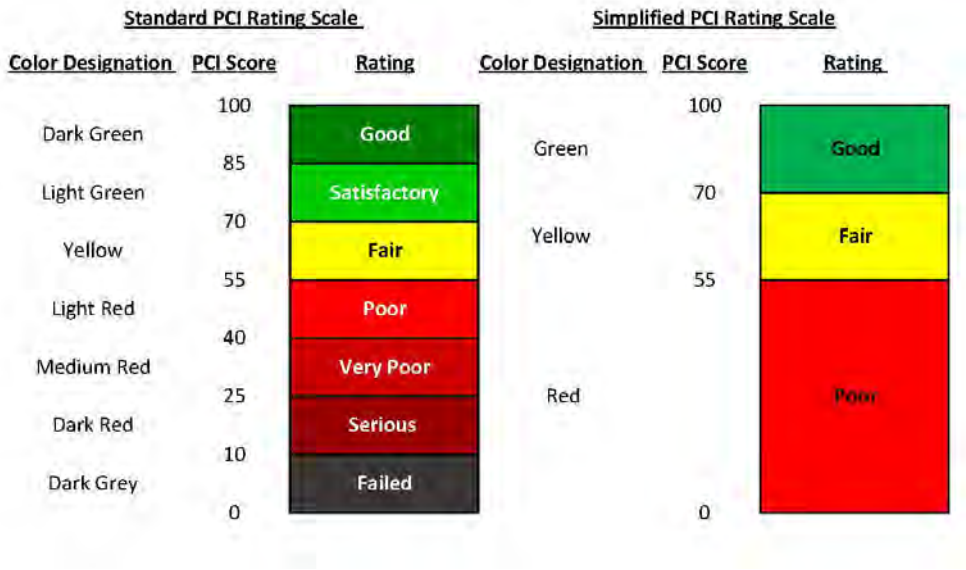


Figure 2-11: The Standard PCI Rating Scale (left) and a custom Simplified PCI Rating Scale (right) are shown above, providing examples of how numerical PCI scores are often interpreted qualitatively (ASTM, 2020).

2.4 Previous Work in Image-Based Pavement Evaluation

The majority of published research on technologically assisted pavement evaluation has revolved around analyzing images obtained from satellite, vehicle mounted, and aerial platforms to assess roadways or parking lots. Some of the earliest efforts focused on determining general classifications of pavement deterioration using reflectance measurements over a wide field of view and were not focused on detecting individual pavement distresses. In these studies the reflectance measurement represented the light intensity reflected by the pavement under natural lighting conditions, measured continuously across a range of electromagnetic wavelengths (350nm to 2450nm). These studies found that the pavement's spectral signature (or the shape of the reflectance curve) remained relatively consistent with respect to age and number of distresses. The overall magnitude of the reflectance across the entire range of measured wavelengths; however, increases with pavement aging due to the gradual loss bitumen (Figure 2-12), but decreases when cracks and other distresses are present (Figures 2-13 & 2-14). This is because as the black bitumen is lost from the pavement over time, the pavement is less dark and reflects more light. Distresses; however, are generally darker sections of pavement, resulting in lower overall reflectance across all wavelengths. These competing mechanisms influencing reflectance made it challenging to determine even general classifications consistently (Herold and Roberts, 2005; Mettas et al., 2015; Mei et al., 2014).

In an attempt to obtain more information and detail than reflectance data could provide, pattern recognition techniques correlated with spectral data were used to quantify bitumen removal and overall aggregate exposure in the form of an Exposed Aggregate Index; how-

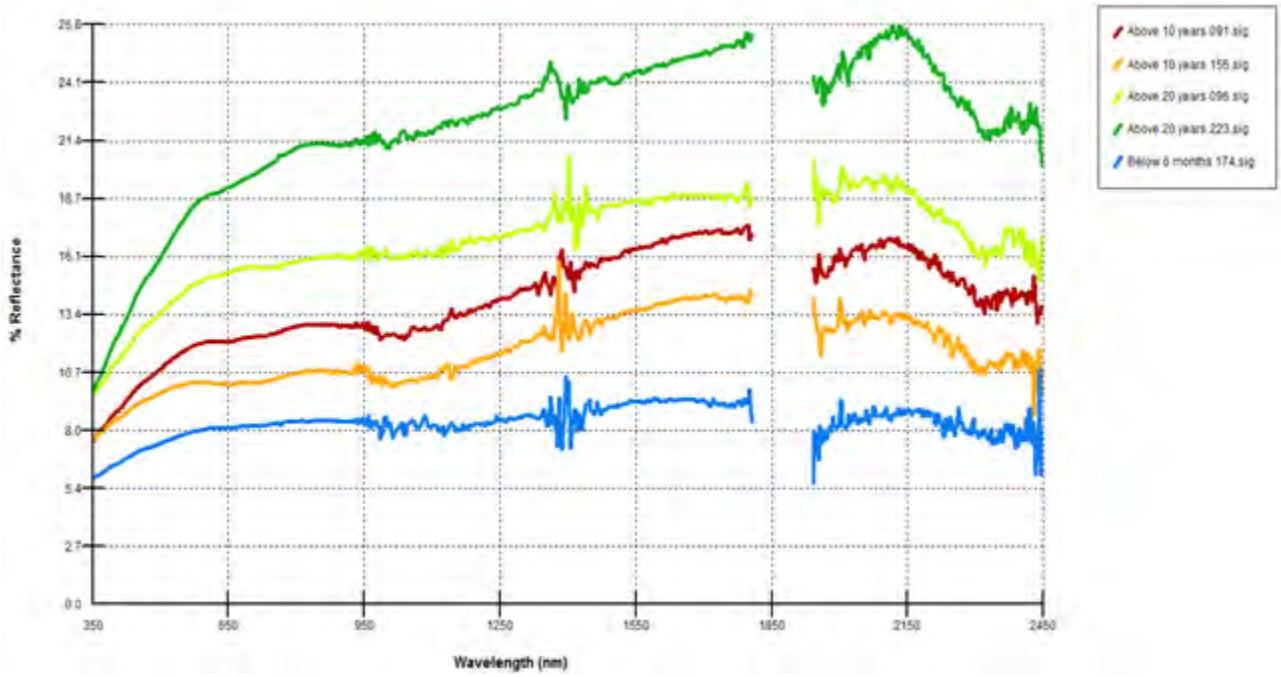


Figure 2-12: Reflectance of different aged roads with no structural damages. The oldest roads are in green and show the highest reflectance due to loss of bitumen. The youngest roads are shown in blue (Mettas et al., 2015).

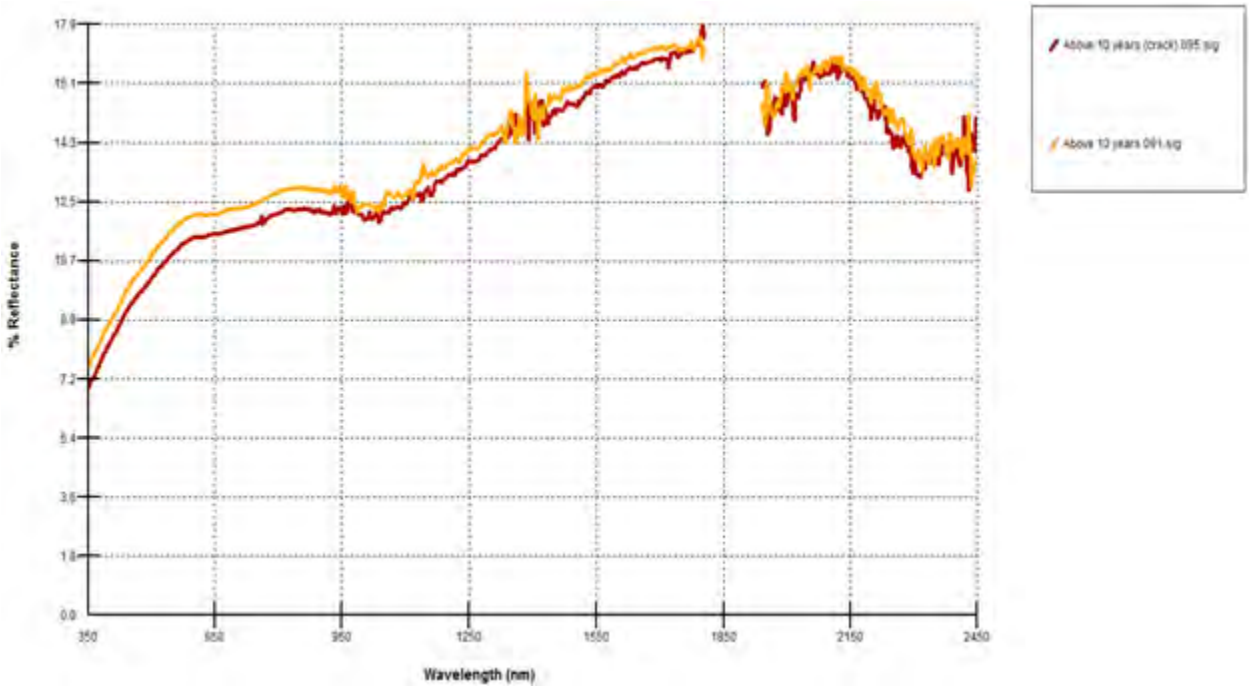


Figure 2-13: Reflectance of 10 year old road with structural damages (red) and without structural damages (orange). The structural damages cause a lower overall reflectance for a road of the same age (Mettas et al., 2015).

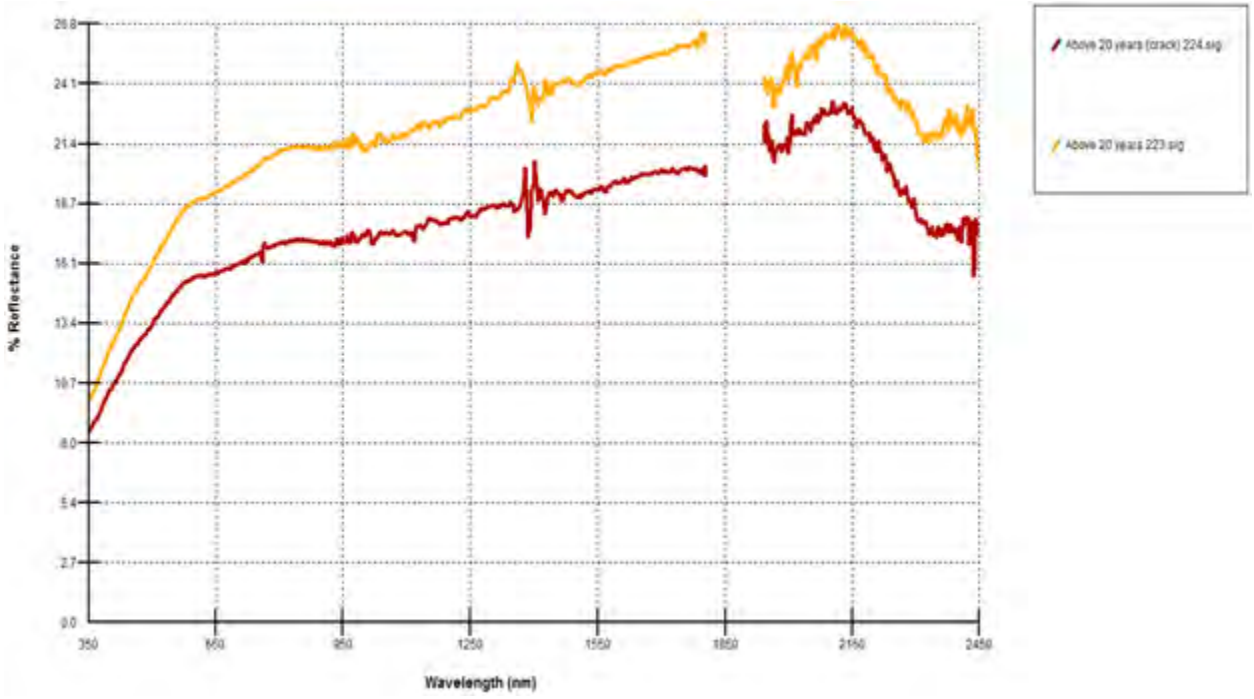


Figure 2-14: Reflectance of 20 year old road with structural damages (red) and without structural damages (orange). The structural damages cause a lower overall reflectance for a road of the same age (Mettas et al., 2015).

ever, fluctuating environmental considerations continue to present challenges in accuracy and precision (Mei et al., 2014). The influence of water particles in the air had noticeable effects on wavelengths ranging from 930-1190nm, 1350-1440nm and 2400-2500nm, and carbon dioxide within water particles further affects wavelengths from 1800-1950nm (Mettas et al., 2015). A proposed solution to this problem was to develop a massive spectral library of images and data that can be referenced under various conditions to determine pavement aging and the relative severity of pavement distresses (Mettas et al., 2016). This approach has potential to augment the speed of generalized evaluation over vast expanses of infrastructure, and has been proposed as a means of conducting preliminary roadway evaluations or identifying generalized zones of interest; however, where more detail is required for things like specific maintenance, this approach is inadequate.

While a bulk image reflectance approach has value in its simplicity and speed, it has inherent limitations regarding the specificity of information that it can provide, especially when applied to an airfield runway pavement evaluation. For this reason researchers have begun to use image segmentation and object identification techniques to begin distress identification and quantification in flexible and rigid pavements.

Research began to investigate morphological tools, neural networks, and image filtering functions to identify and quantify various pavement distresses, starting with cracks. The fundamental difficulty associated with this endeavor for pavements is that the signal to detect is very weakly represented in the context of the overall image (roughly 1.5% of the image on average). Also this signal is weakly contrasted against the background of normal, non-distressed pavement in the image (Chambon and Moliard, 2011). This has resulted in countless methods and approaches, however all still have too much false detection, lack precision in detection, lack accurate geometric analysis of the cracks, and/or simply lack the ability to be generalized over a wide array of images and conditions (Chambon and Moliard, 2011). In 2010, Tsai et al. conducted a comparative study analyzing six of the most prevalent image segmentation methods: statistical/relaxation thresholding, canny edge detection, multiscale wavelets, crack seed verification, iterative clipping, and dynamic optimization-based methods (Tsai et al., 2010). This study concluded that dynamic optimization-based methods are the most promising approach for maintaining accuracy and robust detection across different images conditions (Tsai et al., 2010); however, this conclusion was drawn in 2010, before convolutional neural networks began to improve and quickly become the leading tool in image detection applications.

In Chambon and Muliard's 2011 study, they attempt to address some of the inherent

challenges associated with wavelet morphological and wavelet filtering methodologies to enhance detection performance. When adopting a morphological approach to edge detection and object segmentation, one of the primary limitations has been finding universal threshold values that can be generally applied to a wide array of data sets (Tsai et al., 2010). Edge detection and filtering techniques based on decimated fast biorthogonal dyadic wavelet transforms, Fourier transforms, Gabor's filters, finite impulse filters, and other models are still highly sensitive to crack width variation and lighting effects, thereby lacking robust generalized applicability (Tsai et al., 2010; Chambon and Moliard, 2011). To address these difficulties Chambon and Muliard developed a more sophisticated morphological approach (named Morph) using adaptive filtering to perform wavelet decomposition coupled with a Markovian modeling segmentation (named GaMM) to account for highly variable crack geometry (Chambon and Moliard, 2011). By processing hundreds of real and synthetic images, the study concluded that for crack detection, their Morph approach obtained more true positives, while the GaMM approach was preferred for reducing the number of false positives. Although neither of these approaches reaches sufficient precision, accuracy, and capability for robust generalization, they showed early signs of detection feasibility.

While morphology, edge detection, and filtering techniques were the focus of initial attempts at damage detection, progressions in artificial intelligence and machine learning approaches for object classification and detection have begun to outperform these old methods. Although there are many types of machine learning algorithms, the primary focus of pavement detection efforts have been with support vector machines (SVM), neural networks (ANN), and random forest (RF) algorithms (Pan et al., 2018) with neural networks

being the best performing class of algorithm (Zhang et al., 2018).

2.5 Drone Systems and Related Applications

Although there are a multitude of proposed systems to classify unmanned aircraft systems by weight, wingspan, flight time, intended use, and/or construction (Brooke-Holland, 2012; Arjomandi et al., 2006; Weibel and Hansman, 2004; Zakora and Molodchick, 2014), the convention in this paper will align with the methodology summarized by Hassanalian and Abdelkefi (2017) as it is simple yet sufficiently detailed. This system distinguishes drones into a primary category first by weight and wingspan and then subdivides the classes by configuration as summarized by Figures 2-15 and 2-16.

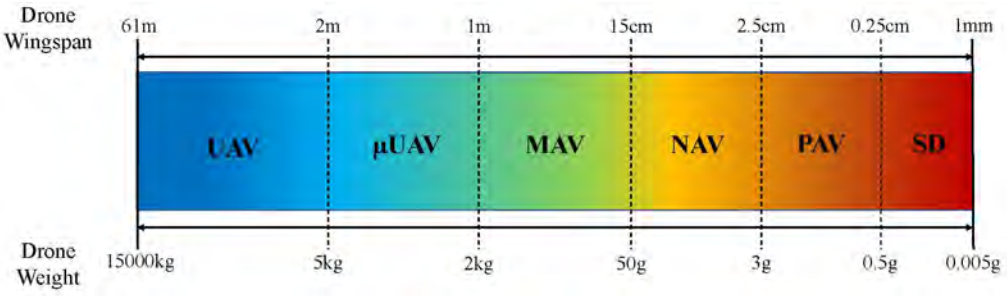


Figure 2-15: Wingspan and weight ranges for drone categories. From left to right: Unmanned Air Vehicle (UAV), Micro Unmanned Air Vehicle (μUAV), Micro Air Vehicle (MAV), Nano Air Vehicle (NAV), Pico Air Vehicle (PAV), and Smart Dust (SD) (Hassanalian and Abdelkefi, 2017).

Unmanned aircraft systems (UASs), especially MAVs, have become an increasingly popular choice among the civil engineering practitioners for their low cost, wide range of sensor compatibility, aerial perspective, and flexibility of movement (Kim et al., 2020). Multiple studies have proven their viability for visual inspection tasks, though some challenges exist with regard to the nature of airfield specific use (Kim et al., 2020; Temme and

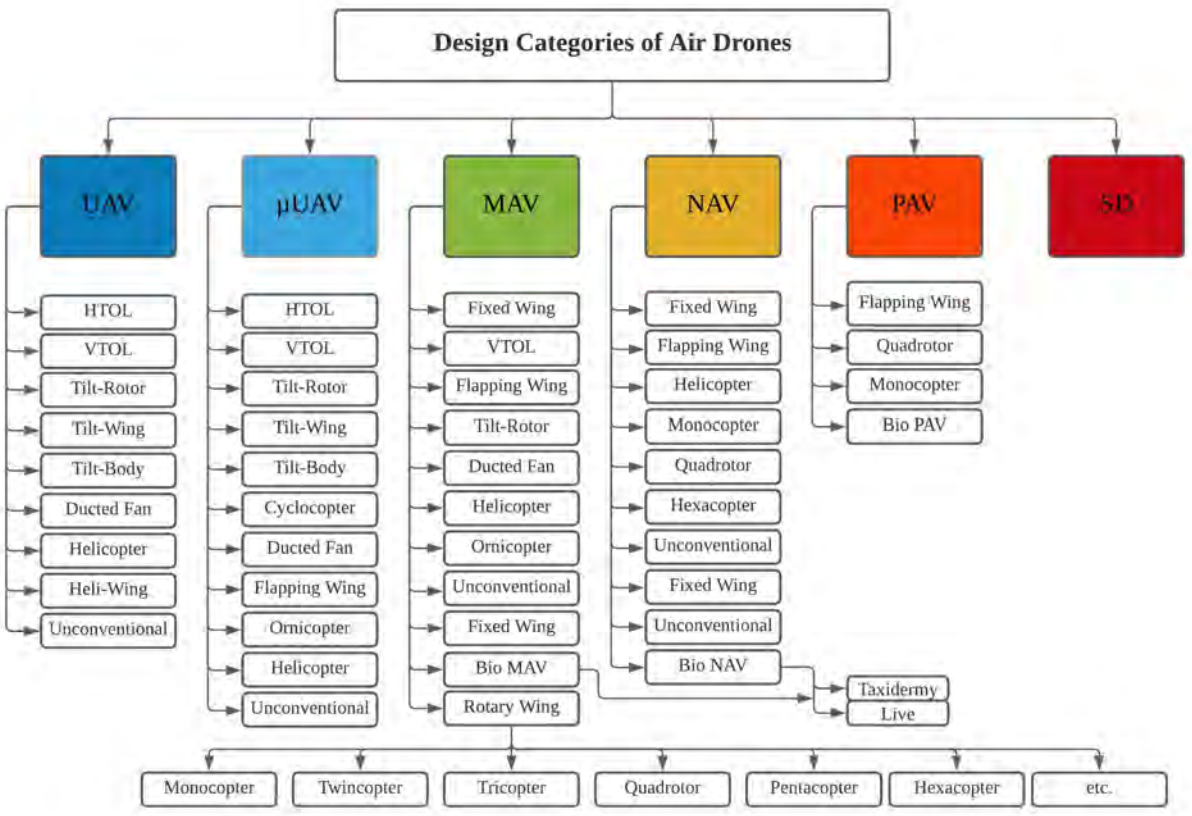


Figure 2-16: Air drone classifications by configuration (Hassanalian and Abdelkefi, 2017). Acronyms: Unmanned Air Vehicle (UAV), Micro Unmanned Air Vehicle (μUAV), Micro Air Vehicle (MAV), Nano Air Vehicle (NAV), Pico Air Vehicle (PAV), and Smart Dust (SD), Horizontal Take-Off Landing (HTOL), Vertical Take-Off Landing (VTOL)

Trempler, 2017).

While entities like the Air Force have already begun to streamline procedures and systematically implement UAS operations over airfields, the process for obtaining clearance and coordinating flight operations is a significant hurdle when operating UAS in the context of airfield assessment. Regulations regarding drone hardware, flight paths, and operator qualifications, instituted by both the Federal Aviation Agency (FAA) and the local flight operations authority must be understood and followed (FAA, 2021b). These restrictions remain a large barrier to data collection and proposed methods of operation; however, as these operations become more common, the process by which approval is gained to perform them will likely be made more efficient. The drones used in this study are classified as MAV Hexacopters and were custom built with autopilot capabilities to carry multiple kinds of imaging systems while remaining compliant with current National Defense Authorization Act (NDAA) restrictions for use over Air Force Runways.

Chapter 3

Machine Learning and Convolutional Neural Networks

3.1 Machine Learning

Machine learning methods are a class of computer algorithms that utilize data to make predictions or decisions. They have recently emerged as one of the most powerful computational tools available for leveraging data from the real world to accomplish a task efficiently; however, it is crucial to understand that no matter the algorithm employed, the human engineer always plays a vital role in the problem solving process. Without a human to frame the problem, acquire and organize the data, design a space of possible solutions, select a learning algorithm and its parameters, apply the algorithm to the data, validate the resulting solutions, and apply it in a usable framework, the algorithm will not successfully perform its intended task (Drori, 2020).

Machine learning relies on the assumption that inductive reasoning is reliable, meaning

that data from the past is an accurate means of determining future behaviors. From this common assumption algorithms can be divided into several distinct categories based on the type of data provided.

The first is supervised learning, in which data inputs are fed into the model with corresponding outputs. In general, the goal of these models is to learn information about how these inputs relate to their outputs for the purpose of either classification or regression. Classification is a task in which input data has a discrete number of possible output categories, or "classes". Then for each input the algorithm tries to determine what the correct output classification should be. For example, if a classification algorithm were to be trained on images of dogs and cats, the input data would be a collection of pictures containing either dogs or cats, and the algorithm would need to determine if the picture contained a dog or cat. Regression tasks are similar to classification, in that each input has an output, however, in regression the number of possible outputs is continuous rather than discrete. For example, if the inputs for an algorithm were a person's age, level of education, and state of residence and the desired output was the person's yearly income, this would be a regression style task, since the possible outputs are a continuous and not discrete. The task would become a classification problem if the desired output were to classify the person as either "wealthy" or "poor" based on this same information.

Unsupervised learning provides inputs, but does not provide any outputs. In general, the goal of these models is to learn patterns or structures inherent in the input data and this can be broken into three general tasks: (1) density estimation, (2) clustering, (3) dimensionality reduction. Density estimation involves taking a set of inputs as a sample and determining the probability that a new input is drawn from that same distribution. Clustering tries to

create groups of data that are similar to each other. Dimensionality reduction seeks to take inputs of dimensionality D and retain the information and relationships of the original data while re-representing them in a new dimensional space d such that $d > D$.

Reinformcement learning is similar to supervised learning, in that output values are being mapped to input data, except in reinforcement learning the input and output relationships are not given in advance to conduct model training. Rather there is an interacting environment in which the algorithm receives an input, produces an output or action, and then receives a reward, penalty, and/or correction to update the model's behavior. In this style of task the goal is to find a policy that maximizes rewards and improves the model's ability to interact in a given environment. A common application of this style of algorithm is teaching computers to win complex games of strategy like "Chess" or "Go". Additional information about other machine learning algorithms can be found in (Russell and Norvig, 2010).

3.2 Neural Networks

Neural networks (NN) are a broad class of supervised machine learning algorithms for which there are many variations and applications. To gain insight into how these algorithms perform their tasks this section will briefly survey some of the fundamental concepts required to understanding their behavior. This summary is not exhaustive and for a deeper and more complete understanding for the material, it is suggested that a full course be taken and/or a textbook be used (Russell and Norvig, 2010) to supplement this introduction.

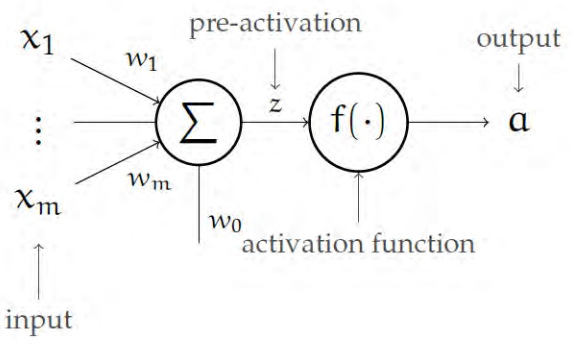


Figure 3-1: A neuron is made of an input vector, x, a weight vector, w, and a bias term, w_0. The dot product of the input vector and weight vector is then added to the the bias term to obtain the pre-activation term. The pre-activation is passed through the activation function to obtain an output, a, for the node (Drori, 2020)

.

3.2.1 The Basic Element: Neurons

The basic element of a neural network is the "neuron", also referred to as a "node" or "unit" (Figure 3.3). A neuron is comprised of a vector of inputs (x), a vector of "weights" (w), and a "bias term" (w_0) that are related through Equation 3.1 to calculate the "pre-activation" (z). Then the pre-activation term is passed through an activation function, like the Rectified Linear Unit shown in Eq. 3.2, to obtain output (a). There are many activation functions used in neural networks like ReLU, the Step Function, the Sigmoid Function, or the Hyperbolic Tangent. Each has different applications and uses and if more information is sought there are many online articles that provide a good starting place. By adjusting the weight matrix and the bias terms of each neuron, the output of each neuron is able to be adjusted. A complete mathematical description of a single neuron is found in Eq 3.3

$$z = w^T \cdot x + w_0 \tag{3.1}$$

$$ReLU(z) = max(0, z) \tag{3.2}$$

$$a = f(z) = f(w^T \cdot x + w_0) \tag{3.3}$$

3.2.2 Layers

Neural network layers are comprised of a set of neurons; usually these neurons are fully connected, meaning every input (x) in the input vector is connected to every neuron in the layer (Figure 3-2). This creates a vector of outputs for a single layer. A single layer of neural network can only represent a linear relationship between input and output data; however, by using the outputs of the first layer as the inputs to subsequent layers of a neurons, non-linear features and relationships in the data can be understood by the model by changing the weights and bias term of each neuron. If a neural network has many layers it is said to be a deep neural network.

While it may be tempting to try to understand or make sense of what kinds of relationships or patterns a neural network is learning at each neuron or through each layer, it is my opinion that attempting to understand this kind of behavior and assign a sort of rational to the network's structure is misguided for the average user. Other machine learning algorithms seek to apply domain specific knowledge to encode meaningful features into a model; however, in a neural network manual feature selection is abandoned and instead the model tries to make connections and understand the data in a way that is best described through the mathematical processes. Employing thousands or millions of neurons in many

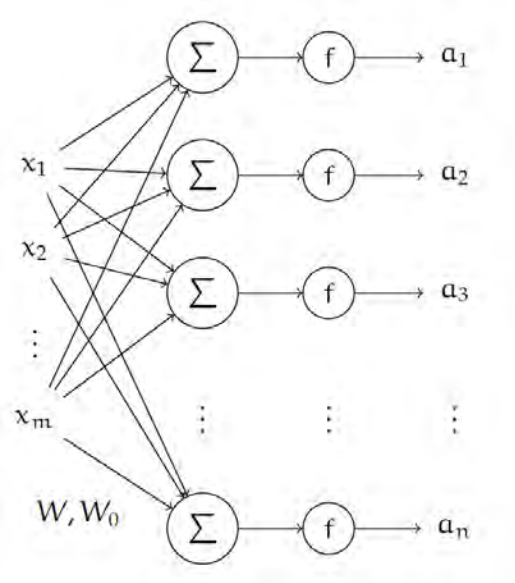


Figure 3-2: The figure displays a fully connected layer in a Neural Network. Every input (x) in the input vector is connected to each neuron in the layer and each neuron has its own vector of weights (w) and a bias term (w_0) to generate an output (a). A vector of outputs is generated for each layer and passed to the next layer as the new set of inputs (Drori, 2020)
.

different layers gives neural networks complexity to analyze or process data in a way that is not necessarily transferable to human logic and reasoning strategies.

As a supervised learning model a neural network is given input data with corresponding output data and asked to perform a task like classification or regression. For the sake of discussion let's pretend we have a neural network used to classify a patient as either healthy or pre-diabetic by using the patient's height, weight, blood pressure, and hours of exercise per week. The input data for a single patient will be fed into the first layer and the network will process these values through the neurons, understanding different features or relationships within the data set in each of the neurons, until the last layer. In this last layer there may be a single numeric output representing the probability the patient is pre-diabetic with a threshold value for this output determining the final classification. The performance of this neural network will depend on the value of the weights and bias terms in each of

the neurons, and to optimize these values to obtain the best performing model, a process known as error back propagation is used.

3.2.3 Error Back Propagation

During the "training" process the goal is for the network to adjust the weights and biases of the model such that the model's predictions match the correct outputs as best as possible. Weights and biases start at initialized values and these values drive convergence and performance. If a model is trained without ever seeing prior data, these weight terms can be initialized through a variety of methods that encourage convergence, like Xavier or Kaiming initialization (Narkhede et al., 2021). An alternative method for initializing weights for a new data set is using "transfer learning", which uses the weights and bias terms of a similar neural network trained on a different data set to initialize the weight values of a new model that will be trained on a new data set. This allows for some of the features learned in the original data to be used to understand the information in the new data, making training faster and convergence more likely. This method of "transfer learning" has been shown to be highly successful and is used in this study (Balada et al., 2021).

The means by which these weights and bias terms are adjusted is through error back propagation. In this process the neural network is initialized with starting values for each of the weights and bias terms. Then data are passed through the network to obtain the initial predictions of the model with the initialized weights and biases. The network's output is represented by $NN(x^{(i)}; w, w_0)$ while the true output associated with the input is represented by $y^{(i)}$. Then, using the network's prediction and the true output, a loss function (L)

calculates a quantitative metric for the difference between these two values. This process is summarized by Eq. 3.4. While different loss functions exist to service different networks and applications, with some function expressing a quantifiable difference between the predicted and true output value , adjusting the weights and bias terms in each layer of the model becomes an optimization problem that seeks to minimize the loss function using gradient decent methods.

$$J(w, w_0) = \sum_i L(NN(x^{(i)}; w, w_0), y^{(i)}) \tag{3.4}$$

3.2.4 Convolutional Neural Networks

While many classes of deep neural networks exist to perform tasks on different data types, since AlexNet in 2012 (Krizhevsky et al., 2012), the most successful method for performing classification tasks on image data has been the CNN. By definition, "convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers" (Bengio et al., 2017). By using a series of these convolutions, as well as pooling and fully connected layers, CNNs construct a hierarchy of nonlinear transformations and take advantage of the spatial locality and translational invariance of pixels to classify images with fewer parameters and connections than a standard deep neural network (Krizhevsky et al., 2012). This essentially means that CNNs are able to look at a picture and identify simple features and shapes, like lines and edges, and then from these simple shapes identify increasingly more complex shapes and features until eventually real world objects and structures can be identified consistently, regardless of ori-

entation or permutation. Traditional classification CNNs could simply tell whether or not a type of object was present in an image. For example, in the sample unit pictured in Figure 2-10, traditional CNNs could determine that there are cracks and patches present in that image, however, they could not identify where in each image these objects were located. More recently developed CNNs; however, can accomplish semantic segmentation, which means they are capable of determining which specific pixels in the image represent cracks, patches, or background objects, resulting in full image outputs like the picture shown in the bottom of Figure 2-10.

Convolutions are mathematical operations on two functions of a real-valued argument (Bengio et al., 2017). Though the operation can easily be expanded to higher dimensional tensors, for the case of two-dimensions, input I interacts with the two-dimensional kernel filter K to produce an output S. In Figure 3-3 this operation is represented by taking the dot product of all the $3x3$ grids in the X matrix with the W matrix to produce each of the elements of the Z matrix.

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n) \tag{3.5}$$

The numerical values that make up the individual elements of the kernel filter (the W matrix in Figure 3-3) are called the weights and are adjusted just like the weights in a standard neural network through error-back propagation. A stride value defines how this kernel filter and convolutional operation moves over the image, thereby producing a new feature map tensor, where each element corresponds to the convolution's output at that point. Then similar to a neuron, a bias term is added to each kernel filter, and these

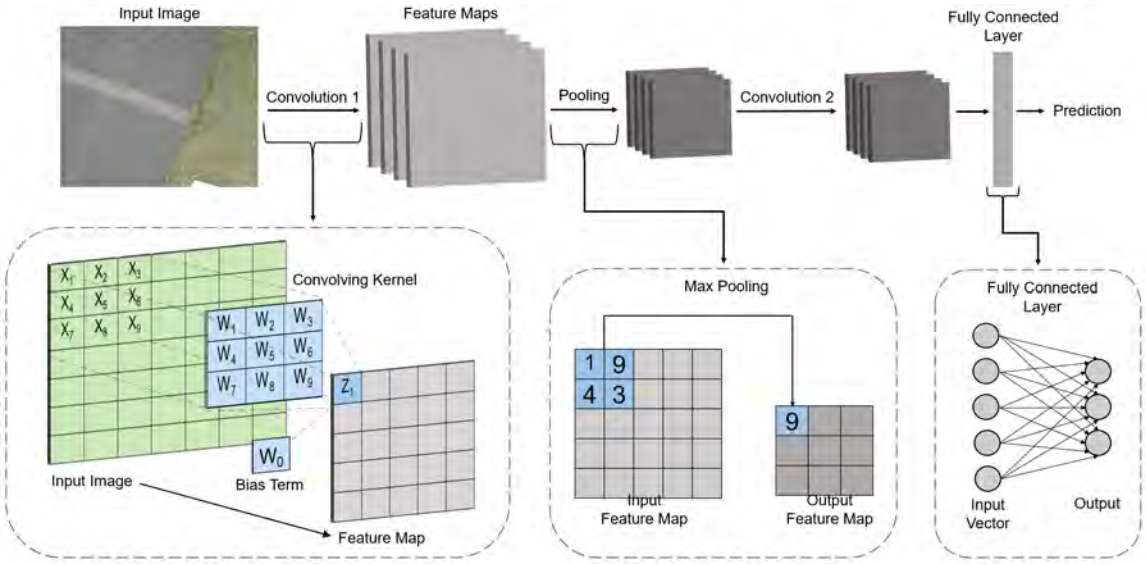


Figure 3-3: Visual representation of how convolutional and max pooling layers construct and condense feature maps.

outputs are passed through an activation function, which introduces non-linearity, allowing more complicated features to be understood by the network (Nwankpa et al., 2018; Gu et al., 2015). Inserted between layers of convolution are pooling layers that serve as the network's means of summarizing features and generalizing the model to accommodate input data variability. Max Pooling is a common pooling operation that functions similarly to a convolutional layer, but rather than applying a kernel filter of weights over the image to produce an output, the output of each operation is simply reported as the maximum value found in the window of size MxN. This process (summarized by Figure 3-3) enables for complex feature information to be stored and aggregated within the network.

After a series of convolutional and max pooling layers are the fully connected layers that map the features to object class predictions using a series of unique weights, bias terms, and activations. The weights and bias terms that exist in the the network's convolutional layers and fully connected layers constitute the total number of parameters for a model.

Unique to semantic segmentation tasks, however, is the addition of a decoder structure that uses stored feature information to reconstruct the image with labels assigned to each pixel.

Chapter 4

Methodology and Results

The desired output of this research is creating an autonomous drone mounted pavement evaluation tool that can independently collect images and output an Airfield PCI. To divide this large objective into discrete sub-objectives, the project team decided to split the system into the tasks of (1) Drone flight and image capture, (2) distress detection within images, and (3) distress dimension calculation and PCI calculation (Fig. 4-1). This chapter outlines the preliminary methods evaluated to complete each of these tasks in a real world test case scenario.

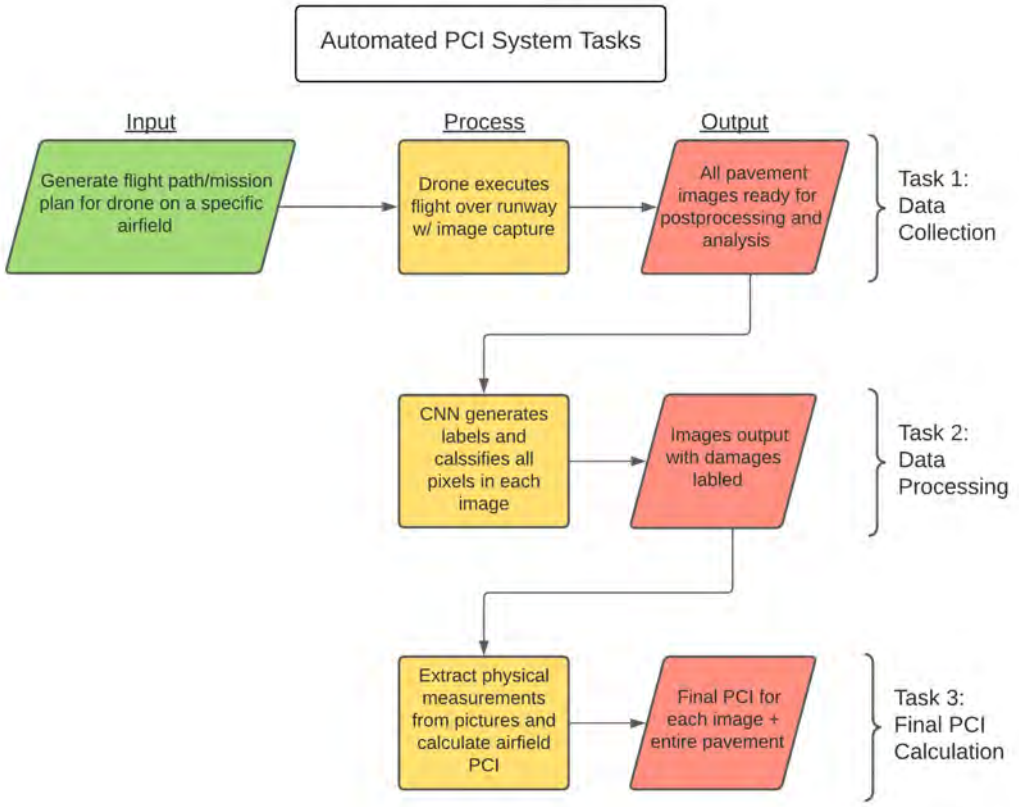


Figure 4-1: Conceptual flow chart summarizing the three general tasks to be performed by an autonomous drone mounted PCI inspection system.

4.1 Site Description

The site selected to develop, implement, and evaluate the proposed methodology was the Aardvark Airfield located at the United States Air Force Academy (USAFA) in Colorado Springs, CO (39°02'06.0"N, 104°50'42.0"W). A Google Earth image of the site is provided in Figure 4-2. Constructed in the mid-1980's, the Aardvark Airfield consists of a single 2300ftx75ft asphalt pavement runway with no other sections or facilities (Freeman, 2017). It currently has three primary functions: (1) perform practice landing patterns, (2) serve as an emergency landing area for the cadet glider pilot programs run out of USAFA's main airfield, and (3) launch and land larger drones built by USAFA's UAS research teams. Clear

Figure 4-2: Google Earth image of Aardvark Airfield (boxed) along northeast boundary of US Air Force Academy (Google, 2021).

signs of small-scale maintenance and full-scale repair are present, though the exact work history is unknown. The most prevalent pavement distresses throughout the runway are longitudinal/transverse cracks and patching, though a few instances of rutting and raveling were also observed.

As a result of the special use case (and a USAFA academic research affiliation), it was fairly easy to receive federal clearance to fly the drone equipment over the airfield both consistently and with flexible flight parameters, like speed and altitude. It is important to note, however, that the United States Air Force Academy does not regularly use Aardvark airfield for powered flights, therefore the loading history of the pavement may vary from a standard airfield pavement. Consequently the distress profile of the entire structure is likely to be more heavily representative of distresses caused by weathering and Colorado's environmental factors than from service loads. We do not suspect that this discrepancy

will significantly affect the overall conclusions drawn by this research, however, it will necessitate further data collection from more commonly used airfields to improve distress detection performance.

4.2 Hardware Specifications

The type of remote sensing application summarized in this report requires UAVs capable of carrying sensors for data collection as well as flying at various speeds and elevations above ground level (AGL). Additionally, to be implemented on a large-scale throughout the Air Force, this project requires a user-friendly UAV flight interface capable of being flown by Air Force Civil Engineers in both auto and manual flying modes. NDAA legal guidelines prohibit the use of certain technological components on drones flown over government property (Congress, 2020). Commercially available drone packages may possess the hardware and software capability to complete this type of work, but this project was limited to those drones permissible under stringent department of defense regulations. Due to these restrictions, a custom built, NDAA compliant MAV Hexcopter (Figure 4-3) was utilized for this study. A summary of relevant USAFA MAV Hexcoper properties is presented in Table 4.1.

Another critical hardware component was the imager chosen for data collection. For a remote sensing application, one of the most important considerations for sensors is sensor weight. Also considered were spectral range and resolution. While data within the visible spectrum are important and allow a user to manually cross-check automated results, data outside of the visible spectrum may expand the range of variations when using a machine

Figure 4-3: The drone used to image the airfield in this study is shown above, having a propeller-to-propeller length (also called ”wheelbase”) of 1.8ft.

Table 4.1: The custom drone system used for all flight and image capture tasks had the following technical specifications.

Technical Specifications - USAFA MAV Hexcopter	
Description	**Specification**
Frame	DJI Flame Wheel F550
Rotors	DJI 2212/920KV (x6)
Weight	3.5lbs
Wheelbase	1.8ft
Maximum Payload	4.2lbs
Battery	HRB 4S 14.8V 6000mAh Li-Po
Endurance	25 minutes
Transmitter	mRo SiK Telemetry Radio V2 915Mhz
Flight Controller	Pixhawk
Ground Control System	Mission Planner v1.3.74

learning approach to data analysis on other distress types in future studies. The imager selected for this study was the FLIR Duo Pro R. Relevant properties of this sensor are presented as Table 4.2.

While many commercial systems are equipped with connectors for mounting an imager to a UAV, the custom nature of the USAFA MAV Hexcopter required special harnesses to be built for the imager and accompanying batteries. This was accomplished using 3D printing and the SolidWorks CAD software. Due to this system's construction; however, an unexpected resonance from the drone's operational components initially distorted the images. This problem was resolved by inserting layers of polystyrene foam into the camera mount to dampen vibrations.

The highly visible nature of the two distresses considered in this study (patches and cracks), meant that an image sensor alone was likely sufficient for detection purposes. Other distresses, like rutting, shoving, and swell, may be less visible and harder to distinguish with an image sensor alone. Future research developments may involve integrating additional sensing types, such as Lidar for 3D mapping or hyperspectral imaging.

Table 4.2: Specifications of the camera used for data acquisition.

Technical Specification - Remote Sensing Imager	
Description	**Specification**
Model	FLIR Duo Pro R
Weight	13.2 oz
Thermal Resolution	640 x 512
Thermal Frame Rate	30Hz
Visible Resolution	4000 x 3000

4.3 Data Collection

The data collection portion of this study was largely focused on optimizing the flight plan to minimize flight time while maintaining a spatial resolution capable of detecting cracks 0.25 inches in width. Spatial resolution was a function of imager properties, so initial flight plans were created with field of view and imager resolution in mind. However, initial flights showed that flight speed had a substantial effect on image quality due to both frame rate and imager stability, so several test flights were taken to optimize for overall image quality as well as resolution. For the hardware used in this study, this led to a requirement that the UAV fly no higher than 98.4 feet AGL at a horizontal speed of approximately 4.5 miles per hour. Inputting these flight parameters into Mission Planner software, the entirety of Aardvark Airfield was successfully imaged via a single pass over the center of the runway.

Table 4.3: Flight conditions used to calculate PCI values.

Flight Parameters for Data Collection	
Flight Parameter	**Value**
Altitude (AGL)	98.4ft
Speed	4.5mph
Resolvable Ground Object Size	0.25in

The values in Table 4.3 represent tuning for the Aardvark Airfield location and for a specific collection of custom hardware components lacking commercial grade robustness and mounting components. Additional optimization would likely be required for varying location and hardware, for example, better image resolution would enable higher flight altitudes, while better image stabilization mounts could enable faster flight speeds. Furthermore, it should be noted that there were some consistent issues with windy conditions

grounding flight operations, so daily and seasonal weather patterns should also be a consideration when creating a flight plan tailored to a specific drone's limitations.

4.4 CNN Model Architecture

Created in 2018, DeepLabV3+ is currently one of the best performing CNN architectures available for "semantic segmentation", a process by which a classification label is assigned to every pixel in an image. Utilizing Atrous Spatial Pyramid Pooling (ASPP), this architecture is able to use several different expansion rates and effective fields of view to extract image features and encode multiple scales of contextual information when first analyzing the input image. Then during the decoding phase, the encoder features are upsampled and linked with corresponding low-level features from the network backbone in the encoder. Gradual reinsertion of spatial information then achieves pixel-level predictions with more distinct object boundaries (Chen et al., 2018; Liu et al., 2021b). This structure is summarized in Fig 4-4.

A technical and mathematical explanation is required to accurately communicate these complex processes (Chen et al., 2018); however, a highly simplified conceptual understanding may think of the classification process as one that also results in a loss of spatial information. So if the network wanted to identify a human face, it must do so by first grouping together individual pixels to extracting lines and curves, which then need to be grouped together to see shapes like ovals and rectangles, which then need to be grouped together to understand structures like eyes and lips, which then need to be grouped together to identify a face. As these features are grouped together for the purpose of classification,

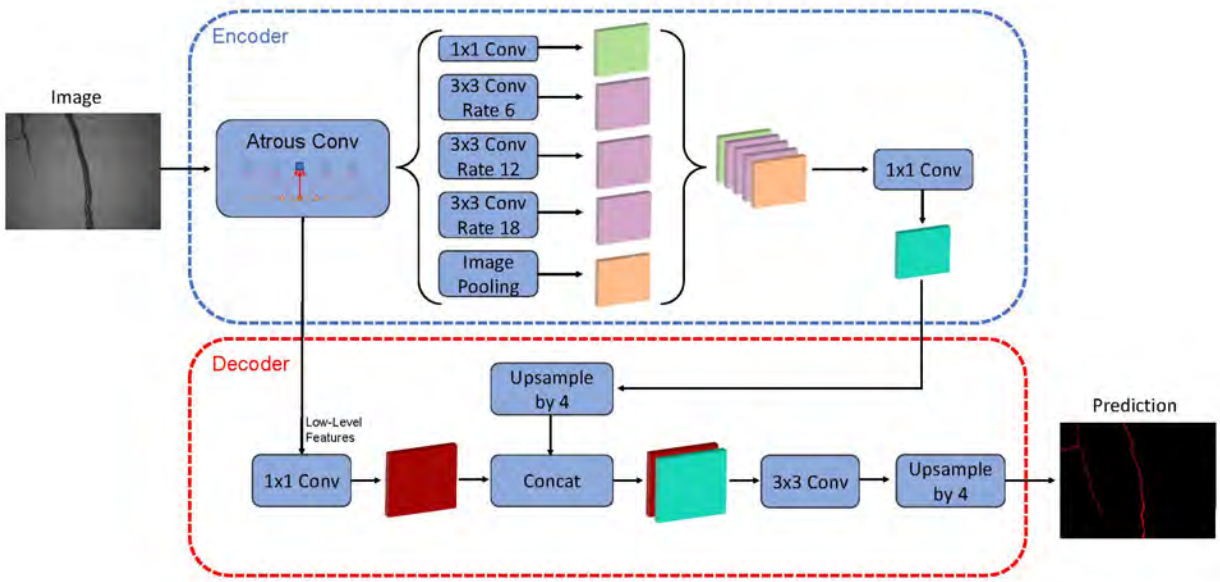


Figure 4-4: DeeplabV3+ encoder-decoder structure. Atrous convolution helps capture information at multiple scales during the encoding process, while the simple decoder module builds more accurate object boundaries for segmentation results (Chen et al., 2018).

spatial resolution is diminished and the matrix of numbers that began as the original image is reduced down. Therefore, it is only at a very low resolution that the computer recognizes a face is present. So by taking different snapshots of the original image and connecting low level classifications within the model to earlier features of a higher resolution, these models reconstruct the high resolution image from the low resolution classification.

While the overarching model architecture was already established and selected based on performance in the literature (Liu et al., 2021b; Chen et al., 2018; Liu et al., 2021a), various choices for the feature extractor, or "backbone", used by the neural network were investigated for this application. DeeplabV3+ is built to accept a variety of feature extracting algorithms from which it builds its encoder-decoder structure. Each feature extractor has a different number of parameters, and therefore a different computational demand. To systematically asses the application specific trade-off between performance and computa-

tional demand, four different network backbones spanning a range of sizes were selected, trained, and tested. Identical training data, test data, hyperparameters, and GPU hardware were used to asses each feature extractor model, the details of which are discussed in Section 4.5. The models evaluated were ResNet-18 (He et al., 2016), ResNet-50 (He et al., 2016), Xception (Chollet, 2017), and Inception-ResNet-V2 (Szegedy et al., 2017).

4.5 Model Data and Training

The CNN trained in this application was created to identify cracks and asphalt patches, as these were the two dominant and adequately represented distresses present on the test runway. To use on other asphalt pavement airfields, other distresses would need to be added in future developments as the data for both training and testing the algorithm become available. Matlab 2021b™ with the Deep Learning Toolkit™ was selected as the means of implementation for this code, though similar model architectures are available for implementation in other coding languages, like Python with the Keras and TensorFlow libraries.

Model training was conducted on a single 48GB NVIDIA™ RTX A6000 GPU with a batch size of 5. Data augmentation was used to introduce variability into the relatively small training data set, with specific values and transformations summarized in Table 4.4. Twenty epochs of model training were conducted to obtain adequate model accuracy convergence, while most other hyperparameters, summarized in Table 4.4, were initially chosen by using recommendations from the Matlab™ documentation (Mathworks, 2021a).

As is the case for all supervised machine learning algorithms, the performance and

Table 4.4: The CNNs trained and tested on the airfield pavement image data were run with the same values.

CNN Hyperparameter and Data Augmentation Values	
Hyperparameter	**Value**
Batch Size	5
Number of Epochs	20
Input Image Resolution	1920 x 1080
Solver	"Adam"
Initial Learning Rate	0.00001
Gradient Decay Factor	0.9
Squared Gradient Decay Factor	0.999
Data Shuffle	"Every-Epoch"
Data Augmentation	**Transform Range**
Random X Translation	-100 to 100 pixels
Random Y Trnaslation	-100 to 100 pixels
Random Rotation	-25 to 25 degrees

utility of the network in real world application is heavily dependent on the quality and properties of the training data and labels used. If the training data used to teach the network do not represent features of the real world data, even a model that achieves perfect accuracy in training will not generalize and perform well on the task when tested. To the best of my knowledge, no open source airfield pavement distress data sets exist for training semantic segmentation CNNs; therefore a novel flexible airfield distress data-set was created for the purpose of this study. The training data set consisted of 500 total RGB images cropped or split to be 1080x1920 in size. Each image was individually labeled by hand. Early iterations of this label making process indicated that model detection performance increased the more time was spent during the image labeling process, because the labels themselves could be generated carefully and accurately (this is a conclusion also found in the literature (Zlateski et al., 2018)).

Of these 500 total images, 60 were taken from online or open sources to introduce variation, and 440 were taken from drone flights over the Aardvark Airfield at a variety of altitudes under varying atmospheric conditions. 80% of these data were used for model training and 20% for validation. In an effort to specifically assess model performance in the context of a full PCI inspection, the test set of images used to evaluate model performance was generated by performing a single test flight over the entirety of the Aardvark Airfield, the result of which was 128 test images that represent the entirety of the runway. Although none of the images used in training were used in testing, because a large percentage of the training data images are taken from the Aardvark Airfield, this imposes limitations on model robustness and attempting to apply this system to other airfields. This is a limitation created by the availability of current airfield image data. We suspect that as more runways

are surveyed and the training set is able to incorporate images from more locations, the model will generalize better and address these limitation.

To evaluate the models, we report the predictions for global accuracy and mean Intersection over Union (IoU). Global accuracy is used as a simple measurement that is easy to communicate, as it just takes the total number of correctly labeled pixels in the image and divides it by the total number of pixels in the image.

$$GlobalAccuracy = TruePositive/(TotalPixels) \tag{4.1}$$

IoU (or the Jaccard Similarity Coefficient) is one of the most commonly used metrics, providing a measurement of statistical accuracy that also penalizes the model's false positive predictions. Mean IoU is calculated by averaging the IoU scores found for each of the object classes in an image.

$$IoU = TruePos./(TruePos. + FalsePos. + FalseNeg.) \tag{4.2}$$

By conducting model training with identical computer hardware, training/test data, and system parameters (Table 4.4), the training time, mean IoU, and global accuracy values were determined for each of the different models. These results are summarized in Table 4.6, showing the Inception-ResNet-v2 model to be the best performing option on this particular data set. While ResNet-50 showed performance results close to that of Inception-ResNet-v2, while requiring only half as much time to train, the availability of powerful computational resources (Table 4.5) meant that Inception-ResNet-v2 was ultimately the

model chosen to complete the distress detection task. Should larger data sets or changes in computational resources arise, however, it is worth noting that the ResNet-50 model exists as a computationally less demanding alternative.

Table 4.5: The computer used to train and test the CNNs in this study was purchased through Lambda Labs (https://lambdalabs.com/) and was equipped with powerful computational resources intended for running large deep learning models.

Hardware Specifications	
Operating System	Ubuntu 20.04 (Includes Lambda Stack for managing TensorFlow, Pytorch, CUDA, cuDNN)
Processor	AMD Threadripper 3960X: 24 cores, 3.80 GHz, 128MB cache, PCIe 4.0
CPU Cooler	Air Cooling
GPU	2x EDU, RTX A6000, 48 GB+ NVLink
Memory (RAM)	256GB
Operating System Drive	1TB SSD (NVMe)
Extra Storage	2TB SSD (SATA)
Case	Vector Case

Table 4.6: Performance summary of all tested models

Base Network	**Parameters (Millions)**	**Training Time**	**Global Accuracy**	**Mean IoU**
ResNet-18	11.7	47min	97.94%	62.24%
ResNet-50	25.6	75min	98.63%	69.77%
Xception	22.9	95min	97.86%	61.92%
Inception-ResNet-v2	55.9	155min	98.99%	71.9%

4.6 PCI Calculation

In this study the Aardvark Airfield was treated as a single runway section divided into 32 individual sample units (Figure 4-5) in accordance with ASTM-D5340. Each of these sample units was inspected first manually, using the existing standard, and then also using the drone images and computer based processing.

Figure 4-5: The Aardvark runway is divided into 32 sample units for the purpose of PCI inspection.

A former Air Force certified PCI inspector and APE Team member assisted in conducting a PCI inspection for each of the 32 sample units, recording distress measurements and severity for the entire runway. These values were input into a spreadsheet that calculated the final PCI scores for each sample unit, as well as the overall runway. These results are summarized in Table 4.7.

To conduct the automated PCI inspection, a flight plan was created and uploaded into the drone's autopilot. The drone then launched, flew the mission, collected video footage of the entire runway, and landed without human input. Still images of each sample unit were cropped from this video footage and processed with the trained CNN to extract crack and patch distresses from the image. An example of the CNN's visual output result is displayed in Figure 4-6 and includes original images with distress marking overlays, individual extractions of each of the distress categories, and single-pixel traces of the cracks.

To calculate the total square feet of patching and the total linear feet of longitudinal/-transverse cracking, the physical dimensions of each pixel were determined using the imager's field of view (FoV), resolution, and flight altitude. Using a consistent flight altitude of 98.4ft, a field of view of 56°x 45°, and a resolution of 4000 x 3000 pixels, each pixel

was calculated to be spatially representative of approximately .0262ft x .0271ft (Equation (4.3)).

$$\begin{aligned} PixelLength &= ((FoV_x/2) \cdot Altitude \cdot 2)/Pixels_x \\ PixelWidth &= ((FoV_y/2) \cdot Altitude \cdot 2)/Pixels_y \end{aligned} \tag{4.3}$$

To calculate the total area of patches in a sample unit the total number of patch labeled pixels was multiplied by the area of each pixel. To calculate the total length of longitudinal and transverse cracks, a single-pixel trace of the crack's path is created using the medial axis transform of the original crack detected (Mathworks, 2021b). This trace is represented by the white pixels in Figure 4-6. The total number of pixels from this trace was then summed and multiplied by the average of the length and width dimensions to get total length. Using an average of the length and width dimensions roughly accounts for the cracks path, moving either longitudinally, transversely, or diagonally across the image.

Distress severity for utility patches was assumed always to be of "medium" severity, as there is not yet a method for distinguishing severity of this nature. Distress severity for longitudinal and transverse cracking was determined by taking the the ratio of total crack labeled pixels (red plus white pixels in Figure 4-6) to total crack labeled pixels in the trace images (just the white pixels). In this way the width or thickness of each crack is represented by the red pixels, while its length is estimated by the white pixels. This ratio serves as a rough indication of crack width, which is a key trait determining severity. "Low" severity cracks had a ratio of 2:1, "medium" severity cracks had a ratio between 2:1 and 7:1, and "high" severity cracks had a ratio above 7:1. Though the manual method of inspection differentiates between severity levels for measurements taken of the same distress type within the same sample unit, the computer based analysis had to assume that

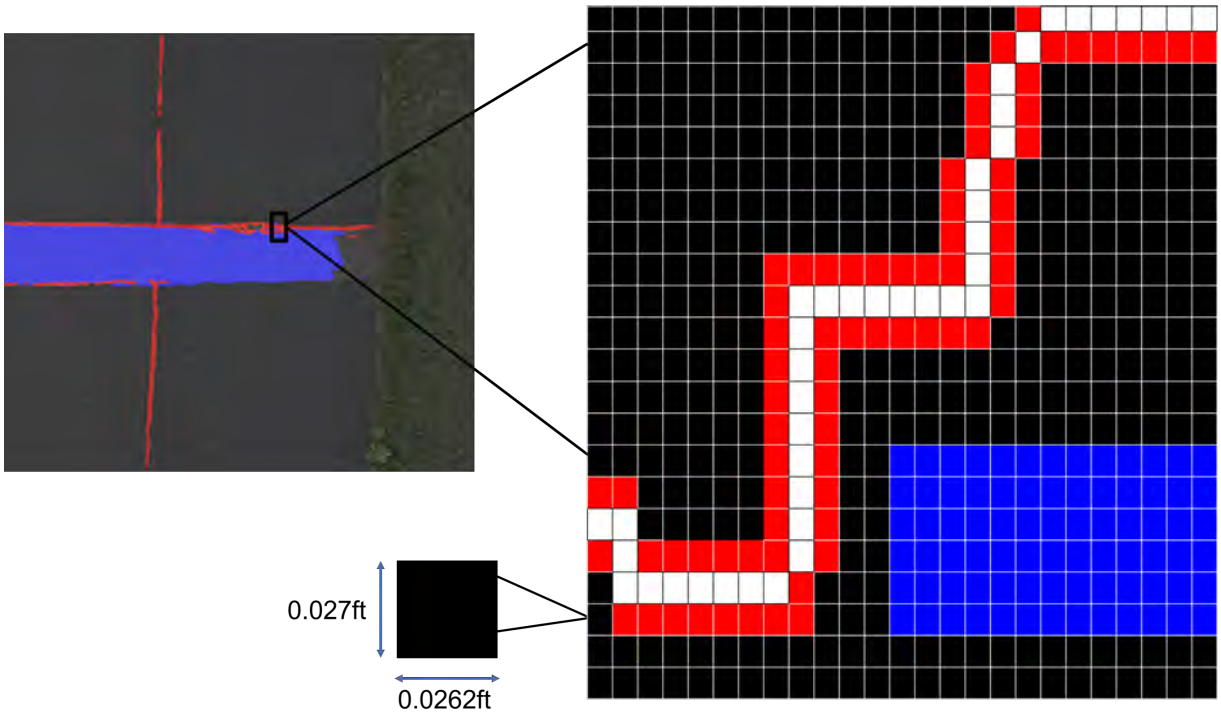


Figure 4-6: At the pixel level, red represents a crack, white is the single-pixel trace through the crack used to determine length, blue represents a patch, and black is the background.

for both cracks and patches a single severity level defines the measurement value obtained for the entirety of the sample unit.

The patch areas and crack lengths, along with their corresponding severity, were then input into the PCI spreadsheet to calculate final PCI scores for each sample unit, as well as the entire runway. The results are displayed alongside the manual inspection results in Table 4.7. Both inspection methods were in close agreement for their determination of the overall PCI, with the manual method yielding a final score of 54 and the autonomous method yielding a score of 56.5.

While these final values are promising, comparing distress measurements and PCI scores for each individual sample unit gives further insight into the algorithm's performance and ways to improve the automated system. The error in area measurements for

utility patching in each sample unit is presented as a histogram in Figure 4-7. Because there was a large range for measurements values, that included zero, the typical percent error or percent difference error metrics did not reliably describe system performance. Therefore, the error metric used in the X-axis was simply a difference, obtained by subtracting the automated system measurement from the manually determined measurement. The same reasoning and difference metric was applied to length measurements of longitudinal/transverse cracking presented in Figure 4-8. Because the range of PCI scores was much more narrow and did not include zero, percent error was used as the metric to compare the PCIs and is displayed in Figure 4-9.

The histograms in Figures 4-7, 4-8, and 4-9 reveal that for most sample units, there is relatively low discrepancy in measurements; however, some outliers and inaccuracies seem to cause both overestimation and underestimation, resulting in a roughly normal distribution of error. Looking more closely at the sample units with the greatest error, it became apparent that the likely cause of this error was from the CNN distress detection stage. On occasion the CNN would find a non-crack or non-patch structure and yield a false positive detection, or the CNN would miss a crack or patch structure and yield false negative detections. From this reasoning we conclude that improving detection results should greatly improve performance and reduce error. (Supplementary material regarding the algorithm's performance on the test images is located in Appendix-A)

Table 4.7: This table includes the utility patching, longitudinal/transverse cracking, and PCI values for each sample unit as determined by the conventional (manual) and drone-based (automated) inspection system.

	Patch Area (sqft)		**Crack Length (ft)**		**PCI**	
Section	**Manual**	**Automated**	**Manual**	**Automated**	**Manual**	**Automated**
1	0	8.5	320	224.9	54	59
2	278	174.2	310	355.1	52	49
3	200	223.8	375	314.1	45	68
4	200	193.8	357	390.6	47	46
5	80	88.6	200	275.9	59	52
6	160	139.1	370	353.5	44	49
7	100	151.1	315	294.8	52	52
8	150	157.1	280	274.8	52	55
9	160	133.5	250	209.8	54	59
10	150	199.8	210	229.8	59	55
11	50	53.8	252	251.4	55	55
12	100	27.0	265	223.3	55	59
13	150	185.2	300	300.1	52	52
14	50	49.7	260	236.6	51	55
15	150	184.4	380	341.8	44	49
16	50	54.5	241	185.7	57	59
17	150	172.8	310	332.1	49	49
18	0	0.0	215	227.6	59	60
19	150	163.3	260	297.4	50	52
20	100	96.7	285	315.1	50	52
21	50	52.4	275	318.8	52	52
22	0	0.0	180	182.8	64	64
23	85	70.0	180	260.2	57	55
24	0	96.6	115	114.8	68	68
25	225	229.5	300	270.6	52	55
26	150	189.4	260	250.9	50	70
27	0	0.0	195	200.0	59	78
28	300	274.1	300	305.0	52	52
29	0	5.2	150	136.4	62	62
30	150	152.4	310	316.0	47	52
31	0	15.6	215	284.0	59	52
32	0	0.1	100	173.1	76	67
				Final PCI:	54	56.5

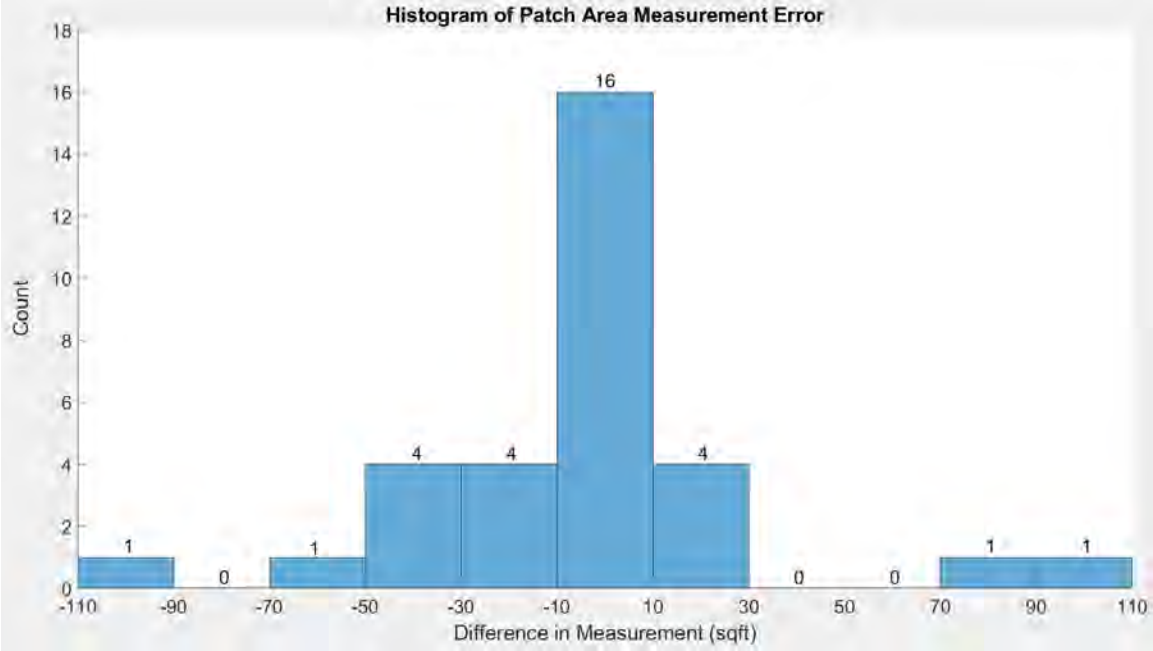


Figure 4-7: Histogram of the error for utility patch area measurements found by the manual vs. automated inspection method for all surveyed sample units. Error is reported as the difference between the manual minus automated measurement.

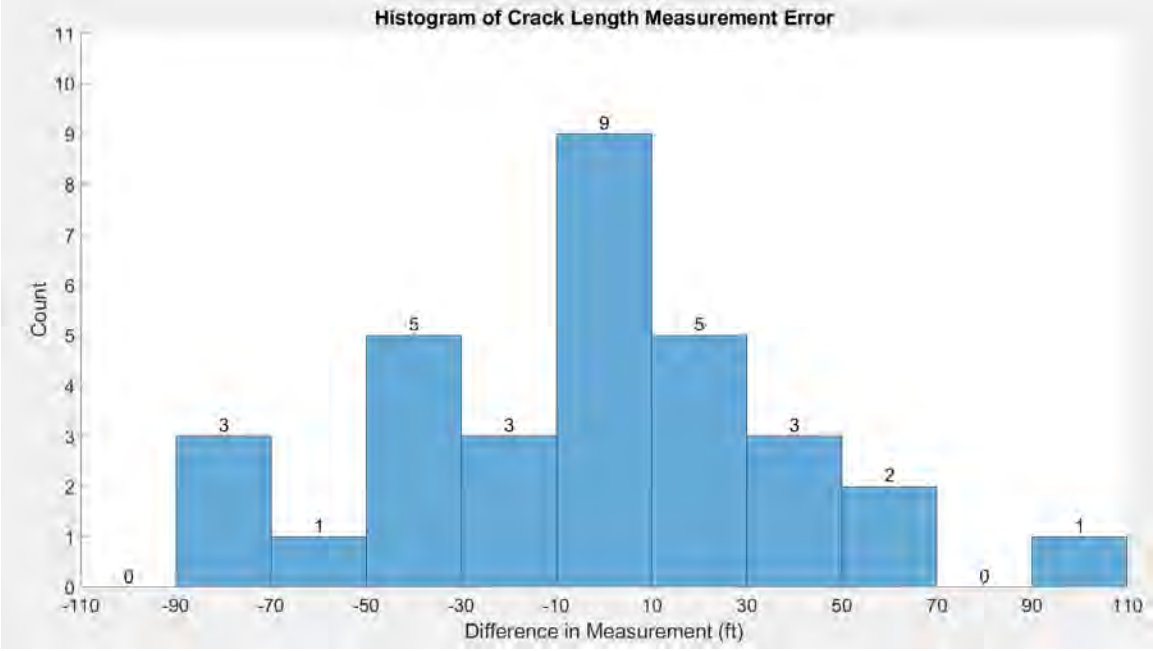


Figure 4-8: Histogram of the error for crack length measurements found by the manual vs. automated inspection method for all surveyed sample units. Error is reported as the difference between the manual minus automated measurement.

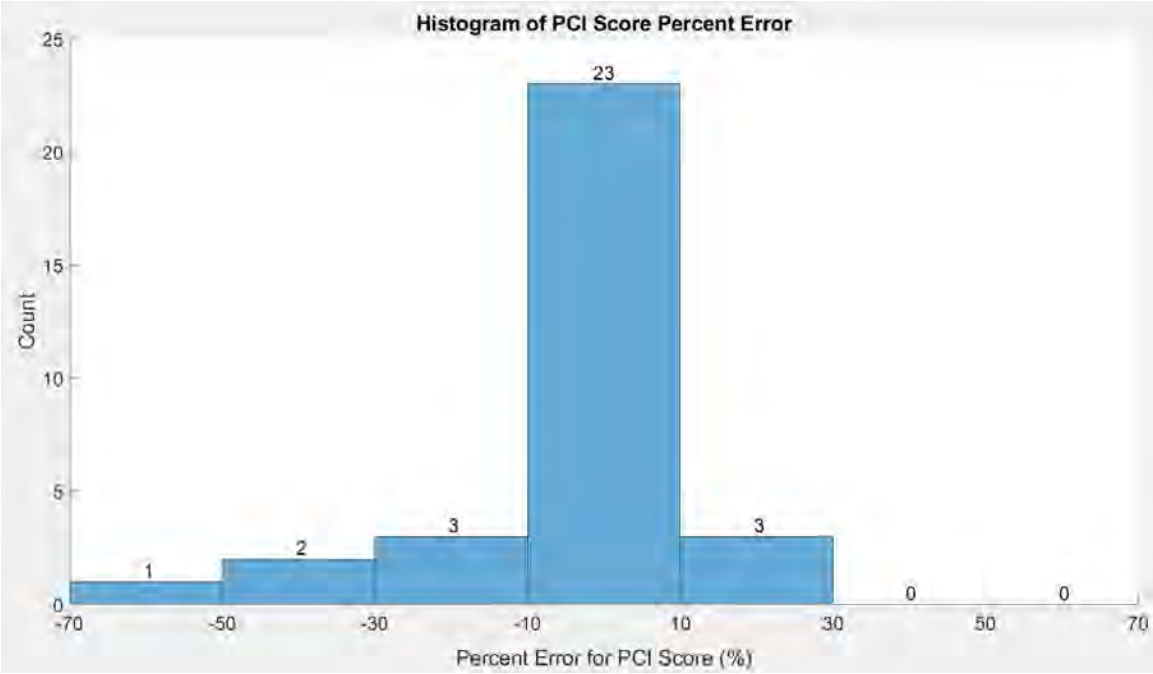


Figure 4-9: Histogram of percent error for the PCI Score found by the manual vs automated inspection method for all surveyed sample units.

Chapter 5

Flexible Strategy for Integrating Autonomous Assessment Technology into US Air Force Airfield Pavement Evaluation Procedures

5.1 Executive Summary

The work presented in this chapter was written as part of Professor Richard de Neufville's Engineering Systems Analysis for Design course, which focused on learning to consider uncertainty when conducting system analysis and design. As opposed to traditional methods that rely on deterministic forecasts of the unknown, by incorporating a system's uncertainties into the design process students were taught how to create and evaluate flexible

systems and strategies that could potentially improve project performance measures. As a Civil Engineering student and a Civil Engineering officer in the United States Air Force (USAF), I wanted to apply this flexible design methodology to the rollout procedure of a new drone-based, autonomous airfield pavement evaluation system that I am currently developing to replace (or augment) the existing manual process. Replacing this manual method with an autonomous inspection method (requiring little to no human input) has the potential to significantly reduce inspection costs across the Air Force, however, as a new technology still in development, there is uncertainty regarding its use in the field. While simplifying assumptions had to be made about some aspects of the new inspection system's performance, the primary variables and uncertainties affecting overall cost of inspection were built into a model that reports the net present value (NPV) of all airfield pavement inspection costs across 220 Air Force installations for 30 years assuming one inspection conducted per year. Using this cost model, the goal was to determine a flexible strategy for replacing inspection procedures with variants of a new drone-based system at each of the 220 bases under consideration.

A deterministic "base case" is first generated that assumes perfect knowledge of the future with no uncertainty, and several integration plans were developed to service this scenario. Then uncertainty is introduced, which increases projected costs of each of these plans. A sensitivity analysis of different system variables determined that parameters related to the cost of drone equipment each year had the most significant impact on the total NPV. With this knowledge several flexible strategies for system integration are then analyzed and presented to best respond to this uncertainty. While the search of possible strategies was not exhaustive, these results generally indicate that transitioning to a new

automated system will not only reduce costs, but will do so while adding new inspection capabilities that do not currently exist with the manual system. A small initial roll-out, with gradual expansion across all the bases was found to be a preferred strategy, and performance benchmarks on reliability and drone failure should be monitored to guide that expansion process for better results.

5.2 Introduction and Motivation

Funded by the US Air Force Civil Engineering Center, there is ongoing research to develop a novel method for conducting an automated airfield pavement condition index (PCI) survey on Air Force owned airfield pavement assets using drone mounted imaging technology. Results from field testing over an auxiliary airfield located at the Air Force Academy in Colorado Springs, CO have shown promising results and have warranted further development of this system. While developing the technological components of this system is currently underway, another significant aspect of this project is how a real-world implementation may occur. The institutional transition from the existing manual method of inspection to an automated, machine-based approach, not only carries with it significant capital investment to meet new equipment requirements, but also introduces a host of logistic and technological uncertainties that could influence the overall cost of the transition. By providing a simplified model of this integration period and using a simulation-based uncertainty analysis, this chapter provides some general insights into factors that might aid the implementation process and the direction of technological developments. A flexible implementation strategy is suggested for integrating this novel and untested system into

real-world Air Force operations across an immensely diverse range of installations. Investigating and developing alternative implementation options that consider uncontrollable uncertainties has the potential to reduce average cost and risk. While the quality of data used to conduct this analysis limits the accuracy of any quantitative results relating to cost, the qualitative conclusions presented still present valuable and useful contributions to the ongoing development of this project.

5.3 System Model

5.3.1 Structure, Organization, and Assumptions

The model used to conduct the analysis in this chapter was constructed and run in MATLAB Version 9.10.1684407 (R2021a). Its final output value is the total cost attributed to the inspection of a specified number of airfields for a given number of years with a set number of inspections conducted each year. The model calculates the total cost of all airfield inspections by looking at each airfield under consideration as a unique entity, calculating the cost of inspection at an individual location, then summing these costs over all locations, for each inspection cycle. All inspection cycles are calculated over a specified time window and a discount rate is used to speak of all costs in terms of net present value (NPV).

As a default position, each airfield is assumed to have been operating a manual pavement inspection system prior to the time interval evaluated in the model. This means all equipment costs for the manual method are assumed to be zero, as each base location should already have the needed equipment at their disposal. Additionally, the replacement

cost of any equipment associated with this method are considered negligible over the time interval being evaluated. Therefore, the only cost that contributes to the completely manual method is the inspection time. Provided to the model is a "rate of inspection" term that represents the average number of square feet that can be inspected and analyzed by a reasonably experienced inspector in one hour (units: ft^2/hr). The total square feet of pavement being inspected at a location is then divided by this "rate of inspection" term to obtain the total number of hours required to inspect the entire runway. Multiplying the total number of hours required, by the cost of labor per hour, yields a reasonable estimate for the total cost of surveying the airfield using the manual method. Because the PCI standard does not necessitate, nor do current manual methods usually practice, the inspection of the entire runway, the model utilizes a sample-based method for determining the area required for manual inspection at each base location. This area is calculated using the minimum "sample unit" requirement specified by the ASTM. This amounts to roughly .8% of each runway needing to be surveyed under the manual method. Additional inspection areas are randomly added to this minimum value, however, to account for additional inspections required for maintenance or repair purposes beyond the scope of a PCI. In contrast, the automated system is treated as always inspecting the entire runway, providing the new capability of classifying and locating all distresses present on the runway.

When a base is selected to switch to a drone-based system the model assumes, for administrative purposes, that an individual base must implement the semi-autonomous system for three inspection cycles to prove safety and functionality, before switching to the fully autonomous version. A one-time initial cost to purchase the drone system is incurred by any base switching from the manual system to a drone system; however, this cost is not

incurred again when switching from the semi-autonomous to the fully autonomous implementations, as all equipment used is assumed to be the same. If multiple systems are bought in the same inspection cycle, an economies of scale factor is applied to capture the savings from purchasing multiple systems at once. Like the manual system, the semi-autonomous system also has its own "rate of inspection" that governs the cost of human labor associated with each inspection. This encapsulates the cost of sending a qualified drone pilot on location to conduct image collection over an entire runway. Unlike the manual and semi-autonomous methods, the human labor cost is assumed to be zero for the autonomous system, as human involvement with the inspection process should be negligible. Drone-based systems have a factor associated with any maintenance costs required to service the system. Considering this maintenance, a single drone is assumed to service a single base for the entire 30-year evaluation period. To account for the likelihood of a drone crash or catastrophic failure event, probability terms are assigned and updated for each base during each inspection cycle that ultimately indicate whether a replacement drone system will need to be purchased for the base. To provide a more conservative estimation of equipment requirements, this model assumes bases are not permitted to share drone systems and must purchase their own system individually.

Because this drone technology is still in its infancy, the model also considers a rough estimation for institutional and technological "learning" or improvement, which is incorporated into the model as a scaling factor that reduces certain costs. Learning factors increase the rate of inspection (decreasing labor costs) and decrease the probability of catastrophic failure (decreasing equipment costs). Learning factors grow logarithmically with each inspection cycle to symbolize diminishing marginal improvement over time. For example,

one may imagine that in the first year of implementing the drone system at a specific base, the flight path is very long and inefficient. But after adjustments and learning, the flight path is updated and made 50% more efficient in year two, greatly reducing the cost. But then maybe in year three, the flight path can only be made 10% more efficient than the year prior.

Because both airfield specific and Air Force wide learning will take place the longer the system is used, learning factors are divided into "local learning" and "global learning" factors. Local learning factors are unique to a specific base. Examples of this may be more efficient flight paths or a special operating procedure given local weather conditions. Once a base implements a type of drone system, its local learning rate factors begin to reduce costs with each inspection cycle, however, this learning remains local, and the benefits do not extend to any other bases. Simultaneously, though to a lesser degree, global learning also takes place with every inspection cycle at every airfield a drone-based system is in use. An example of this may be new software or hardware developments that improve drone stability and reduce the risk of crashing. Any cost reductions attributed to global learning apply to every airfield operating a drone-based platform, regardless of prior use history. Because the manual method of inspection has been implemented in the field for many years, it is assumed to have reached steady state cost and is not diminished by any "learning" factors. This process is visually summarized by Figure 5-1.

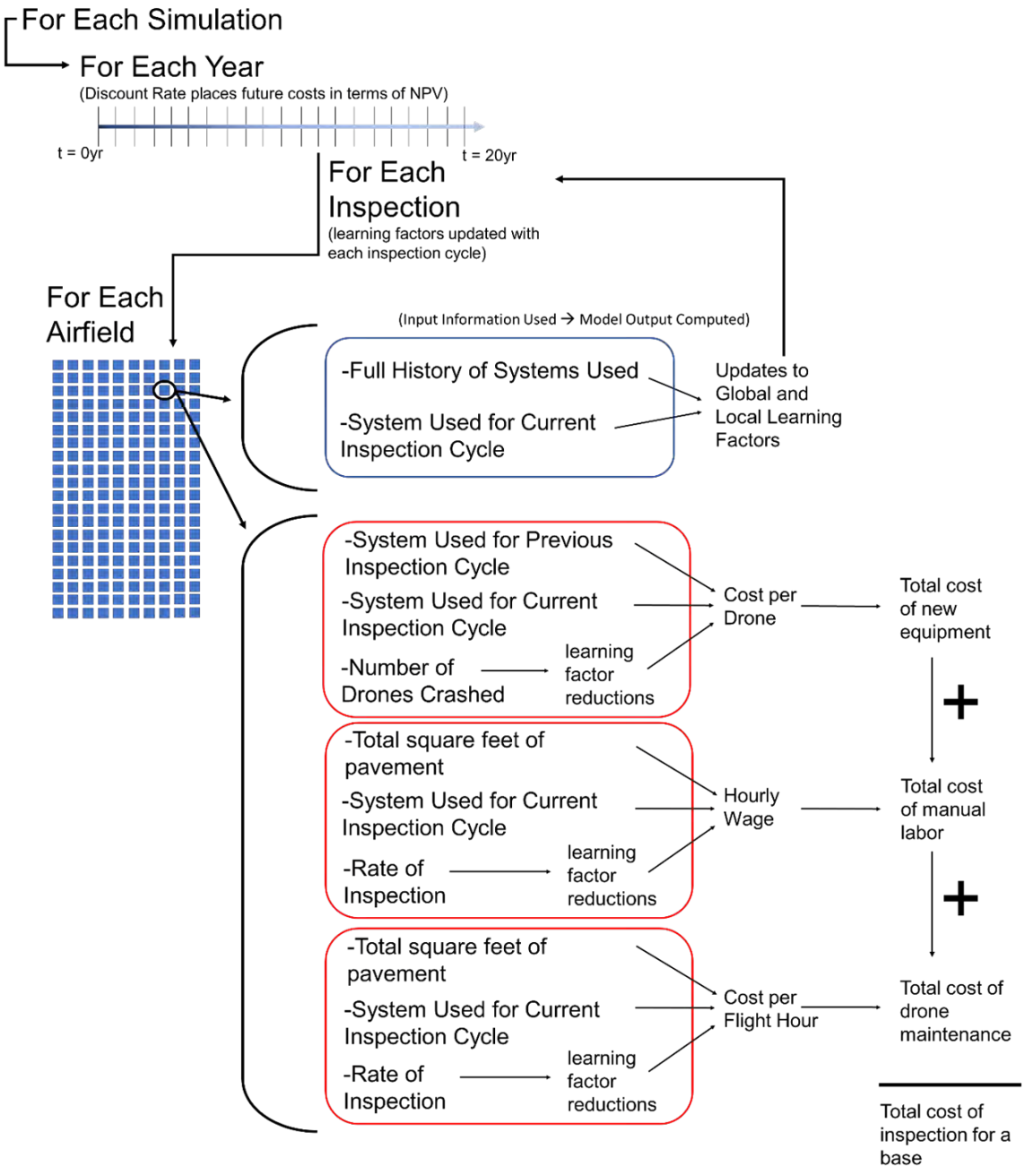


Figure 5-1: A qualitative visual summary of how the model is structured to calculate the cost incurred by conducting a single inspection cycle at a single runway.

5.3.2 Data and Parameter Values

Applying this method of uncertainty analysis (Section 5.4.1) to the nature of this problem is aimed at helping develop a preliminary qualitative strategy for implementing a drone-based inspection system across a network of Air Force installations. With this goal in mind, using highly accurate data to generate simulations of this model is not essential to the analysis. The following section summarizes the values and probability distributions used by the model to conduct simulation.

The Air Force reports supporting 2.2 billion square feet of pavement evaluations across roughly 220 military bases worldwide (Ford, 2020). Using these numbers as reference, 220 total installations are used in the model. In the interest of simplicity, the pavement area per airfield is uniformly assigned to be 10,000,000ft^2, which is just the average area calculated by taking the total ft^2 of pavement inspected and dividing that by the total number of installations. The discount rate used was 2.4% and was selected was based on the 30-year federal guidelines for government projects (Vought, 2019). An economies of scale factor of .90 was selected to reflect a nominal impact, as the number of drones being purchased at any one time will never be exceptionally large given that each base is only assumed to operate a single system. One inspection per year is assumed, with a time interval of 30 years evaluated in each simulation. The hourly wage used to calculate human labor costs for both drone piloting and manual inspection is taken randomly from a right skewed distribution with an average value of $22/hr. $22/hr is used as the average because it is roughly the hourly rate of an E-4 (Senior Airman), in the Air Force, though individuals of another rank or pay scale also conduct PCI inspections or drone piloting duties. To

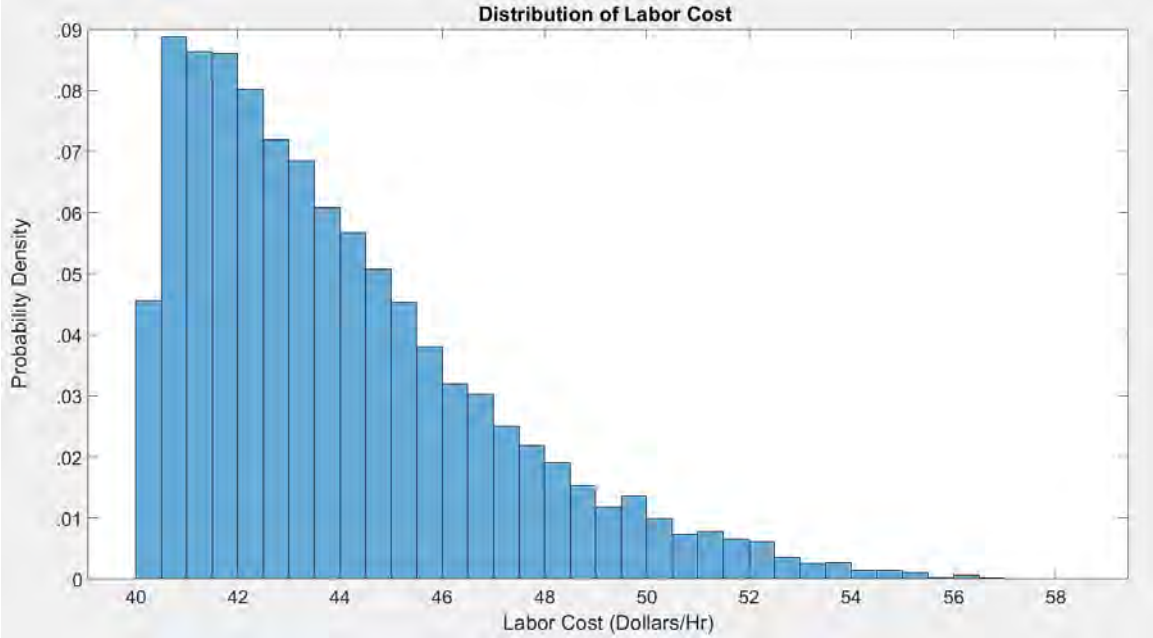


Figure 5-2: The cost of labor used in each simulation is obtained by taking a random value from this right skewed distribution.

conservatively estimate the institutional cost of labor used in the model we then double the wage to account for overhead, resulting in a right skewed distribution with an average rate of $44/hr (Figure 5-2). The maintenance cost of $5 per flight hour was estimated from average life expectancy values for drone parts like motors, propellers, batteries, and other electronic components. The cost of a drone system is averaged to $10,000, based on rough quotes of an existing Air Force platform being tested.

Due to the scarcity of data, values and distributions for “semi-autonomous drone crash probability”, “autonomous drone crash probability”, “semi-autonomous drone rate of inspection”, “autonomous drone rate of inspection” and “manual rate of inspection” were estimated using experimental data from inspections conducted over a single airfield in Colorado. Rates of inspection are sampled from a normal distribution and drone crash probabilities were selected to reflect conservative estimations based on field observations and

anecdotal information provided by experienced drone pilots. Local and global learning rates for both crash probability and rate of inspection were simply selected as conservative estimations of improvement. Due to the speed of implementation and highly specified nature of certain practices related to inspection and safe flight, local learning is considered to occur at an order of magnitude higher than global learning. Because learning is assumed to have diminishing returns with time, as the most obvious and effective improvements will likely be found and implemented first, both learning rates demonstrate logarithmic improvement with time. The numeric values and distributions for all parameters are summarized in Table 5.1.

Table 5.1: All parameters and their type of uncertainty distribution are summarized in this table. For more details regarding how they are used in calculations please request the code for this analysis from the author.

Parameter	Initial Value	Distribution Type
Number of Airfields	220	Singular Value
Years	30	Singular Value
Inspections/Year	1	Singular Value
Full Coverage Airfield Area	10M (ft^2/Airfield)	Singular Value
Sampled Airfield Area	80,000 (ft^2/Airfield)	Normal, but cannot be below 80,000ft^2
Discount Rate	2.4%	Singular Value
Economies of Scale (Alpha Factor)	0.90	Singular Value
Drone Cost	$10,000	Normal
Hourly Cost of Labor (Manual)	$44/hr	Skewed Right
Hourly Cost of Labor (Drone Pilot)	$44/hr	Skewed Right
Maintenance Cost (Drone)	$5/hr	Normal, but cannot be below 0
Rate of Inspection (Manual)	5,000ft^2/hr	Normal, but cannot be below 2,500 ft^2/hr
Rate of Inspection (Semi-autonomous Drone)	200,000ft^2/hr	Normal
Rate of Inspection (Autonomous Drone)	300,000ft^2/hr	Normal
Probability of Crash (Semi-autonomous Drone)	0.1	-Normal, but cannot be below 0
Probability of Crash (Autonomous Drone)	0.25	-Normal, but cannot be below 0
Learning Factor (Drone Crash-Global)	0.01	-Normal, but cannot be below 0
Learning Factor (Drone Crash-Local)	0.001	-Normal, but cannot be below 0
Learning Factor (Inspection Time-Global)	0.005	-Normal, but cannot be below 0
Learning Factor (Inspection Time-Local)	0.001	-Normal, but cannot be below 0

5.4 Base Case

5.4.1 Deterministic vs Uncertainty Case

Before evaluating implementation strategies with uncertainty considerations, we first conduct a preliminary analysis on a deterministic base case model, which reflects no uncertainty or distributions of possible values. Consequently, all parameters are set to their initial value summarized in Table 5.1.

Using this deterministic case, three basic strategies for drone system integration were evaluated after a baseline case was run to represent continuing current practices without automation. The first plan proposes the most rapid and expedited shift, calling for the purchase and implementation of a semi-autonomous drone system at every airfield immediately at t=0. Then at year three all systems switch to a fully autonomous implementation. The second plan takes a moderately phased approach, switching 55 airfields to a semi-autonomous system initially at $t = 0$. Every five years thereafter, another 55 airfields adopt a semi-autonomous system. After three years of semi-autonomous implementation, bases will switch to a fully autonomous system. The third plan takes the most delayed and phased approach, adding 11 airfield drone systems a year. After three years of semi-autonomous implementation, bases switches to a fully autonomous system. Running the model for each of these scenarios results in a single value output that is the total cost in terms of NPV.

Then, to simulate uncertainty and data variability each parameter, instead of being a single value, is replaced by a distribution of values from which the model randomly selects for each iteration of Monte Carlo simulation. A thousand simulations are run in this study to create a distribution of possible outcomes, rather than a single value. Target curves

Table 5.2: This table summarizes the strategies implemented in the base case evaluation, as well as average total cost for the three implementation strategies as compared to the manual only method.

Plan	Implementation Strategy	Total Costs (NPV) - Deterministic	Total Costs (NPV) - W/ Uncertainty
Manual Only	-No drone system implemented, only manual inspections.	$3,364,200	$12,262,000
1	-Switch all 220 installations to a semi-autonomous system at t=0. -Switch all 220 installations to fully autonomous systems after 3 years of prior drone use.	$12,511,000	$13,911,000
2	-Switch 55 installations to a semi-autonomous system at t=0. -Switch an additional 55 installations to a semi-autonomous system every 5 years after. -Installations switch to a fully autonomous systems after 3 years of prior drone use.	$10,369,000	$13,772,000
3	-Switch 11 installations to a semi-autonomous system at t=0. -Switch an additional 11 installations to a semi-autonomous system every year after. - Installations switch to a fully autonomous systems after 3 years of prior drone use.	$8,508,200	$13,137,000

are produced to summarize the results of these simulations. These curves represent the cumulative distribution function of all simulation outputs, with the x-axis representing total system cost in terms of NPV and the y-axis representing the likelihood a value from the output distribution falls below the corresponding cost related by the curve.

For each plan, the deterministic value and the uncertainty distribution's average value is recorded in Table 5.4, and the target curves produced by the uncertainty analysis are shown in Figure 5-3.

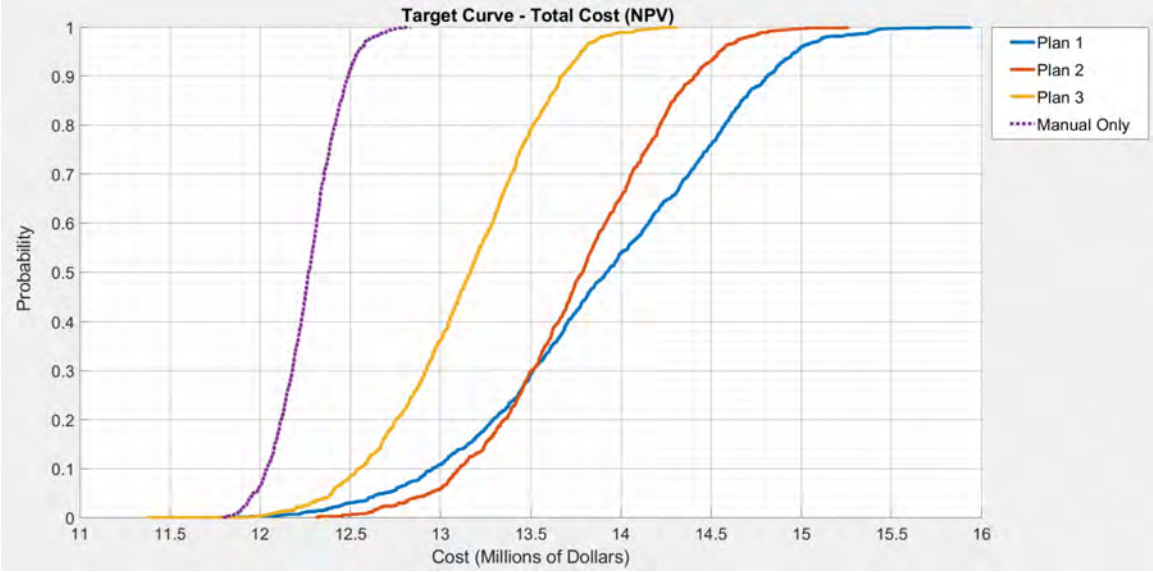


Figure 5-3: These cumulative distribution functions correspond to the distributions of total costs produced by a thousand interactions of Monte Carlo simulation. Each of the curves correspond to a different implementation strategy. Because cost in term of NPV is represented on the horizontal axis, lower values and distributions shifted left are desirable.

One key observation from the target curves shown in Figure 5-3 is that the rate of implementation seems to be a factor improving general performance, as the most delayed and steadily phased implementation protocol yield a dominant left shifted curve when compared to the other drone-based strategies.

5.4.2 Sensitivity Analysis

To attempt to improve upon this initial analysis of costs, new flexible strategies that respond to uncertainty are developed and considered. The first step taken toward constructing a flexible strategy is gaining a better understanding of how the system variables individually influence the behavior of the overall model. Because implementation plan number three had a lower average cost, as well as a dominant target curve, plan three's was used to generate the results of the sensitivity analysis. Holding all other variables constant, individ-

Table 5.3: Input ranges used in the deterministic model of Plan 3 to generate the tornado diagram found in Figure 5-4

Parameter	Base Case Value	Lower Bound	Upper Bound
Probability of Drone Crash	Semi-Autonomous: 0.9 Fully Autonomous: 0.75	Semi-Autonomous: 0.6 Fully Autonomous: 0.6	Semi-Autonomous: 0.99 Fully Autonomous: 0.99
Cost of Drones	$10,000	$5,000	$20,000
Learning: (Drone Crash)	(For Both Drone Systems) Local: 0.01 Global: 0.001	(For Both Drone Systems) Local: 0.1 Global: 0.01	(For Both Drone Systems) Local: 0 Global: 0
Manual Survey Sample Area	80,000 ft^2	80,000 ft^2	500,000 ft^2
Rate of Inspection	Manual: 20,000 ft^2/hr Semi-Autonomous: 200,000 ft^2/hr Fully Autonomous: 300,000 ft^2/hr	Manual: 40,000 ft^2/hr Semi-Autonomous: 400,000 ft^2/hr Fully Autonomous: 600,000 ft^2/hr	Manual: 2,500 ft^2/hr Semi-Autonomous: 25,000 ft^2/hr Fully Autonomous: 100,000 ft^2/hr
Cost of Drone Maintenance	$5/hr	$2/hr	$15/hr
Cost of Labor	$44/hr	$25/hr	$88/hr
Learning: (Rate of Inspection)	(For Both Drone Systems) Local: 0.005 Global: 0.001	(For Both Drone Systems) Local: 0.05 Global: 0.01	(For Both Drone Systems) Local: 0 Global: 0

ual parameters were changed to a reasonable upper, and then lower, bound. This process helped determine the quantitative behavior of the model in response to each change. A summary of the variables examined, as well as the corresponding ranges along which they were evaluated, is presented in Table 5.3.

The findings of the sensitivity analysis are summarized by the tornado diagram displayed in Figure 5-4. From these results we conclude that factors directly or indirectly related to the cost of drone acquisition are the most influential variables on overall project cost. This is something that can be practically accomplished by trying to leverage economies of scale, the time value of money, or restrictions on acquisition price (assuming price variability). Equally, if not more impactful to this model, are costs incurred by crashing drones that break and must be replaced. Because this model assumes every drone crash requires replacement, this result indicates that there is substantial value to be gained by rugged tech-

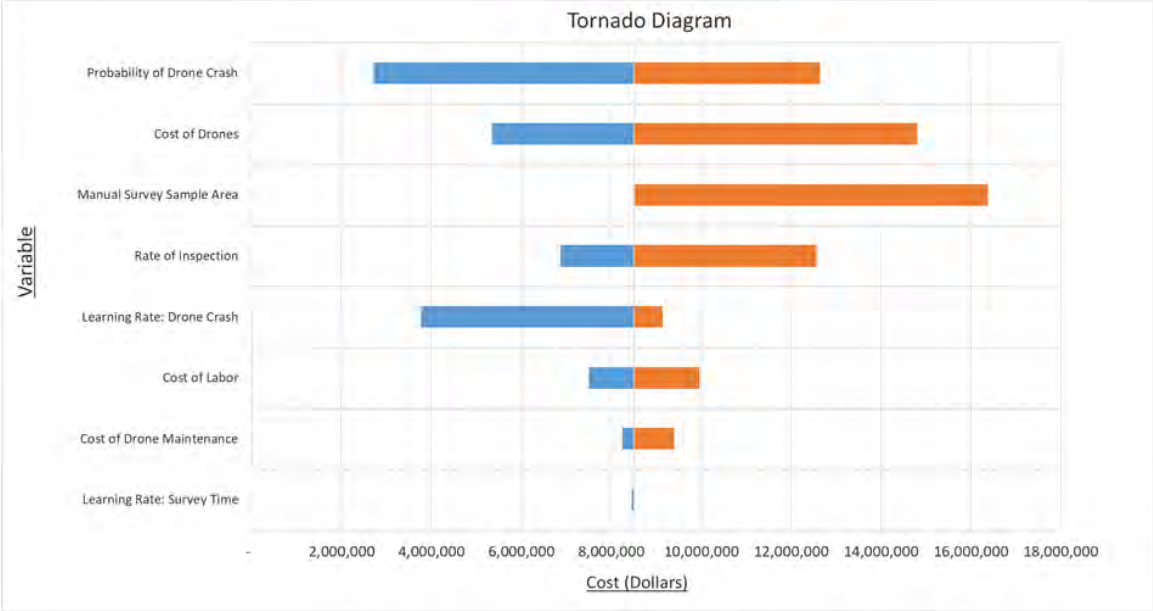


Figure 5-4: By evaluating the upper and lower bounds of parameter values, a range of possible outputs is generated corresponding to each variable. This range suggests the relative influence of different variables on the total cost. Variables with the greatest influence are located at the top of the diagram, with least influential variables at the bottom.

nological drone designs that either reduce crashing or do not require drone replacement in the event of a crash. Paying a slight premium for this technological feature, given the uncertainty and variability of real-world operating conditions, is worth considering.

5.5 Incorporating Flexibility

5.5.1 Flexible Strategies Investigated

The data used in this analysis are highly uncertain due to the novelty of the systems investigated, but general trends and qualitative information remain relevant. The sensitivity analysis provides more detailed insight into the relationship between total cost and parameter values, showing that variables influencing the cost of drone equipment, either from

crashing or new acquisitions, heavily impacted total costs as well.

With these system characteristics in mind, three alternative flexible strategies are proposed and compared to the previous plans. All plans acquire drones in 55 unit increments to try to benefit from economies of scale. The first flexible option then also capitalizes on the reduced crash probability supported by the learning rate, starting with just 55 drone systems at $t = 0$, with no set time schedule placed on future additions. Rather, 55 new systems are implemented only when the total crash number of crashes in the previous inspection period drops below 15% percent. The second option places an emphasis on rapid acquisition, while still considering price fluctuations, ordering 55 drone systems each year the price of drones falls below $9,000, until all airfields acquire a drone-system. While this random price fluctuation model may not mimic the contract-based procurement strategy often used by the military, this case is meant to represent strategies that are willing to make a marginal sacrifice in expediency in exchange for favorable pricing. The third option is a combination of both flexible options 1 and 2, and acquires an additional 55 drones any year the number of drone crashes is below 15% and the current drone price is less than $9,000. To evaluate these strategies, the average NPV, standard deviation, range, and shape of the cumulative distribution functions are used to qualitatively compare the resulting output distributions for total cost, as well as total cost of drone equipment only.

5.5.2 Model Results

With uncertainty considered, a thousand simulations of each plan are conducted by the model, generating distributions for the total cost of all inspections, as well as the total

Table 5.4: These are the flexible option strategies evaluated in this analysis compared to the existing method. Also reported are the averages, standard deviations, and ranges of the distributions for total cost and drone equipment cost.

Plan	Implementation Strategy	Total Costs (NPV): 1. Average 2. (std) 3. [range]	Drone Costs (NPV): 1. Average 2. (std) 3. [range]
Manual Inspection	-No drone system implemented, only manual inspections.	1. $12,262,000 2. ($165,000) 3. N/A	N/A
Flexible Option 1	-Initialize 55 installations with semi-autonomous system at t=0. -Add 55 drone systems in the following inspection cycle if total drone crash rate drops below 15% until all airfields have been switched. -Fully autonomous system is applied after 3 years of drone use.	1. $10,354,00 2. ($2,554,000) 3. [$4,490,000 - $14,466,000]	1. $320,820 2. ($189,750) 3. [$0 - $719,220]
Flexible Option 2	-Initialize 55 installations with semi-autonomous system at t=0. -Add 55 drone systems per inspection cycle so long as the cost of drones is <$9,000 until all airfields have been switched. -Fully autonomous system is applied after 3 years of drone use.	1. $9,125,300 2. ($1,903,700) 3. [$4,364,000 - $12,613,000]	1. $261,540 2. ($153,960) 3. [$0 - $706,810]
Flexible Option 3	-Initialize 55 installations with semi-autonomous system at t=0. -Add 55 drone systems per inspection cycle so long as the cost of drones is <$9,000 and the total drone crash rate from the previous cycle is below 15%. Continue until all airfields have been switched. -Fully autonomous system is applied after 3 years of drone use.	1. $11,110,000 2. ($1,967,000) 3. [$5,197,000 - $14,801,000]	1. $300,820 2. ($190,660) 3. [$0 - $1,026,900]

cost of drone equipment, over the 30-year evaluation period. The averages and standard deviations of these distributions are summarized in Table 5.4, while the target curves of the total cost and cost of drone equipment are presented in Figures 5-5 and 5-6, respectively.

Looking at both the total cost as well as the cost of drones we observe all three flexible strategies having lower average costs than the previously evaluated plans, as well as the existing manual method. Additionally, although each flexible strategy has a larger range of outcomes and more variability, their target curves are still shifted left compared to the other options, making them a preferred method of implementation.

Looking more at each of the flexible strategies reveals Flexible Plan 2, which leveraged

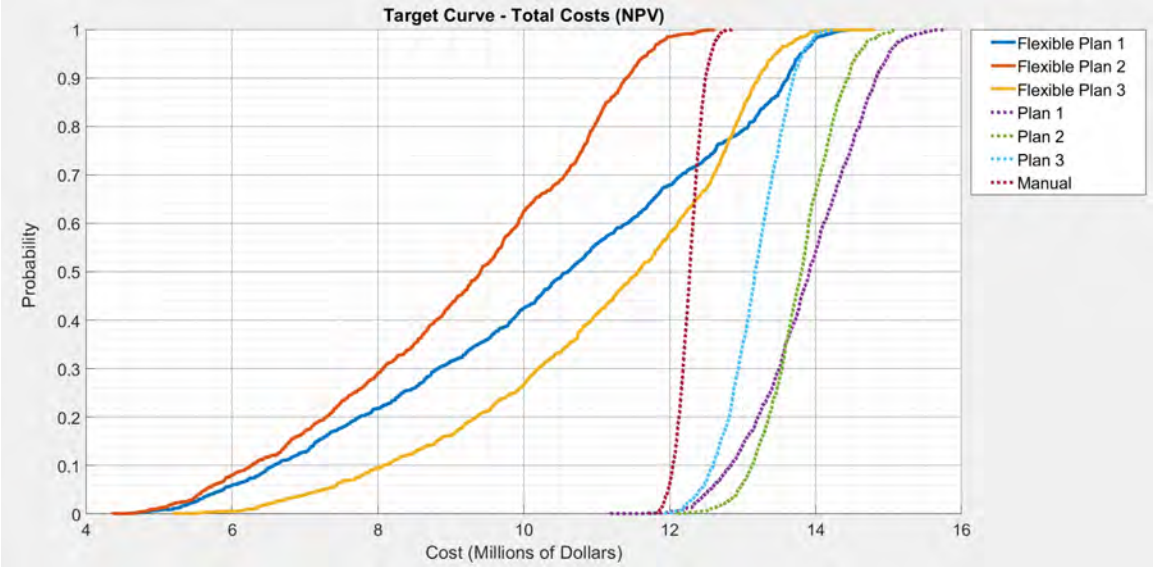


Figure 5-5: These cumulative distribution functions correspond to the distributions of total costs spent on inspection over the 30-year evaluation window. Each of the lines correspond to a different implementation strategy. Because cost is represented on the horizontal axis, lower values and distributions shifted left are desirable.

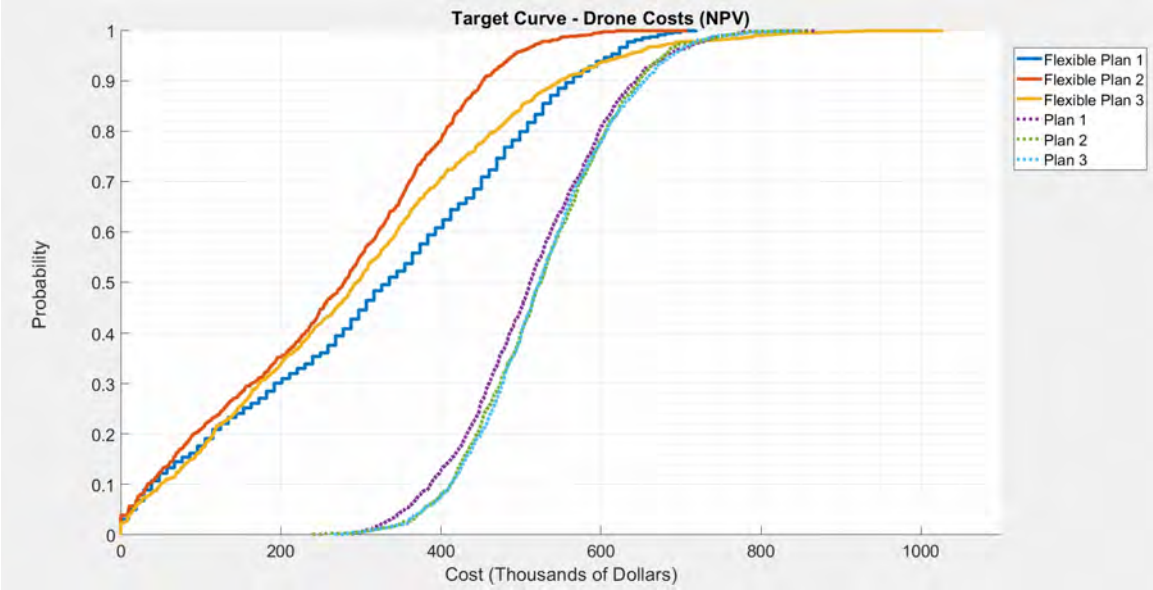


Figure 5-6: The cumulative distribution functions correspond to the distributions of total costs spent on drone equipment over the 30-year evaluation window. Each of the lines correspond to a different implementation strategy. Because cost is represented on the horizontal axis, lower values and distributions shifted left are desirable.

lower acquisition costs of drones, performed the best with respect to all evaluation criteria. However, Flexible Plan 1, which used crash rate as its decision rule also performed well, displaying similar target curve shape at an average cost of only 13% higher. Flexible Plan 3, which combined the decision rules for both Flexible Plan 1 2, looked to be the least desirable of the three flexible options. Flexible Plan 3 had a higher average value and a similar range and standard deviation to the other options when looking at total costs. Looking at the drone cost data; however, we see the characteristics of Flexible Plan 3 looking much closer to the best option (Flexible Plan 1) in terms of total cost. The likely explanation is that too many acquisition criteria delay the implementation of the drone system past some optimal point. While the initial uncertainty analysis in Figure 5-3 appears to favor delayed drone implementation, the diminished performance of Flexible Plan 3 seems to penalize more stringent decision rules that delay drone implementation. With this observation it is reasonable to hypothesize that the mechanisms within the system that reward delayed implementation compete with those that reward expedient implementation, and at some point, they theoretically reach an optimum. However, given uncertainty is the driving factor behind this analytical approach, it is inherently futile to try to search for this optimum balance mathematically, as with each different possible simulation, the data, and therefore optimum implementation scheme, would change. Rather it is most important to know these factors and relationships exist within the system and that one does not completely dominate the other. As certain parameters, like drone cost, are more accurately and precisely determined with time, adjustments can be made to the implementation plans.

5.5.3 Strategy Development

Due to the high uncertainty present in this model and the data used, a specific plan is not proposed by this analysis, rather a list of strategic options and considerations is presented, with generalized recommendations based on empirical findings. The first strategic consideration is that neither rapid implementation nor delayed phasing is heavily favored under the model assumptions. However, should any factors favoring delayed implementation like high drone costs, elevated crash probabilities, faster manual inspection times, or high drone inspection times exist, it may be best to put stringent performance criteria on rollout or wait for lower drone costs before widespread implantation is attempted. Conversely, should any factors favoring early implementation like a low discount rate, economies of scale, lower drone costs, consistent drone performance, fast inspection times, elevated manual labor costs, or rapid system improvement occur, it may be best to operate under lest stringent decision rules and favor rapid implementation. The combination of quantitative parameters used in this model may be thought of as a single point of reference with respect to these considerations. Incorporating this sort of flexibility into the rollout procedure may also reveal benefits not considered by the model at all. Because this is a new system technological complications or unforeseen conditions may arise that completely inhibit the system's ability to perform the inspection task. A monitored flexible option could allow for a return to manual method at early stages, reducing risks unaccounted for in the model. For this reason, though not tested in simulation, we recommend an extensive real world test study, to eliminate as many of these crash associated complication possibilities as well.

The next recommendation is that implementation begin, at least in small scale testing, as

soon as the system is technologically viable and capable of accomplishing the inspection task. While the data used in this study are estimates, results still give confidence that an automated drone-based inspection system could be implemented in a variety of ways, under a variety of circumstance, that all reduce the costs of current manual methods, while providing a full runway inspection capability previously unsupported. The addition of this capability, as well as the quantitative results of this model further support the existing sentiment that the development and eventual implementation of such an automated system has economic value potential.

Chapter 6

Conclusions

6.1 Autonomous System Performance

This study presents a viable approach and positive initial results toward automating airfield pavement condition assessments, specifically catered to the United States Air Force (USAF). The proposed automation utilizes state-of-the-art hardware and machine learning algorithms while remaining firmly within the existing framework of USAF pavement assessment and mission goals. There are several major conclusions from this current body of work, summarized as follows:

- It is possible to implement automated data collection and analysis procedures within the existing base maintenance mission and framework for the United States Air Force:
  - National Defense Authorization Act (NDAA) compliance creates strong standards to which all technology must adhere, but does not prohibit the implemen-

tation of automation for base maintenance and property condition assessment.

- Hardware selection is critical to mission success:
  - While required specifications are unique to each application, careful consideration must be given to Unmanned Aerial Vehicle (UAV)/pilot interaction and imager properties such as frame rate, field of view, and resolution.
  - When new imager and mounting components are applied to a drone, the system should be field tested at different speeds and wind conditions to ensure minimal vibrational interference and image distortion.
- Machine learning algorithms can be applied to pavement conditions:
  - The quality of the images and their corresponding labels significantly affect model performance.
  - Both cracking and utility patching distresses can be detected on a pixel-by-pixel basis.
  - Far more data will be required to train better models and evaluate if this approach is generalizable to other runways, with different backgrounds, and more distress types.

Building on this body of work, there is more work to be done to ensure successful implementation of automated data collection and analysis procedures. The author suggests the following items as most critical to project success:

- New UAV and imaging technologies must continue to be developed and allowed to operate:

  - The number of NDAA compliant hardware and software components, like autopilots, cameras, GPS sensors, and ground control stations are limited. New products or commercialized packages could help optimize cost and performance.
  - Drone technology must become reliable and resilient enough to operate safely in a variety of environmental conditions.
  - Airfield administration must begin to allow more drone operations over runways to identify problems and continue hardware and software development.

- Data Acquisition and Management
  - Significantly more data of different airfields and different distresses must be obtained and made available to expand the CNN approach to more airfields.
  - A PCI for a concrete pavement airfield requires a slightly different analysis methods. While a flexible pavement inspection measures the linear feet or square feet of a distress in a sample unit, concrete pavements do not require physical measurements and instead count the number of concrete slabs (within a 20 slab sample unit) that contain a specific distress. This would require distress detection and identification for each slab, but not necessarily measurement. Therefore, the image segmentation performed on flexible pavements in this research, which is a more difficult CNN task, may not be necessary. Because concrete pavements only require slab counts for each distress, a simpler CNN may be used to identify or place bounding boxes around different distress types in concrete pavements instead.

  - Greater investments in computational power and data storage will be needed to accommodate and use the data generated by autonomous systems.
  - While cracks and patches are distinguishable with visible image data, expanding detection to other distress types may prove difficult. Other types of sensors (ex. Lidar/3D scanning) may be better suited for detecting other distress types (ex. shoving, rutting, swell)

- Capitalizing on Advances in Machine Learning
  - Distress detection accuracy is a major bottleneck to improving system generalizability, accuracy, and precision.
  - Significant advances in machine learning approaches and algorithms occur often and should be monitored for replacing the existing algorithm.

6.2 Implementation and Flexibility Analysis

Implementing this sort of flexible strategy in the military's organizational structure also presents logistic and organizational challenges. The first is that there are constantly changing administrative regulations on drone flight over an installation that may be locationally dependent. While the current policy within the Air Force structure is already gaining momentum toward alleviating restrictions, or streamlining approval processes, this would still require significant planning prior to drone system implementation to secure the necessary approval for operations, assuming such approval even exits at the desired time of implementation. Additionally, continued monitoring of drone systems used across the globe

may prove difficult if all systems function independently from one another and must rely on human reports of system performance. Consequently, it may prove beneficial to incorporate a linked data monitoring system across all systems that can automatically update the status and performance of each drone. This capability would make centralized monitoring easily possible and provide accurate data for triggering expansion decision rules. If this sort of centralized monitoring system is in place it would also make sense to have a single person or team responsible for the complete implementation from start to finish. This continuity would not only prevent any loss information regarding flexible implementation but would also enable continued learning and improvement of the system until adequate performance is reached and monitoring can be reduced solely to maintenance purposes.

Additionally, this analysis gives us insights into what technical design parameters ought to be considered and prioritized when developing or selecting a drone platform for implementation. Clearly intuition, as well as the sensitivity analysis indicate total cost of the drone system is a highly influential variable and should be minimized. What this model also reveals, however, is that drone crash probability just as significant, if not more significant of a parameter than upfront cost. The model assumes that with every crash event, a full replacement system will need to be purchased, however, if drones are developed that either reduce the probability of crashing or do not require total replacement in the event of crash, this will make the automated system even more cost efficient. For this reason, ruggedness of design, reliability, and ease of repair should be a heavily valued characteristic of whatever drone system is developed or selected to carry out this task.

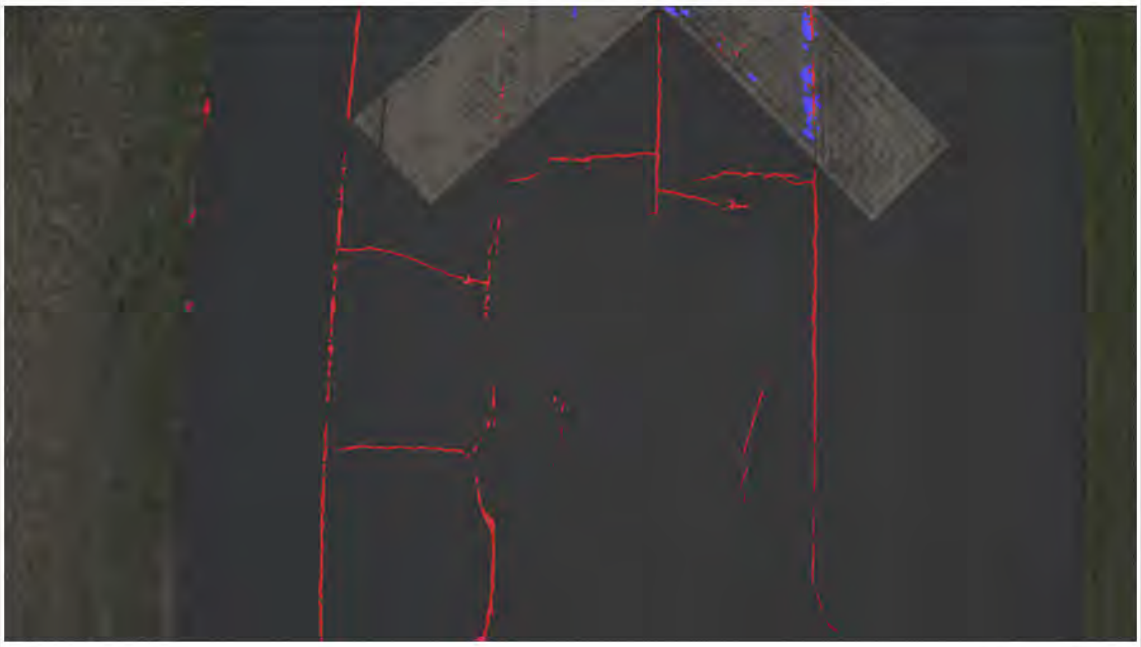
Disclaimer

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government

Appendix A

PCI Image Analysis

While the main body of the text presents the calculation results this appendix provides a detailed summary and analysis on the performance of the CNN detection and physical measurement calculation.



Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 1
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.991
Detection IoU Score (Cracks): 0.296
Detection IoU Score (Patches): 0

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	0	8.5	-8.5
Crack Length (ft)	320	224.9	95.1
PCI	54	59	-9.3%

Comments:
Patch Area Detection: There are only false positive detections here. The presence of paint could be affecting the detection.
Crack Length Detection: Many cracks are missed by the algorithm with some small false positive detection on the edges where the pavement meets the grass.
Patch Area Calculation: The calculated area could be plausible.
Crack Length Calculation: It is logical that the automated crack length is much less than the manual value given the number of false negative detections.
Distress Severity: Most crack severity levels are "high" in both auto and manual inspections, which causes good agreement between PCI values.
Overall Conclusion: Disparity between manual and auto is mostly accounted for by the CNN's false negative crack detections, not the physical measurement calculations. It is logical that the PCI of the manual inspection is less given the crack length measured is greater in the manual method.

Figure A-1: Sample Unit 1

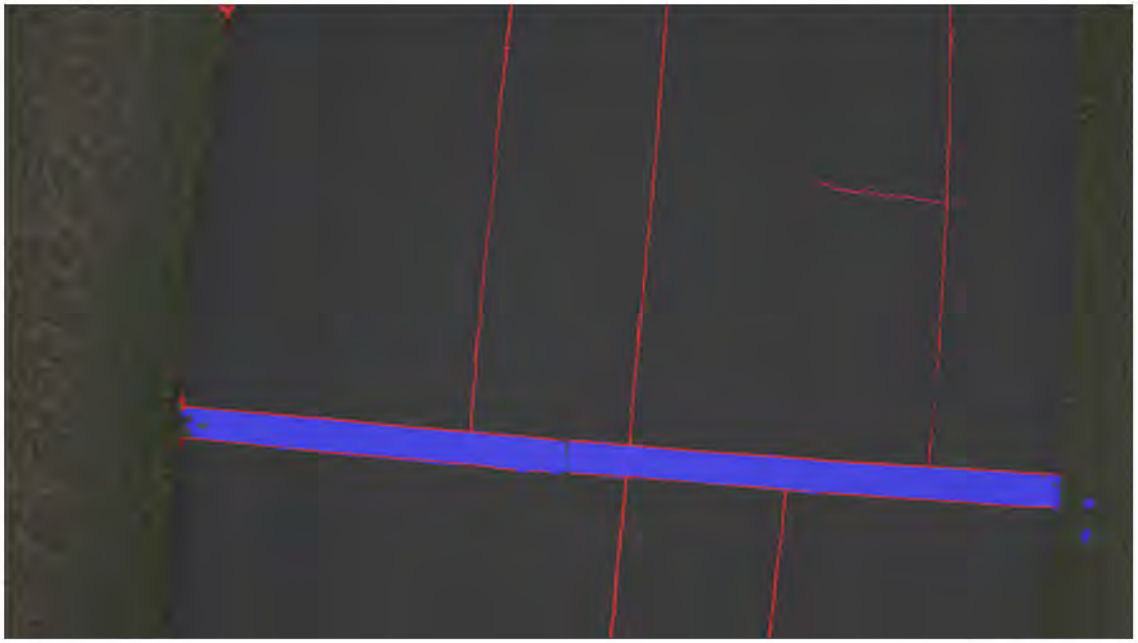
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 2
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.982
Detection IoU Score (Cracks): 0.221
Detection IoU Score (Patches): 0.785

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	278	174.2	103.8
Crack Length (ft)	310	355.1	-45.1
PCI	52	49	5.8%

Comments:
Patch Area Detection: There are mostly false negative detections. Small patches of false positives are present in painted areas and in the grass.
Crack Length Detection: Cracks are missed by the algorithm in painted areas. Small false positive detection is present on the edges where the pavement meets the grass.
Patch Area Calculation: A smaller automated area is expected given the large presence of false negatives.
Crack Length Calculation: Though there appear to be many false negatives, the automated value is higher. This may indicate a small overestimation in the calculation method.
Distress Severity: Most crack severity levels are "high" in both auto and manual inspections, which causes good agreement between PCI values.
Overall Conclusion: Disparity between manual and auto is mostly accounted for by greater crack length. It is logical that the PCI of the auto inspection is less given the crack length measured is greater in the automated method.

Figure A-2: Sample Unit 2



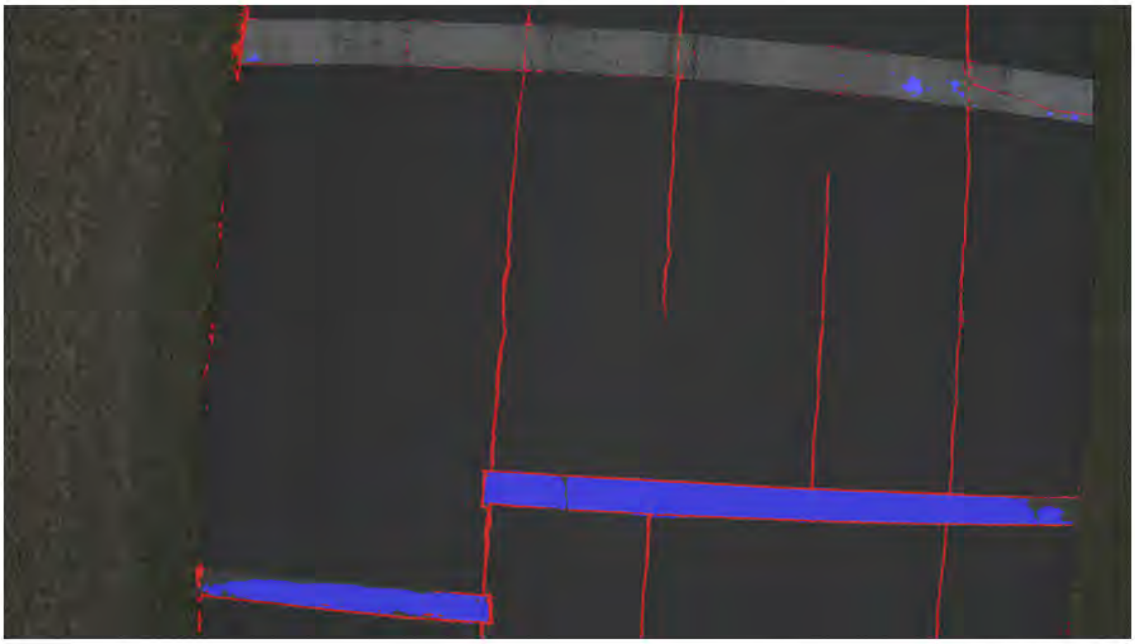
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 3
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.992
Detection IoU Score (Cracks): 0.344
Detection IoU Score (Patches): 0.940

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	200	223.8	-23.8
Crack Length (ft)	375	314.1	60.9
PCI	45	68	-51.1%

Comments:
Patch Area Detection: Detection appears near perfect with equally small amounts of false positives and false negatives.
Crack Length Detection: Only a few false negatives present.
Patch Area Calculation: Very small disparity between values is expected, error in manual method or automated calculation assumptions could explain the difference.
Crack Length Calculation: It would not appear that all 60ft of difference between methods is accounted by false negatives, suggest auto method may underestimate.
Distress Severity: Half of the cracks are labeled "high" severity in the manual, but all are marked as "medium" in the auto. Most of the patching is marked as "low" in the manual, but "medium" in the auto.
Overall Conclusion: The CNN detected the distresses well. Disparity between manual and auto is likely due to underestimated crack length calculation as well as underestimated crack severity by the auto method.

Figure A-3: Sample Unit 3



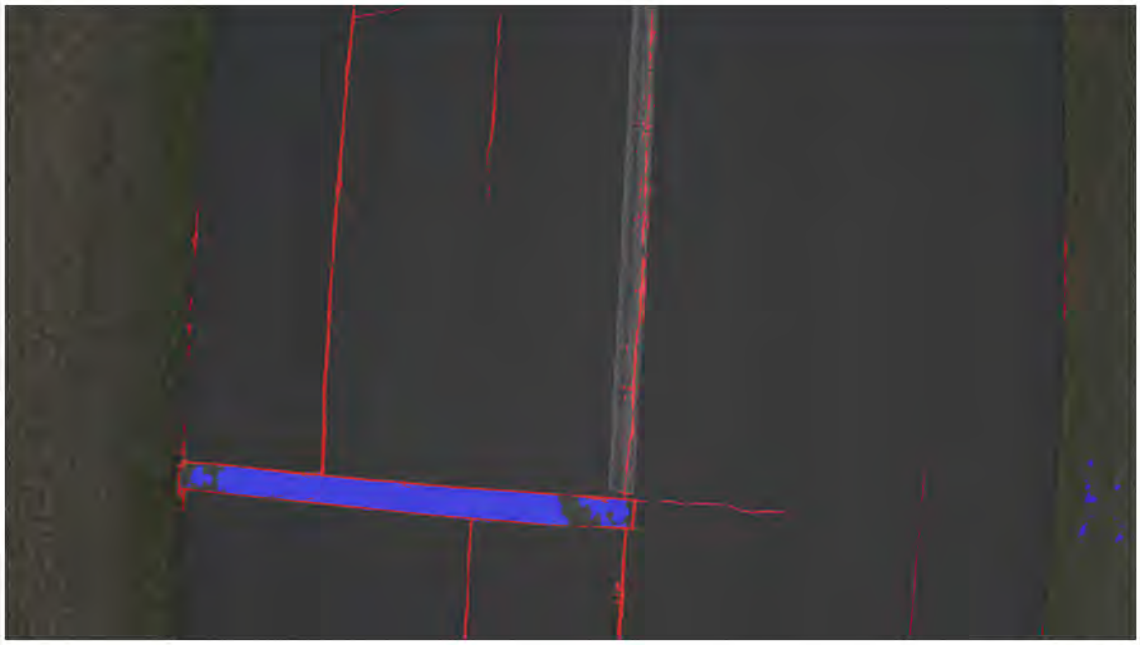
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 4
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.985
Detection IoU Score (Cracks): 0.271
Detection IoU Score (Patches): 0.862

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	200	193.8	6.2
Crack Length (ft)	357	390.6	-33.6
PCI	47	46	2.1%

Comments:
Patch Area Detection: There are mostly false negative detections. Small patches of false positives are present in painted areas.
Crack Length Detection: False positive detection is present on the edges where the pavement meets the grass.
Patch Area Calculation: A smaller automated area is expected given the presence of false negatives.
Crack Length Calculation: The many false positive detections on the edge seem to account for the overestimation in the auto measurement.
Distress Severity: Most of the cracks are labeled "high" severity in the manual, but all are marked as "medium" in the auto. Most of the patching is marked as "low" in the manual, but "medium" in the auto.
Overall Conclusion: Differences in PCI are expected due to differences in distress severity, but the accuracy of the PCI still matches between auto and manual. The underestimation in crack severity for the auto method, but overestimation in patch severity may be the reason PCI values remain similar between auto and manual.

Figure A-4: Sample Unit 4



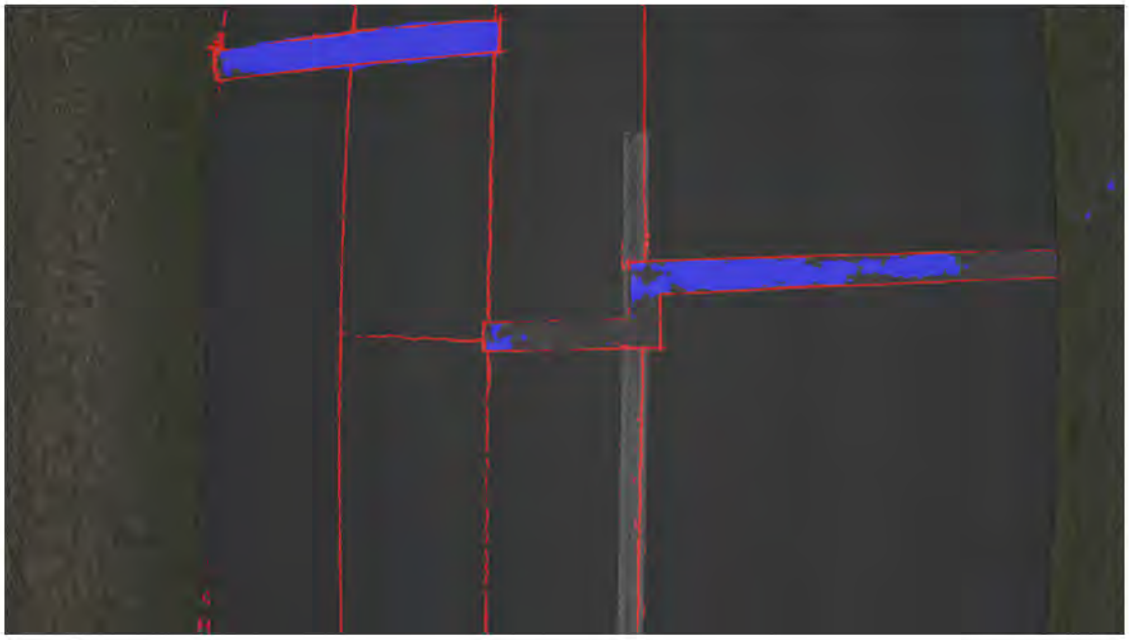
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 5
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.990
Detection IoU Score (Cracks): 0.244
Detection IoU Score (Patches): 0.804

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	80	88.6	-8.6
Crack Length (ft)	200	275.9	-75.9
PCI	59	52	11.9%

Comments:
Patch Area Detection: There are mostly false negative detections. Small patches of false positives are present in the grass.
Crack Length Detection: False positive detection is present on the edges where the pavement meets the grass and in the painted areas.
Patch Area Calculation: A smaller automated area is expected given the presence of false negatives; however, the larger value indicates the calculation may be slightly overestimating from the detection image.
Crack Length Calculation: The false positive detections on the edge and paint may account for the overestimation in the auto measurement.
Distress Severity: All cracks are labeled "high" severity in the manual and auto. Most of the patching is marked as "low" in the manual, but "medium" in the auto.
Overall Conclusion: A lower PCI value for the auto method is expected due to a greater crack length value reported by the auto method. The difference in crack length seems to be a result of the CNN's false positive crack detections.

Figure A-5: Sample Unit 5



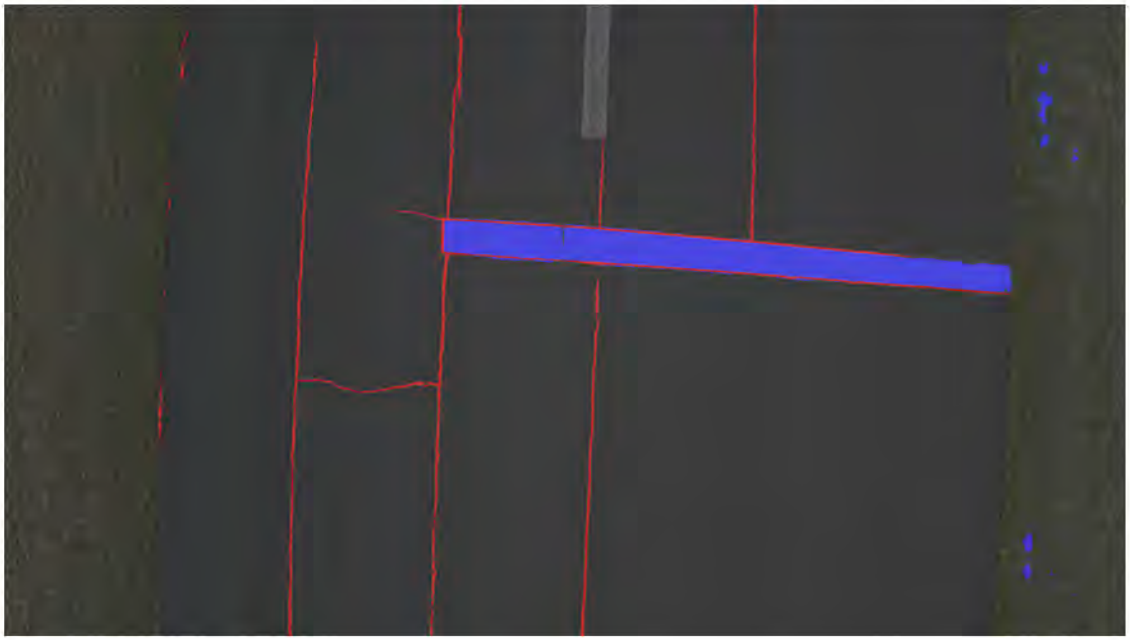
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 6
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.975
Detection IoU Score (Cracks): 0.255
Detection IoU Score (Patches): 0.578

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	160	139.1	20.9
Crack Length (ft)	370	353.5	16.5
PCI	44	49	-11.4

Comments:
Patch Area Detection: There are mostly false negative detections. Small patches of false positives are present in the grass.
Crack Length Detection: False negative detection is present on the edges of utility patches.
Patch Area Calculation: A smaller automated area is expected given the presence of false negatives.
Crack Length Calculation: The false negative detections on the edge of patches may account for the difference in lengths measured.
Distress Severity: Most cracks are labeled "high" severity in the manual and auto. Most of the patching is marked as "low" in the manual, but "medium" in the auto.
Overall Conclusion: The PCI score reported by the auto is greater than the manual and this is likely a result of poor CNN distress detection. The CNN shows many false negatives for crack and patch detection.

Figure A-6: Sample Unit 6



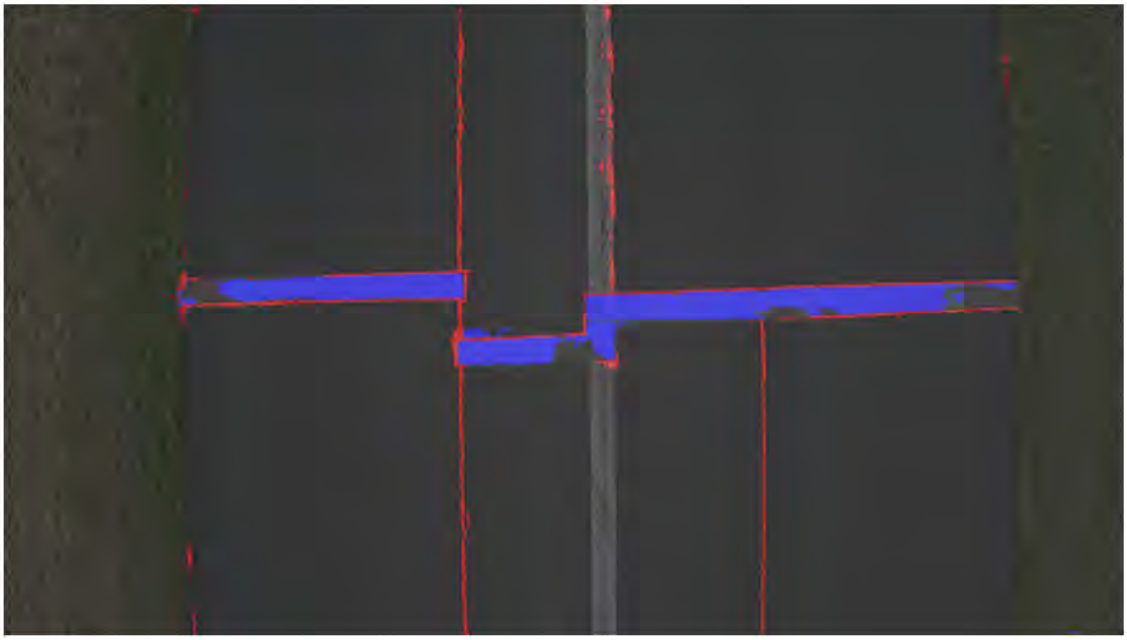
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 7
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.991
Detection IoU Score (Cracks): 0.314
Detection IoU Score (Patches): 0.911

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	100	151.1	-51.1
Crack Length (ft)	315	294.8	20.2
PCI	52	52	0%

Comments:
Patch Area Detection: There are mostly false positive detections in the grass.
Crack Length Detection: False positive detection is present on the edges where the pavement meets the grass. False negatives exist on the edge of patches
Patch Area Calculation: A larger automated area is expected given the presence of false negatives; however, the much larger value indicates the calculation may be slightly overestimating from the detection image.
Crack Length Calculation: Close values are expected due to only a mall amount of both false positive and false negative detections.
Distress Severity: All cracks are labeled "high" severity in the manual and auto. Most of the patching is marked as "low" in the manual, but "medium" in the auto.
Overall Conclusion: Good agreement between PCI values for the auto and manual methods is expected given similar physical measurements, good CNN detection performance, and similar distress severity determinations.

Figure A-7: Sample Unit 7



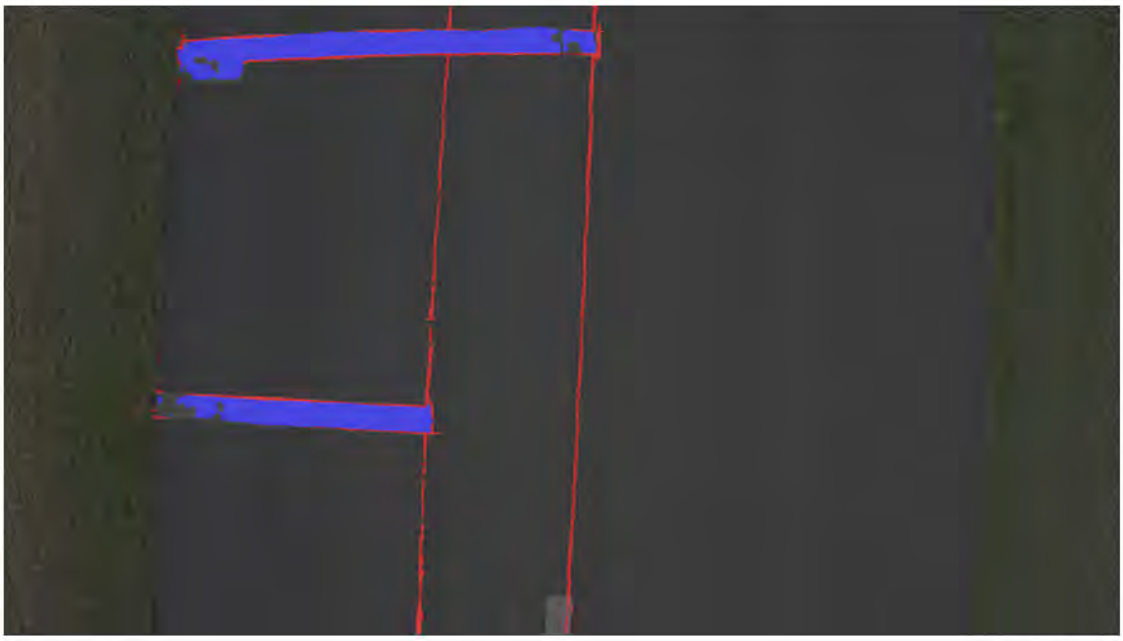
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 8
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.984
Detection IoU Score (Cracks): 0.244
Detection IoU Score (Patches): 0.737

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	150	157.1	-7.1
Crack Length (ft)	280	274.8	5.2
PCI	52	55	-5.8%

Comments:
Patch Area Detection: There are mostly false negative detections. Small patches of false positives are present in the grass.
Crack Length Detection: False positive detection is present on the edges where the pavement meets the grass and in the painted areas.
Patch Area Calculation: A smaller automated area is expected given the presence of false negatives; however, the larger value indicates the calculation may be slightly overestimating from the detection image.
Crack Length Calculation: The false positive detections on the edge and paint may account for the overestimation in the auto measurement. False negatives crack detection is present on the edge of patches.
Distress Severity: All cracks are labeled "high" severity in the manual and auto. Most of the patching is marked as "medium" in the manual and auto.
Overall Conclusion: Good agreement between PCI values for the auto and manual methods is expected given similar physical measurements, good CNN detection performance, and similar distress severity determinations.

Figure A-8: Sample Unit 8



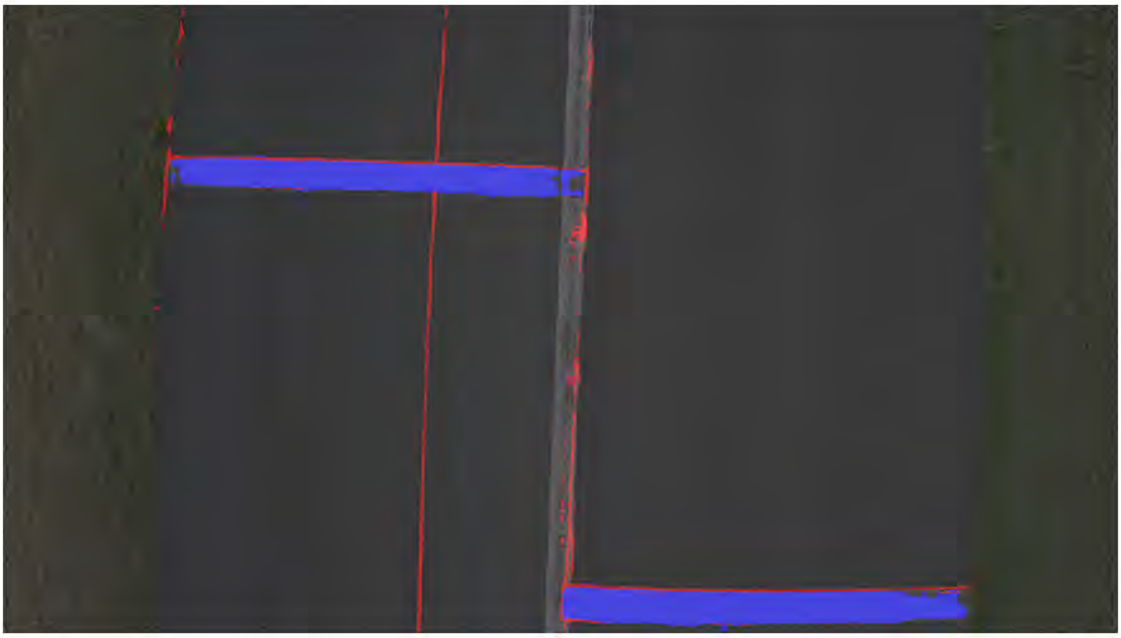
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 9
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.991
Detection IoU Score (Cracks): 0.271
Detection IoU Score (Patches): 0.859

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	160	133.5	26.5
Crack Length (ft)	250	209.8	40.2
PCI	54	59	-9.3%

Comments:
Patch Area Detection: There are mostly false negative detections.
Crack Length Detection: False negative detection is present on the edges of patches.
Patch Area Calculation: A smaller automated area is expected given the presence of false negatives.
Crack Length Calculation: The false negative detections on the edge of patches may account for the lower value in the auto method.
Distress Severity: Most cracks are labeled "high" severity in the manual and auto. Most of the patching is marked as "low" in the manual, but "medium" in the auto.
Overall Conclusion: The PCI score reported by the auto is greater than the manual and this is likely a result of poor CNN distress detection. The CNN shows many false negatives for crack and patch detection.

Figure A-9: Sample Unit 9



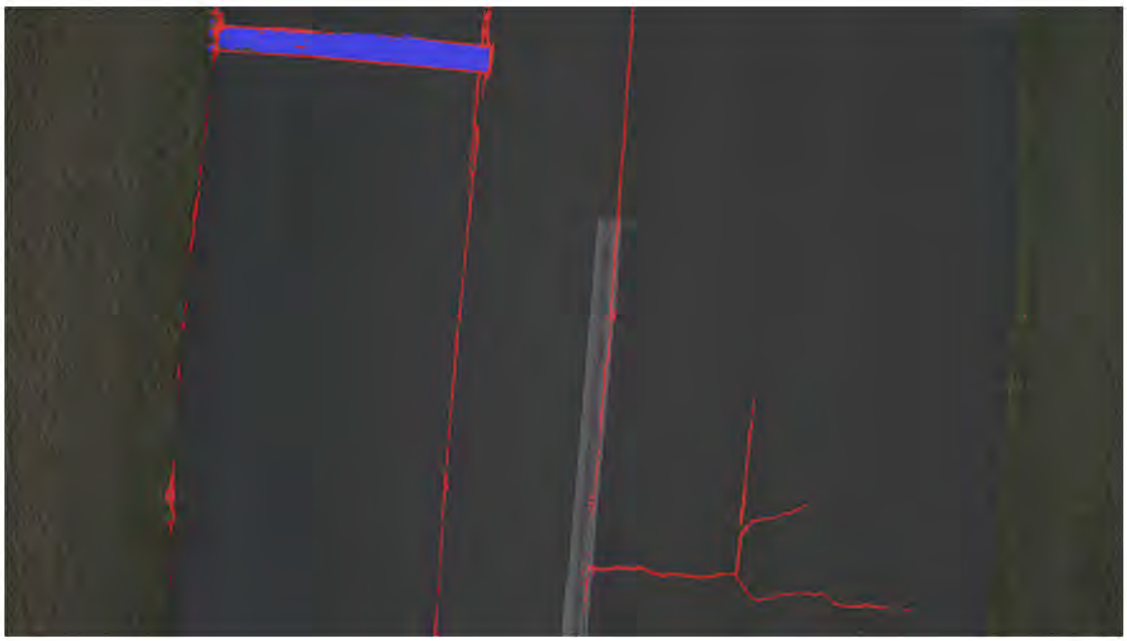
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 10
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.991
Detection IoU Score (Cracks): 0.253
Detection IoU Score (Patches): 0.908

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	150	199.8	-49.8
Crack Length (ft)	210	229.8	-19.8
PCI	59	55	6.8%

Comments:
Patch Area Detection: There are a few false negative detections.
Crack Length Detection: False negative detection is present on the edges of patches.
Patch Area Calculation: A smaller automated area is expected given the presence of a few false negative detections; however, we observe a larger auto patch area. This may mean the manual method had measurement errors or the auto area calculation overestimated.
Crack Length Calculation: The false negative detections on the edge of patches may account for the lower value in the auto method.
Distress Severity: Most cracks are labeled "high" severity in the manual and auto. Most of the patching is marked as "low" in the manual, but "medium" in the auto.
Overall Conclusion: The PCI score reported by the auto is less than the manual and this is likely a result of the auto's patch area being overestimated and a higher severity level.

Figure A-10: Sample Unit 10



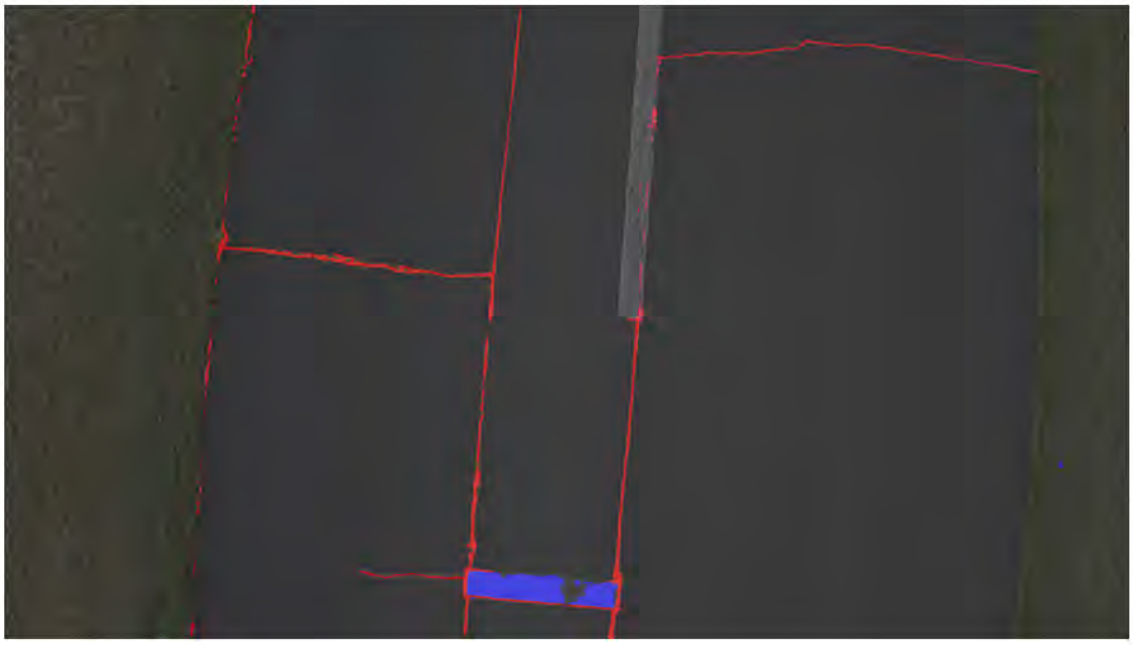
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 11
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.992
Detection IoU Score (Cracks): 0.289
Detection IoU Score (Patches): 0.840

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	50	53.8	-3.8
Crack Length (ft)	252	251.4	0.6
PCI	55	55	0%

Comments:
Patch Area Detection: Detection appears accurate.
Crack Length Detection: There are false negative detections in the painted areas and false negatives on the edge where the pavement meets the grass.
Patch Area Calculation: The calculated area makes sense.
Crack Length Calculation: The false negatives and false positives may have balanced each other leading to very similar measurements.
Distress Severity: Distress Severity: Most cracks are labeled "high" severity in the manual and auto. Most of the patching is marked as "low" in the manual, but "medium" in the auto.
Overall Conclusion: Good agreement between PCI values for the auto and manual methods is expected given similar physical measurements, good CNN detection performance, and similar distress severity determinations.

Figure A-11: Sample Unit 11



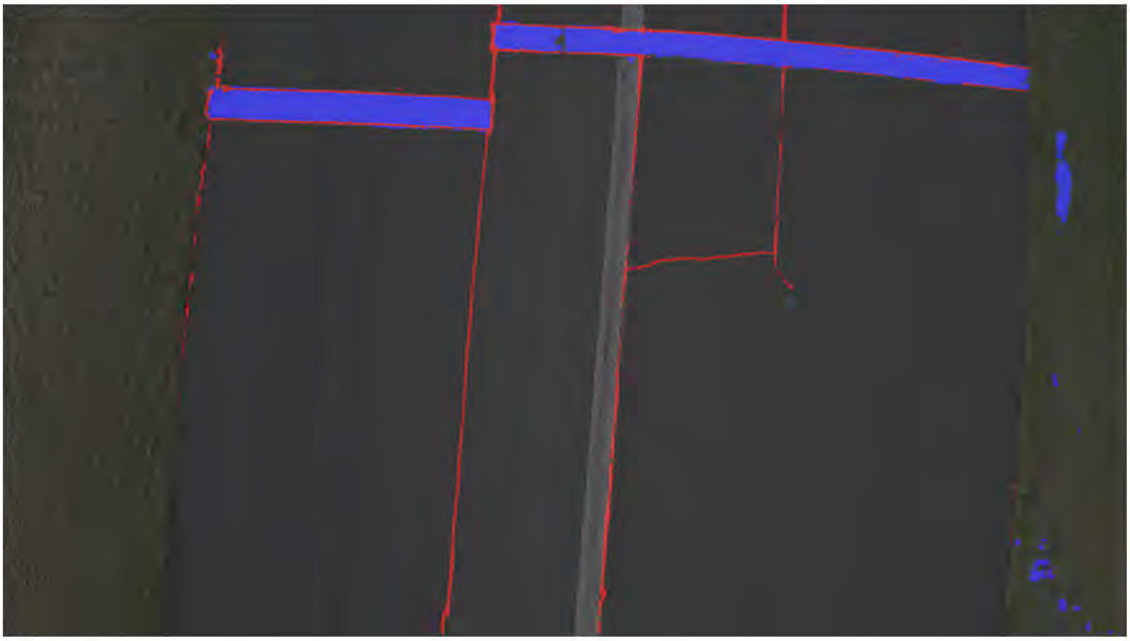
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 12
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.992
Detection IoU Score (Cracks): 0.255
Detection IoU Score (Patches): 0.791

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	100	27	73
Crack Length (ft)	265	223.3	41.7
PCI	55	59	-7.3%

Comments:
Patch Area Detection: There are mostly false negative detections.
Crack Length Detection: There are false negative detections in the painted areas and false negatives on the edge where the pavement meets the grass.
Patch Area Calculation: There may be underestimation in the calculation or in this instance there was likely error in the manual record as this patch looks a little too small for 100sqft.
Crack Length Calculation: The false negatives in the painted areas may account for the lower auto crack measurement value.
Distress Severity: Distress Severity: Most cracks are labeled "high" severity in the manual and auto. Most of the patching is marked as "low" in the manual, but "medium" in the auto.
Overall Conclusion: The PCI score reported by the auto is greater than the manual and this is likely a result of poor CNN distress detection. The CNN shows many false negatives for crack and patch detection.

Figure A-12: Sample Unit 12



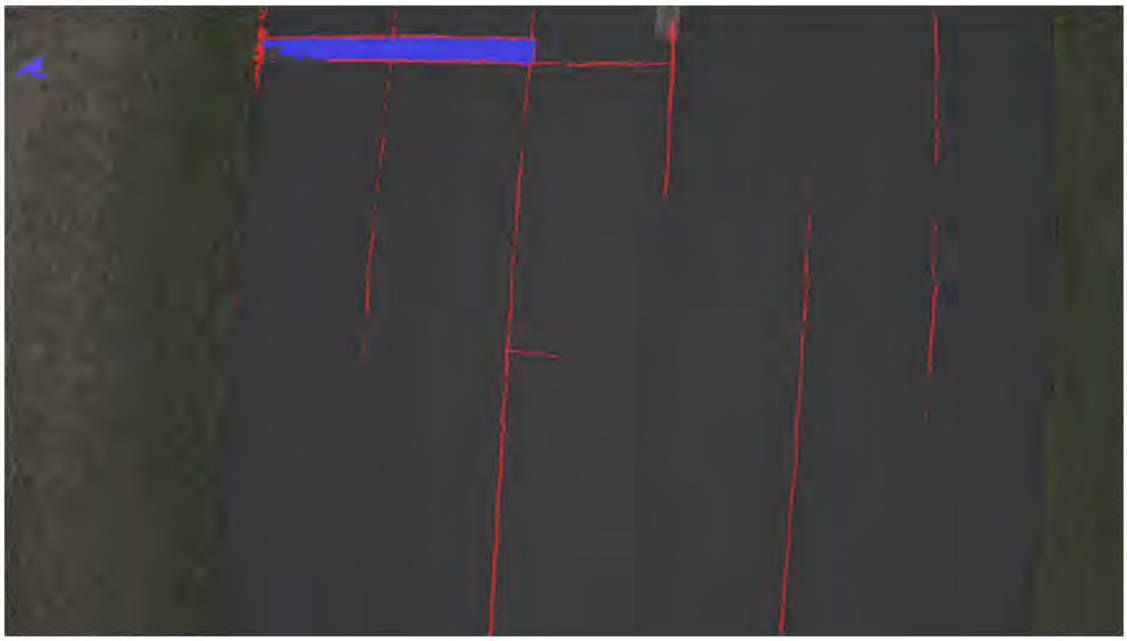
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 13
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.988
Detection IoU Score (Cracks): 0.266
Detection IoU Score (Patches): 0.859

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	150	185.2	-35.2
Crack Length (ft)	300	300.1	-0.1
PCI	52	52	0%

Comments:
Patch Area Detection: There are some false positive detections in the grass.
Crack Length Detection: There are false negative detections in the painted areas and false negatives on the edge where the pavement meets the grass.
Patch Area Calculation: The false positives in the grass mean a higher automated square footage is expected.
Crack Length Calculation: The false negatives and false positives may have balanced each other leading to very similar measurements.
Distress Severity: Distress Severity: All cracks are labeled "high" severity in the manual and auto. All patching is marked as "medium" in the manual and auto.
Overall Conclusion: Because the measurements and severities mostly align it makes sense that the PCIs align.

Figure A-13: Sample Unit 13



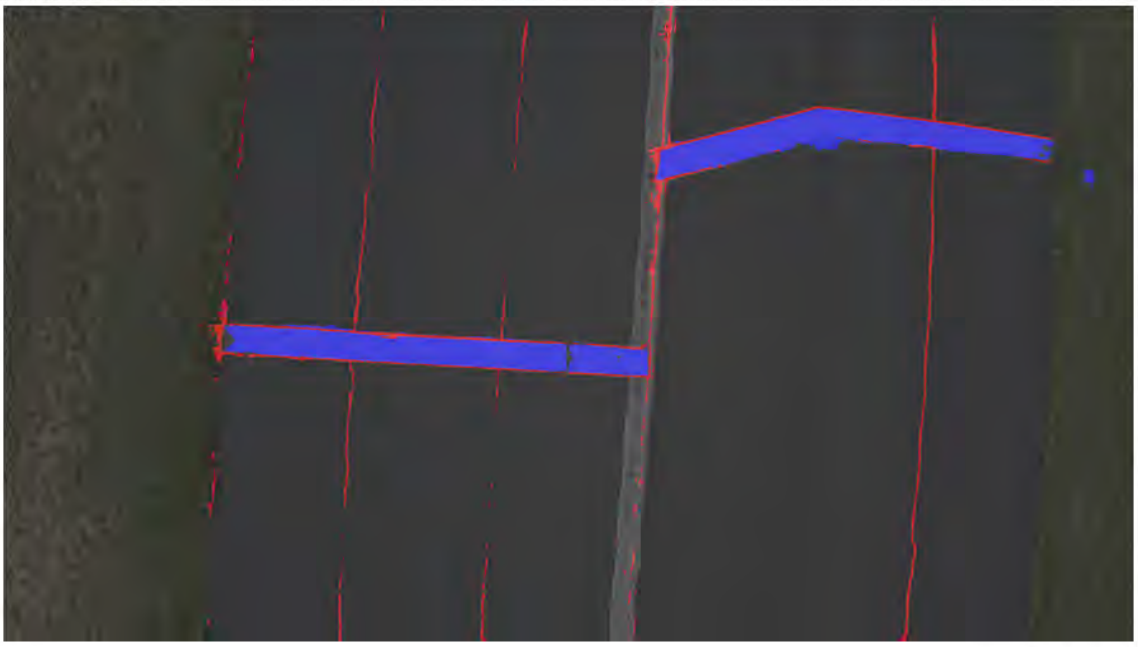
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 14
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.993
Detection IoU Score (Cracks): 0.338
Detection IoU Score (Patches): 0.820

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	50	49.7	0.3
Crack Length (ft)	260	236.6	23.4
PCI	51	55	-7.8%

Comments:
Patch Area Detection: There are very small patches of false negatives and equally small patches of false positives in the grass.
Crack Length Detection: Many cracks are missed by the algorithm with some small false positive detection on the edges where the pavement meets the grass.
Patch Area Calculation: The calculated area could be plausible.
Crack Length Calculation: It is logical that the automated crack length is less than the manual value given the number of false negative detections.
Distress Severity: Most cracks are labeled "high" severity in the manual and auto. Most of the patching is marked as "low" in the manual, but "medium" in the auto.
Overall Conclusion: Disparity between manual and auto is mostly accounted for by false negative crack detection. It is logical that the PCI of the auto inspection is greater given the crack length measured is greater in the manual method.

Figure A-14: Sample Unit 14



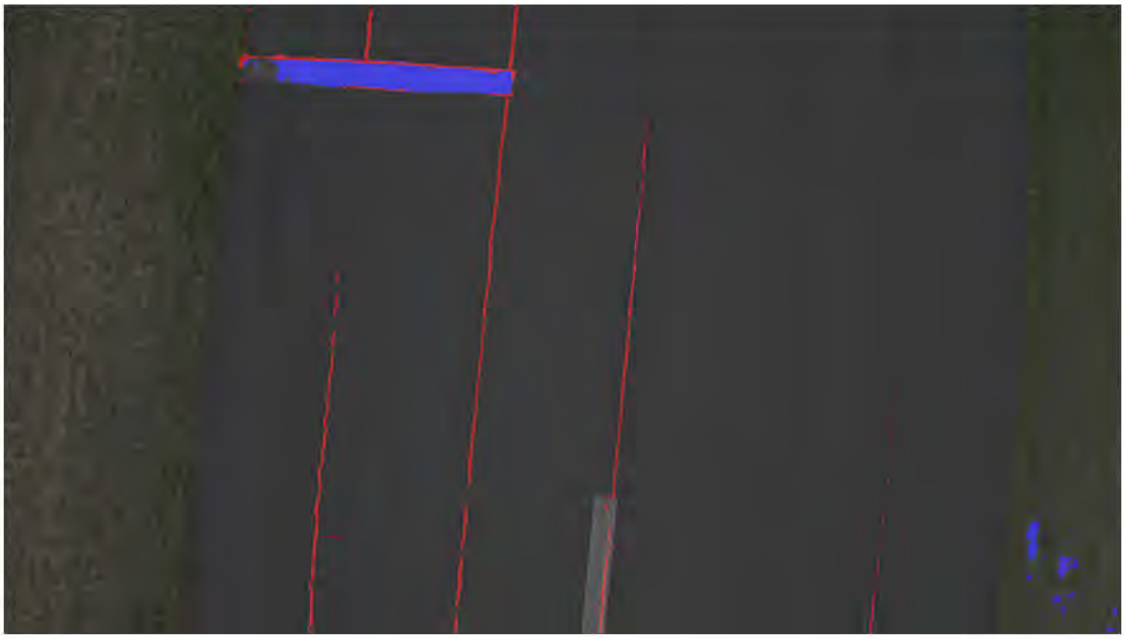
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 15
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.989
Detection IoU Score (Cracks): 0.284
Detection IoU Score (Patches): 0.897

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	150	184.4	-34.4
Crack Length (ft)	380	341.8	38.2
PCI	44	49	-11.4%

Comments:
Patch Area Detection: There is some false positive detection in the grass
Crack Length Detection: Many cracks are missed by the algorithm in painted areas with some small false positive detection on the edges where the pavement meets the grass.
Patch Area Calculation: The calculated auto area should be greater than the manual given the number of false positives.
Crack Length Calculation: It is logical that the automated crack length is much less than the manual value given the number of false negative detections.
Distress Severity: Most cracks are labeled "high" severity in the manual and auto. Most of the patching is marked as "low" in the manual, but "medium" in the auto.
Overall Conclusion: The PCI score reported by the auto is greater than the manual and this is likely a result of poor CNN distress detection. The CNN shows many instances of alse negatives for crack detection.

Figure A-15: Sample Unit 15



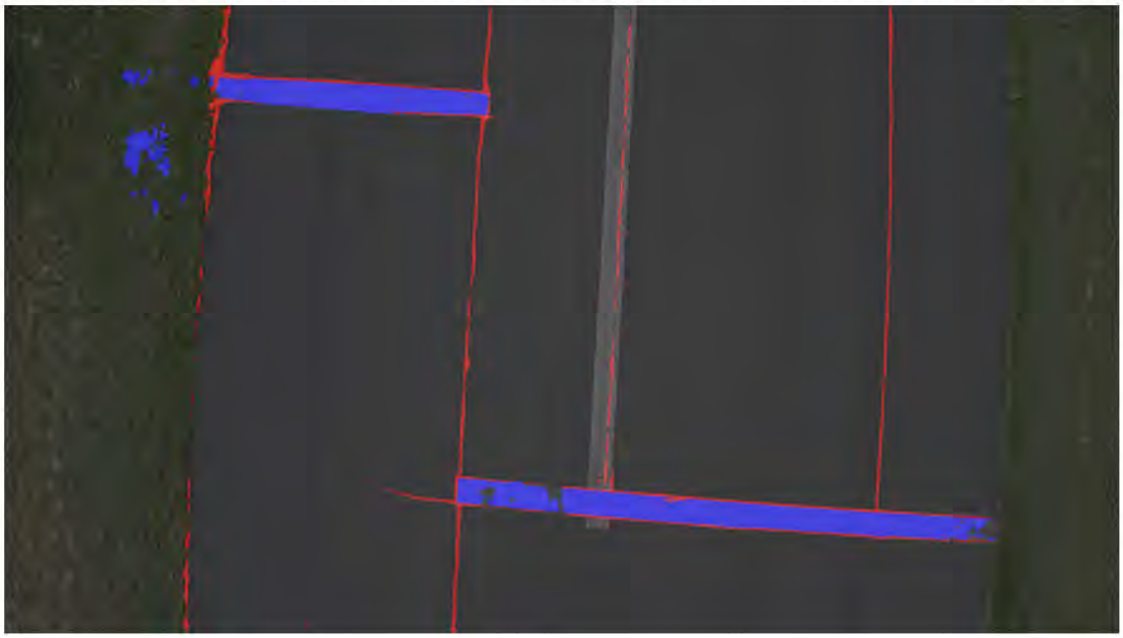
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 16
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.993
Detection IoU Score (Cracks): 0.330
Detection IoU Score (Patches): 0.738

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	50	54.5	-4.5
Crack Length (ft)	241	185.7	55.3
PCI	57	59	-3.5%

Comments:
Patch Area Detection: There is some false positive detection in the grass and false negatives in the patch.
Crack Length Detection: Many cracks are missed by the algorithm in painted areas and areas where there are thin cracks.
Patch Area Calculation: The calculated auto area should be roughly the same as the manual given the false positives and false negatives.
Crack Length Calculation: It is logical that the automated crack length is much less than the manual value given the number of false negative detections.
Distress Severity: Most cracks are labeled "high" severity in the manual and auto. Most of the patching is marked as "low" in the manual, but "medium" in the auto.
Overall Conclusion: The PCI score reported by the auto is greater than the manual and this is likely a result of poor CNN distress detection. The CNN shows many instances of false negatives for crack detection.

Figure A-16: Sample Unit 16



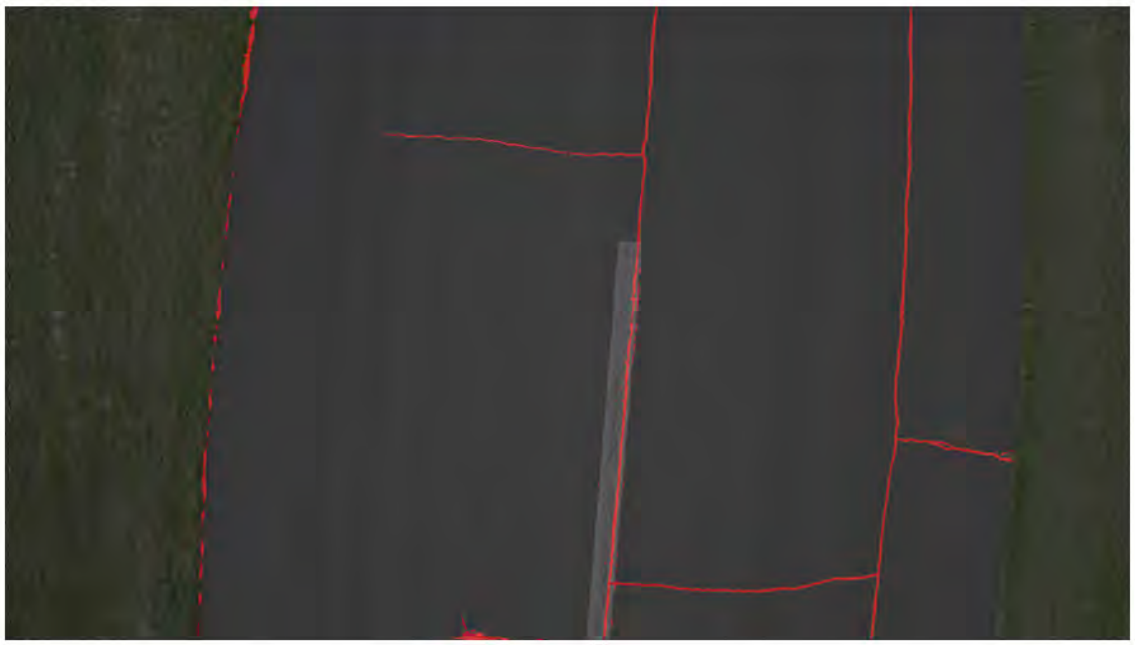
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 17
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.987
Detection IoU Score (Cracks): 0.252
Detection IoU Score (Patches): 0.842

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	150	172.8	-22.8
Crack Length (ft)	310	332.1	-22.1
PCI	49	49	0%

Comments:
Patch Area Detection: There are false positive detections in the grass.
Crack Length Detection: Some cracks are missed by the algorithm in painted areas. Some small false positive detections appear on the edges where the pavement meets the grass.
Patch Area Calculation: False positive detections mean the calculated area could be plausible.
Crack Length Calculation: It is logical that the automated crack length is greater than the manual given the false positive detection along the runway's edge.
Distress Severity: Most crack severity levels are "high" in both auto and manual inspections. Most patch severity is "low" in the manual but "medium" in the auto method.
Overall Conclusion: Good agreement between PCI values is expected given similar measurements and distress severities used.

Figure A-17: Sample Unit 17



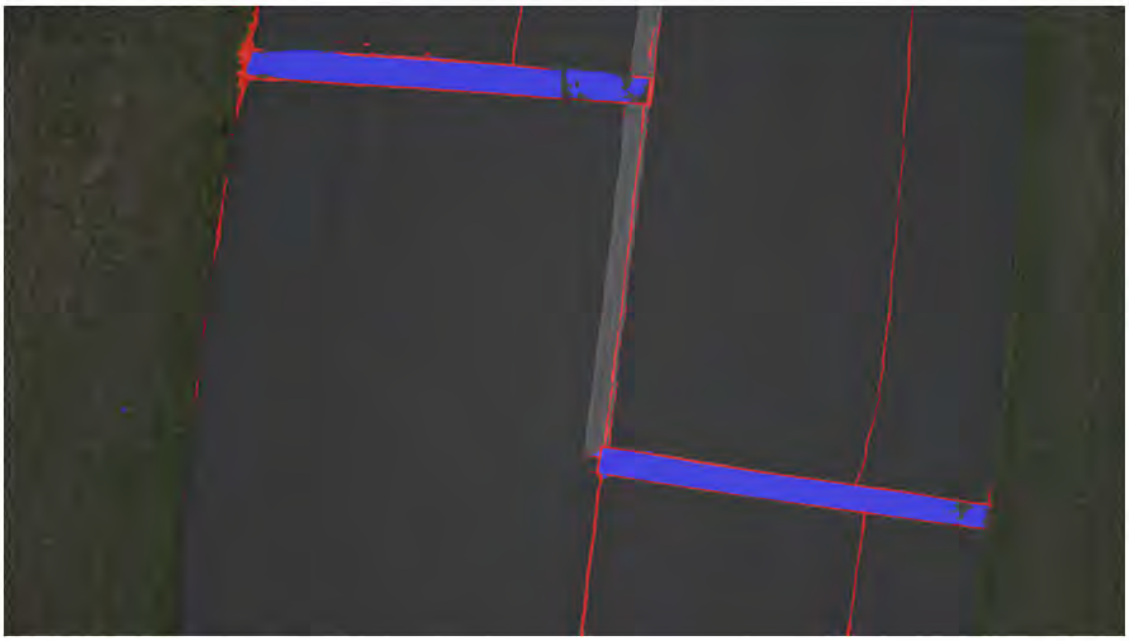
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 18
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.994
Detection IoU Score (Cracks): 0.326
Detection IoU Score (Patches): NA

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	0	0	0
Crack Length (ft)	215	227.6	-12.6
PCI	59	60	-1.7%

Comments:
Patch Area Detection: No patches are present, and none were detected.
Crack Length Detection: Some small false positive detections appear on the edges where the pavement meets the grass.
Patch Area Calculation: N/A
Crack Length Calculation: It is logical that the automated crack length is greater than the manual given the false positive detection along the runway's edge.
Distress Severity: Most crack severity levels are "high" in both auto and manual inspections.
Overall Conclusion: Good agreement between PCI values is expected given similar measurements and distress severities used.

Figure A-18: Sample Unit 18



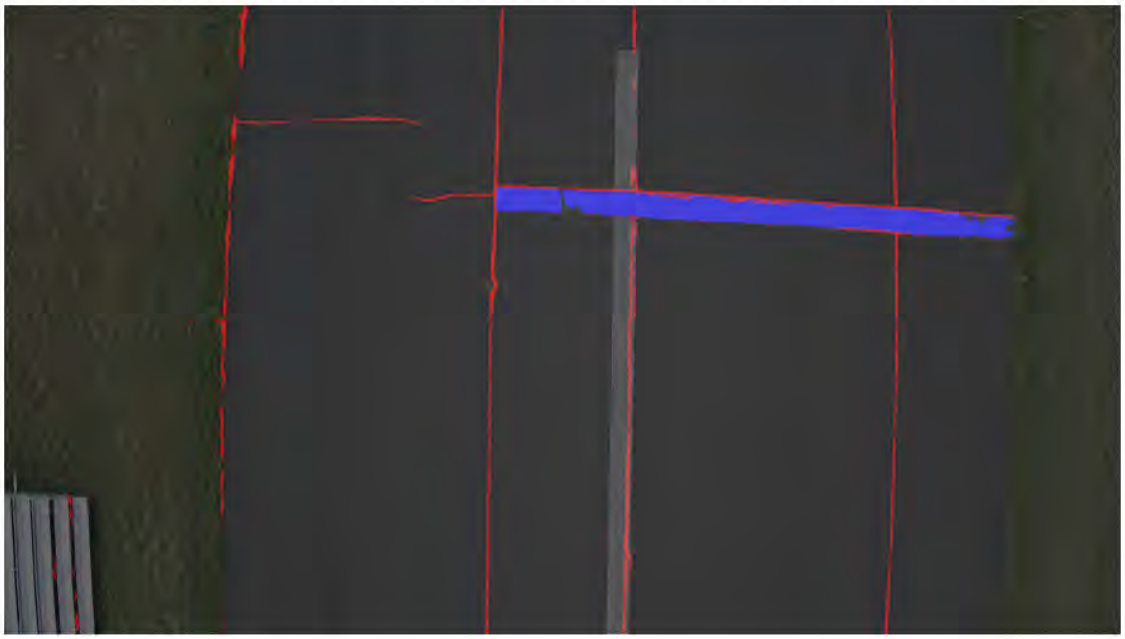
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 19
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.990
Detection IoU Score (Cracks): 0.246
Detection IoU Score (Patches): 0.900

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	150	163.3	-13.3
Crack Length (ft)	260	297.4	-37.4
PCI	50	52	-4.0%

Comments:
Patch Area Detection: There are some small patches of false negatives.
Crack Length Detection: Some cracks are missed by the algorithm in painted areas. Some small false positive detections appear on the edges where the pavement meets the grass.
Patch Area Calculation: Slight overestimation may be possible given we only see false negatives, but the auto measurement is still higher.
Crack Length Calculation: It is logical that the automated crack length is greater than the manual given the false positive detection along the runway's edge.
Distress Severity: Most crack severity levels are "high" in both auto and manual inspections. Most patch severity is "medium" in the manual and auto.
Overall Conclusion: Good agreement between PCI values is expected given similar measurements and distress severities used.

Figure A-19: Sample Unit 19



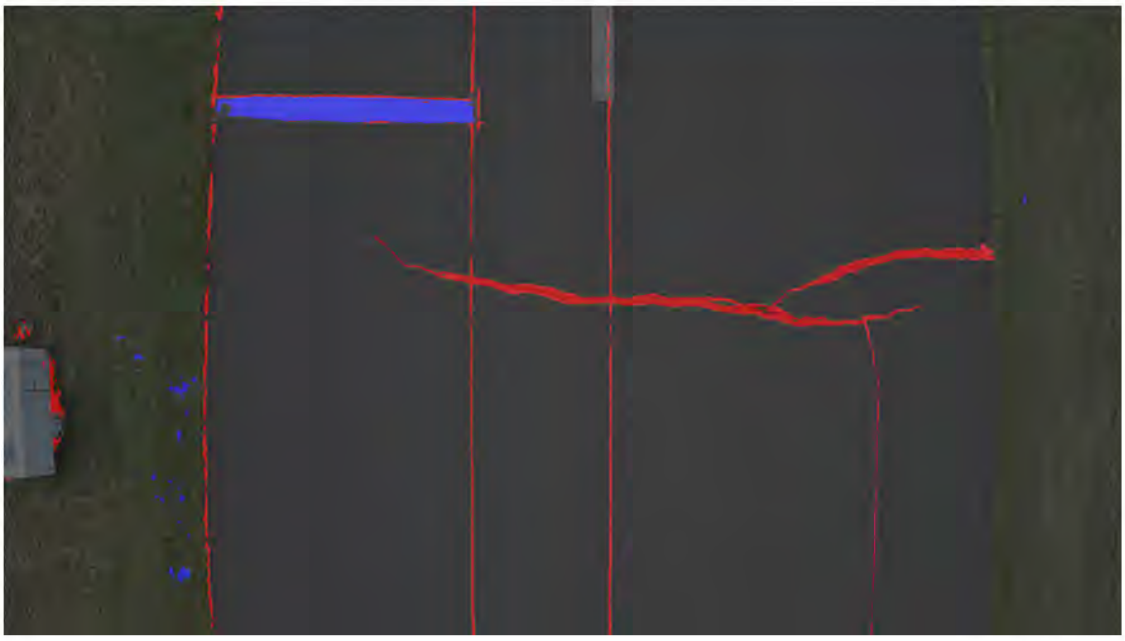
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 20
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.990
Detection IoU Score (Cracks): 0.222
Detection IoU Score (Patches): 0.897

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	100	96.7	3.3
Crack Length (ft)	285	315.1	-30.1
PCI	50	52	-4.0%

Comments:
Patch Area Detection: There are some small patches of false negatives.
Crack Length Detection: Some cracks are missed by the algorithm in painted areas and along patch edges. Many false positive detections appear on the edges where the pavement meets the grass and along the bleachers.
Patch Area Calculation: Slight overestimation may be possible given we only see false negatives and the measured values are almost the same.
Crack Length Calculation: It is logical that the automated crack length is greater than the manual given the false positive detection along the runway's edge.
Distress Severity: Most crack severity levels are "high" in both auto and manual inspections. Most patch severity is "medium" in the manual and auto.
Overall Conclusion: Good agreement between PCI values is expected given similar measurements and distress severities used.

Figure A-20: Sample Unit 20



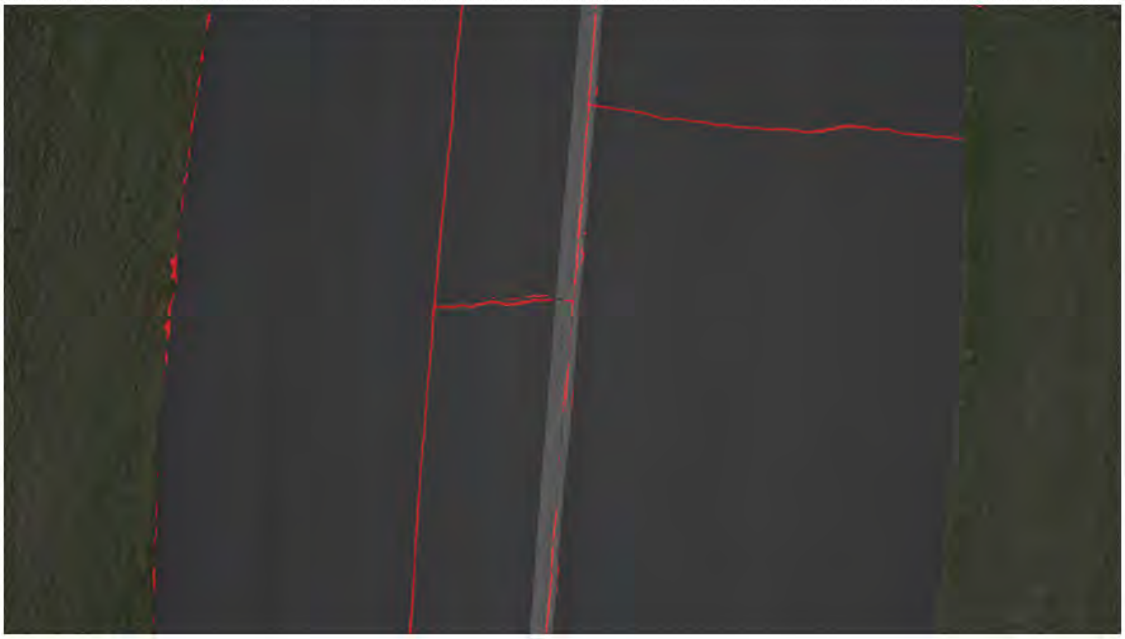
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 21
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.990
Detection IoU Score (Cracks): 0.512
Detection IoU Score (Patches): 0.801

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	50	52.4	-2.4
Crack Length (ft)	275	318.8	-43.8
PCI	52	52	0%

Comments:
Patch Area Detection: There are some small patches of false negatives. There are some small patches of false positives in the grass.
Crack Length Detection: Some cracks are missed by the algorithm in painted areas and along thin cracks. Many false positive detections appear on the edges where the pavement meets the grass and along the trailer.
Patch Area Calculation: False positives and false negatives may have balanced to yield an accurate calculation value.
Crack Length Calculation: It is logical that the automated crack length is greater than the manual given the false positive detection along the runway's edge.
Distress Severity: Most crack severity levels are "high" in both auto and manual inspections. Most patch severity is "medium" in the manual and auto.
Overall Conclusion: Good agreement between PCI values is expected given similar measurements and distress severities used.

Figure A-21: Sample Unit 21



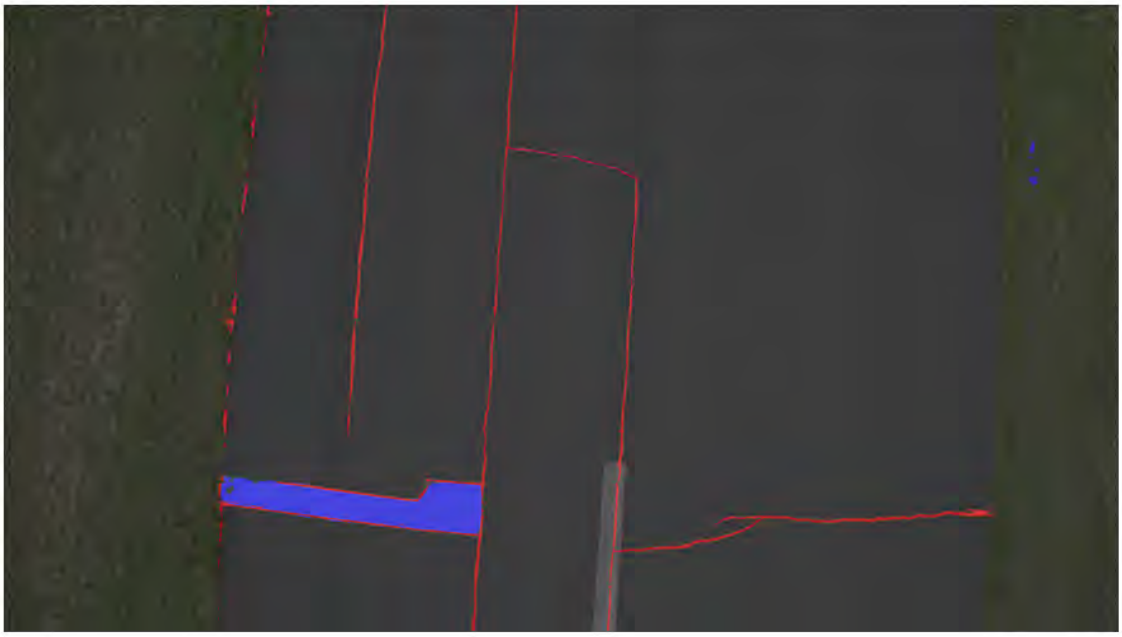
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 22
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.996
Detection IoU Score (Cracks): 0.283
Detection IoU Score (Patches): NA

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	0	0	0
Crack Length (ft)	180	182.8	-2.8
PCI	64	64	0%

Comments:
Patch Area Detection: No patches are present, and none were detected.
Crack Length Detection: Some small false positive detections appear on the edges where the pavement meets the grass. Some false negatives are in painted areas.
Patch Area Calculation: N/A
Crack Length Calculation: It is logical that the automated crack length is the same as the manual given the mix of false positives and false negatives.
Distress Severity: Most crack severity levels are "high" in both auto and manual inspections.
Overall Conclusion: Good agreement between PCI values is expected given similar measurements and distress severities used.

Figure A-22: Sample Unit 22



Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 23
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.994
Detection IoU Score (Cracks): 0.352
Detection IoU Score (Patches): 0.925

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	85	70	15
Crack Length (ft)	180	260.2	-80.2
PCI	57	55	3.5%

Comments:
Patch Area Detection: Detection performance is very good.
Crack Length Detection: Some small false positive detections appear on the edges where the pavement meets the grass.
Patch Area Calculation: Because detection is good, the difference in measurement may be error in the manual measurement or auto underestimation.
Crack Length Calculation: It is logical that the automated crack length is greater than the manual given the false positive detection along the runway's edge.
Distress Severity: Most crack severity levels are "high" in both auto and manual inspections. Most patch severity is "medium" in the manual and auto.
Overall Conclusion: Good agreement between PCI values is expected given similar measurements and distress severities used.

Figure A-23: Sample Unit 23

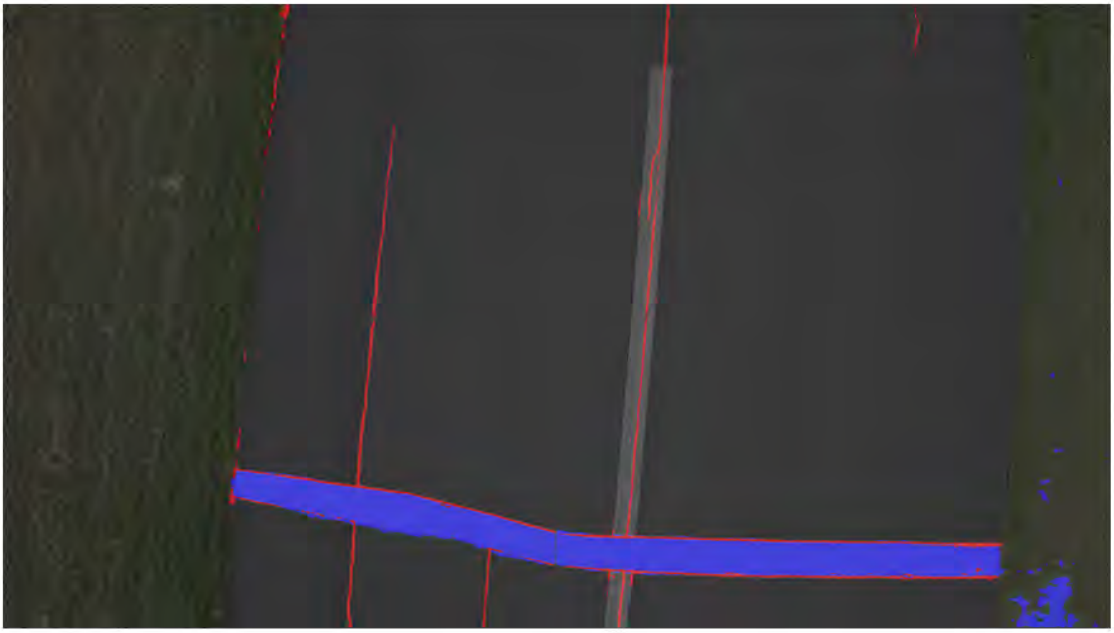
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 24
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.981
Detection IoU Score (Cracks): 0.330
Detection IoU Score (Patches): 0

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	0	96.6	-96.6
Crack Length (ft)	115	114.8	0.2
PCI	68	68	0%

Comments:
Patch Area Detection: No patches are present, but there is a large patch of false positive detection in the grass.
Crack Length Detection: Some small false positive detections appear on the edges where the pavement meets the grass. Some false negatives are in painted areas.
Patch Area Calculation: This is expected given there are only false positives.
Crack Length Calculation: It is logical that the automated crack length is the same as the manual given the small mix of false positives and false negatives.
Distress Severity: Most crack severity levels are "high" in both auto and manual inspections.
Overall Conclusion: Good agreement between PCI values is expected given similar measurements and distress severities used.

Figure A-24: Sample Unit 24



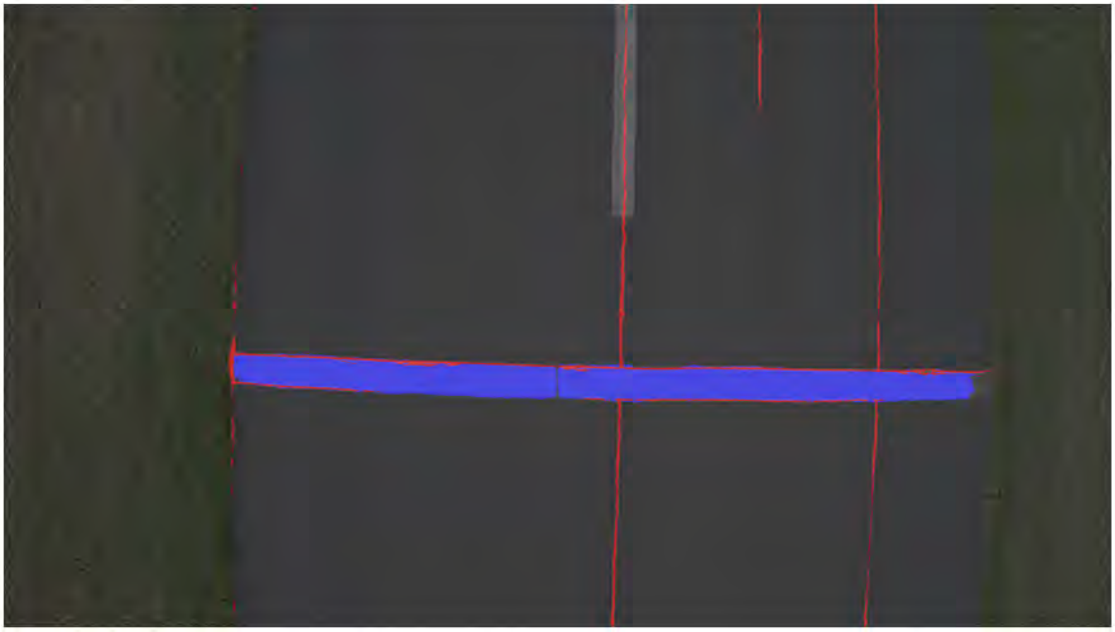
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 25
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.989
Detection IoU Score (Cracks): 0.270
Detection IoU Score (Patches): 0.866

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	225	229.5	-4.5
Crack Length (ft)	300	270.6	29.4
PCI	52	55	-5.8%

Comments:
Patch Area Detection: There are false positive detections in the grass.
Crack Length Detection: Some cracks are missed on along the edge of patches. Some small false positive detections appear on the edges where the pavement meets the grass.
Patch Area Calculation: The calculation likely underestimated measurements given there are false positives, but the values are similar.
Crack Length Calculation: It is logical that the automated crack length is less than the manual given the false negatives along the patch edges.
Distress Severity: Most crack severity levels are "high" in both auto and manual inspections. Most patch severity is "medium" in the manual and auto method.
Overall Conclusion: A better PCI score is expected in the auto method given a smaller value for total crack length.

Figure A-25: Sample Unit 25



Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 26
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.992
Detection IoU Score (Cracks): 0.296
Detection IoU Score (Patches): 0.916

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	150	189.4	-39.4
Crack Length (ft)	260	250.9	9.1
PCI	50	70	-40%

Comments:
Patch Area Detection: There are only a few small instances of false negatives.
Crack Length Detection: Only a few false negatives and false positives are present.
Patch Area Calculation: The calculation for area is likely overestimating because the auto measurement is larger, but there are only false negative detections.
Crack Length Calculation: The agreement between auto and manual measurements is expected given good detection.
Distress Severity: Half of the cracks are labeled "high" severity in the manual, but all are marked as "medium" in the auto. Most of the patching is marked "medium" in the auto and manual.
Overall Conclusion: Disparity between manual and auto is likely due to underestimated crack severity by the auto method. This clearly has a large effect on the PCI results.

Figure A-26: Sample Unit 26

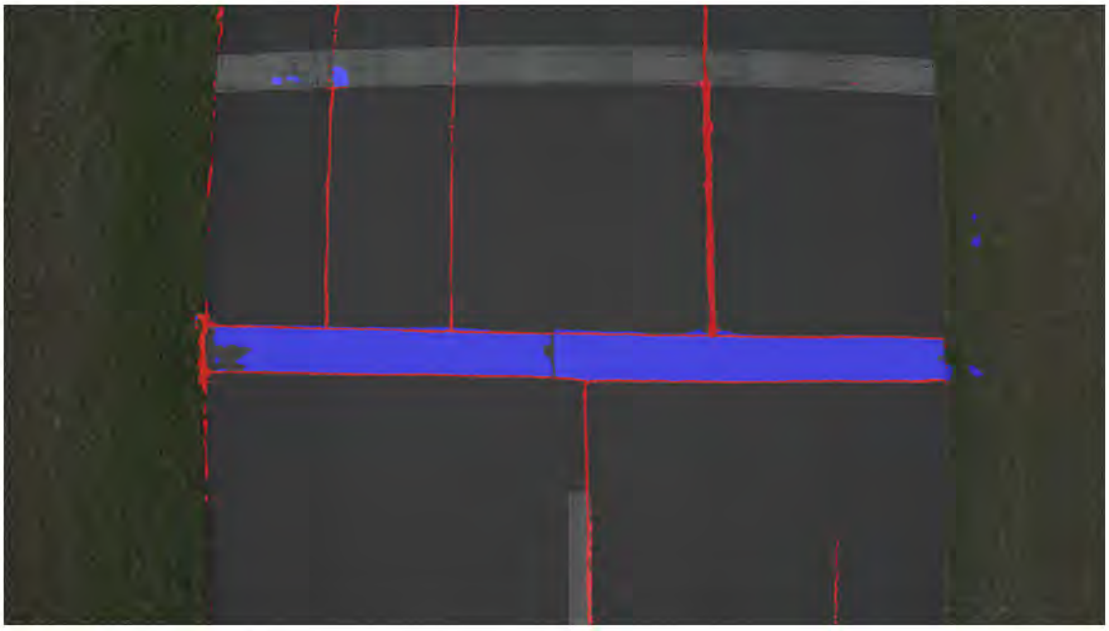
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 27
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.996
Detection IoU Score (Cracks): 0.327
Detection IoU Score (Patches): NA

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	0	0	0
Crack Length (ft)	195	200	-5.0
PCI	59	78	-32.2%

Comments:
Patch Area Detection: No patches are present, and none were detected.
Crack Length Detection: Some small false positive detections appear on the edges where the pavement meets the grass. Some false negatives are in painted areas.
Patch Area Calculation: N/A
Crack Length Calculation: It is logical that the automated crack length is the same as the manual given the mix of false positives and false negatives.
Distress Severity: The cracks are labeled "high" severity in the manual, but all are marked as "medium" in the auto. Most of the patching is marked "medium" in the auto and manual.
Overall Conclusion: A higher auto PCI is likely due to underestimated crack severity by the auto method. This clearly has a large effect on the PCI results.

Figure A-27: Sample Unit 27



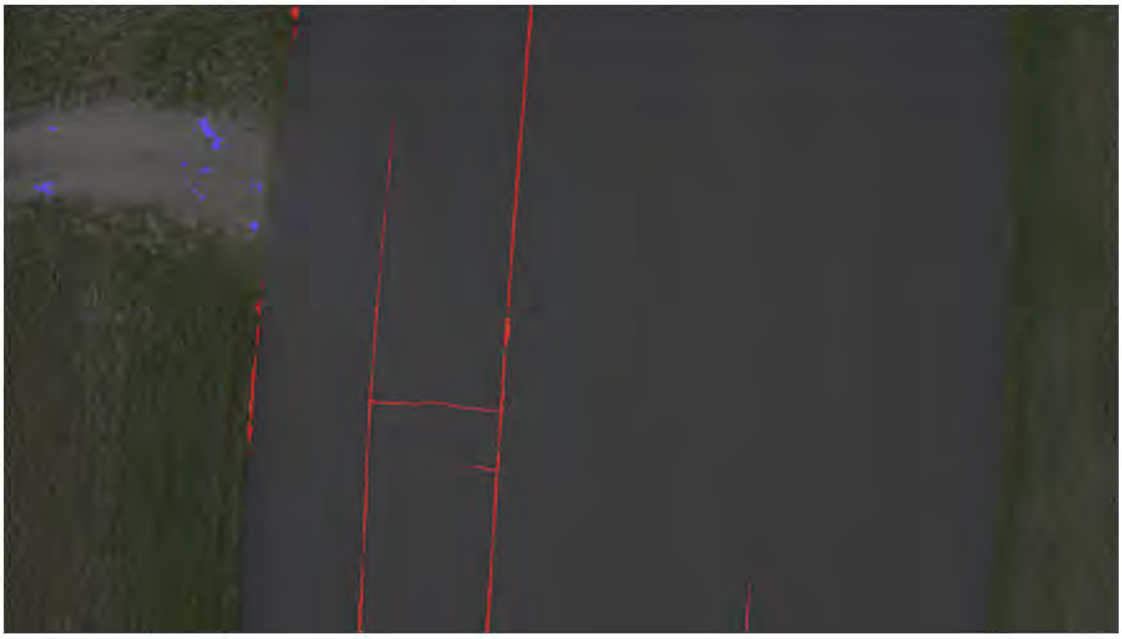
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 28
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.988
Detection IoU Score (Cracks): 0.388
Detection IoU Score (Patches): 0.897

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	300	274.1	25.9
Crack Length (ft)	300	305	-5.0
PCI	52	52	0%

Comments:
Patch Area Detection: There are only a few small instances of false negatives. There are some false positives shown in painted areas too.
Crack Length Detection: Only a few false negatives and false positives are present.
Patch Area Calculation: The lower auto measurement is likely accurate given the dominance of false negatives.
Crack Length Calculation: The agreement between auto and manual measurements is expected given both false positive and false negatives in detection..
Distress Severity: Cracks are labeled as "high" severity in both manual and auto methods. Patches are labeled "low" in the manual, but "medium in the auto.
Overall Conclusion: Fair agreement between distress measurements and severities explains the agreement between PCI values.

Figure A-28: Sample Unit 28



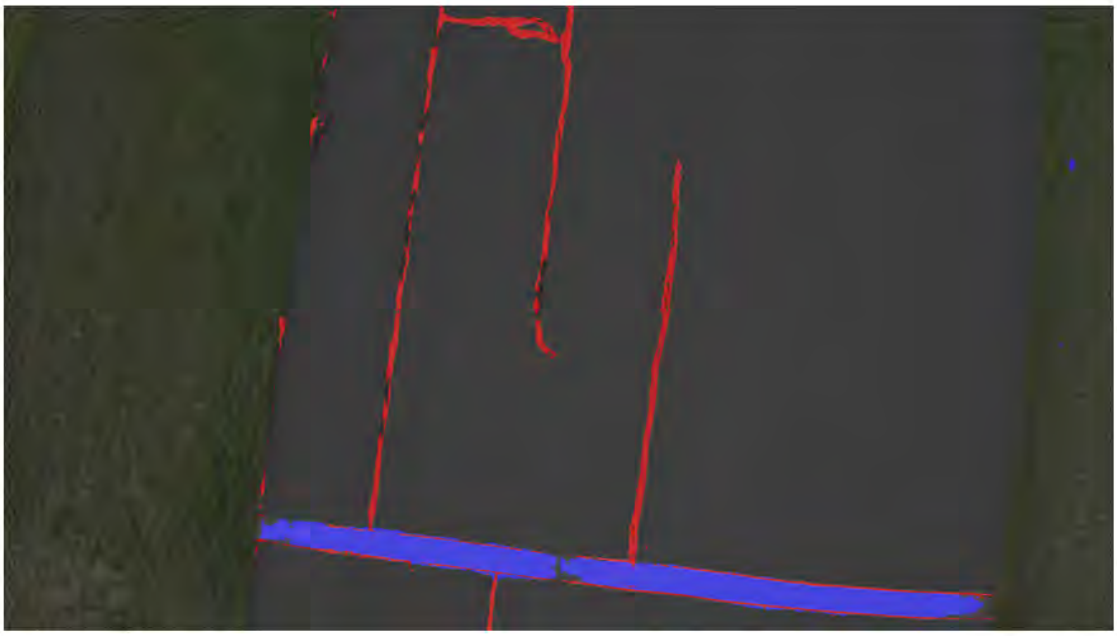
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 29
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.996
Detection IoU Score (Cracks): 0.360
Detection IoU Score (Patches): 0

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	0	5.2	-5.2
Crack Length (ft)	150	136.4	13.6
PCI	62	62	0%

Comments:
Patch Area Detection: No patches are present, but there is a small patch of false positive detection in the grass.
Crack Length Detection: Some small false positive detections appear on the edges where the pavement meets the grass. Some false negatives are in areas where cracks are very thin.
Patch Area Calculation: This is expected given there are only false positives.
Crack Length Calculation: It is logical that the automated crack length is the same as the manual given the small mix of false positives and false negatives.
Distress Severity: Most crack severity levels are "high" in both auto and manual inspections.
Overall Conclusion: Good agreement between PCI values is expected given similar measurements and distress severities used.

Figure A-29: Sample Unit 29



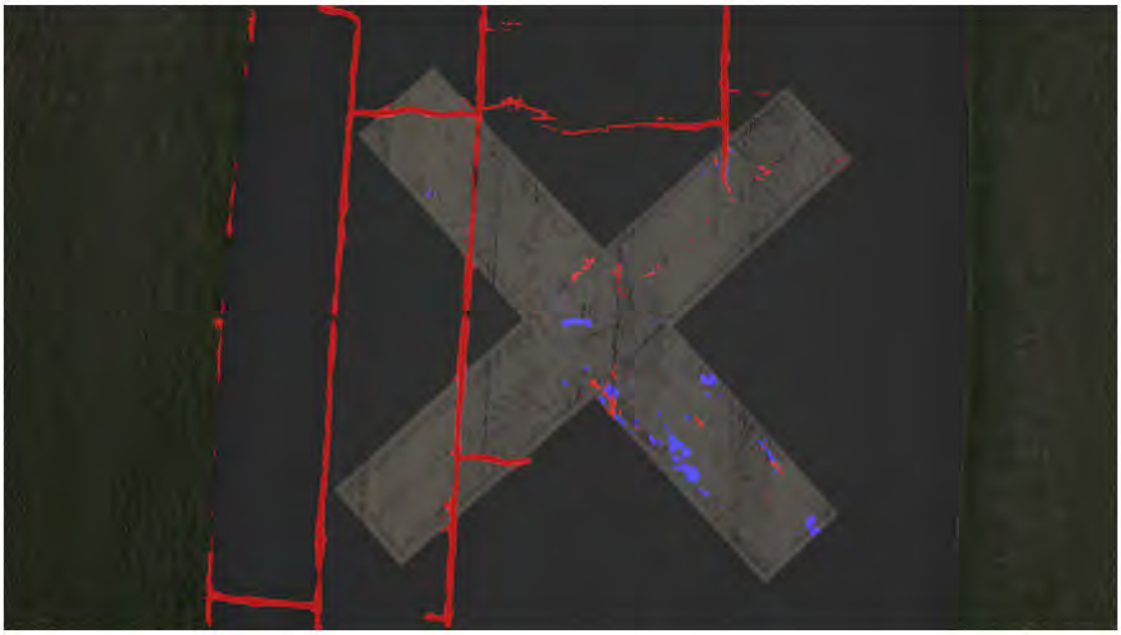
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 30
Sample Unit Area: 5000 ft²

Detection Accuracy: 0.990
Detection IoU Score (Cracks): 0.605
Detection IoU Score (Patches): 0.869

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	150	152.4	-2.4
Crack Length (ft)	310	316	-6.0
PCI	47	52	-10.6%

Comments:
Patch Area Detection: There are only a few small instances of false negatives.
Crack Length Detection: Only a few false negatives and false positives are present.
Patch Area Calculation: The calculation for area is likely good given good detection and similar measurement values.
Crack Length Calculation: The agreement between auto and manual measurements is expected given good detection.
Distress Severity: Most cracks in manual and auto methods are "high" severity. Patching in manual and auto methods is "medium" severity.
Overall Conclusion: Disparity between manual and auto seems counterintuitive given similar measurements and distress severity. The reason for this is not evident.

Figure A-30: Sample Unit 30



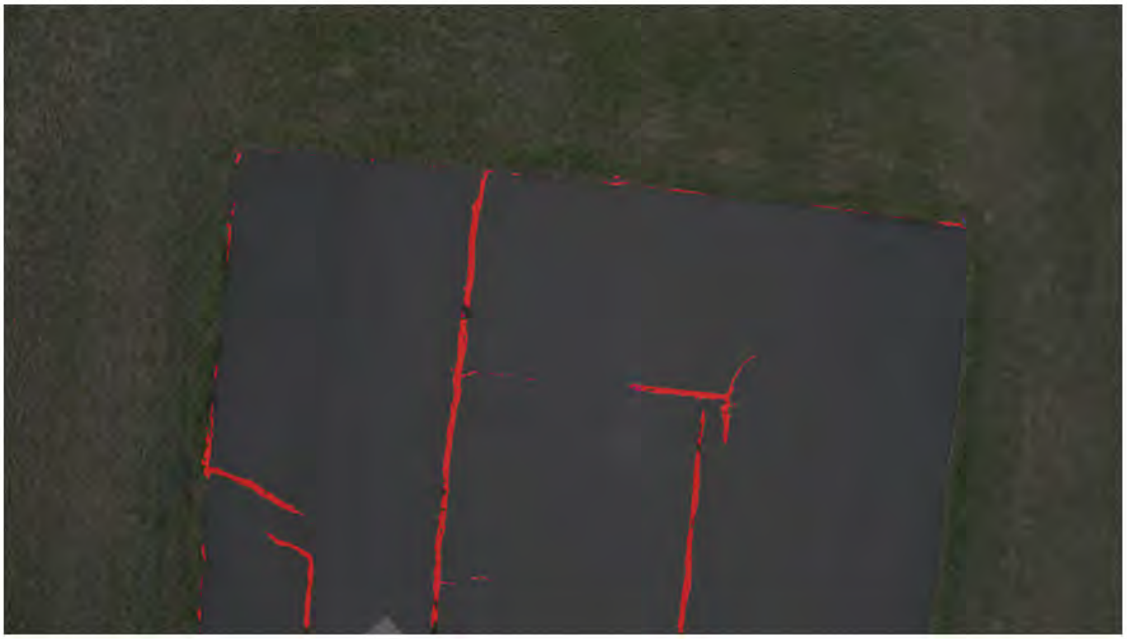
Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 31
Sample Unit Area: 4620 ft²

Detection Accuracy: 0.988
Detection IoU Score (Cracks): 0.594
Detection IoU Score (Patches): 0

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	0	15.6	-15.6
Crack Length (ft)	215	284	-69
PCI	59	52	11.9%

Comments:
Patch Area Detection: No patches are present, but there is a small patch of false positive detection in the painted area.
Crack Length Detection: Some small false positive detections appear on the edges where the pavement meets the grass and in painted areas.
Patch Area Calculation: This is expected given there are only false positives.
Crack Length Calculation: It is logical that the automated crack length is greater than the manual given the false positive detection.
Distress Severity: Most crack severity levels are "high" in both auto and manual inspections.
Overall Conclusion: The presence of false positive patch detection by the CNN may be the reason behind the auto method having a lower PCI value than the manual.

Figure A-31: Sample Unit 31



Airfield: Aardvark
Branch: Runway
Section: 1
Sample Unit #: 32
Sample Unit Area: 3000 ft²

Detection Accuracy: 0.997
Detection IoU Score (Cracks): 0.678
Detection IoU Score (Patches): 0

	Manual	Automated	Error (Manual – Auto)
Patch Area (ft²)	0	0.1	-0.1
Crack Length (ft)	100	173.1	-73.1
PCI	76	67	11.8%

Comments:
Patch Area Detection: No patches are present.
Crack Length Detection: Some false positive detections appear on the edges where the pavement meets the grass.
Patch Area Calculation: Patch are calculated is insignificant.
Crack Length Calculation: It is logical that the automated crack length is greater than the manual given the number of false positive detections along the pavement edge.
Distress Severity: Half of the crack severity measurements are "medium" and half are "high" in the manual method, but only "high in the automated.
Overall Conclusion: Greater crack length and severity in the auto method would drive the PCI score to be lower than the manual.

Figure A-32: Sample Unit 32

Appendix B

Image Labeling with Matlab

Training and evaluating a CNN for segmentation requires image data and ground truth labels of these images. During training and testing every image fed into the network has a corresponding "mask" (or labeled image) the network receives as an input. Instead of containing RGB light intensity information like the original image, the image label contains the categorical label associated with each pixel in a particular image. Matlab™ has an app available in the Computer Vision Toolbox™ to help generate these labels (Figure B-1). This manual will walk through the basics of using the image labeler app. More information can be found in the official Matlab documentation under "Image Labeler".

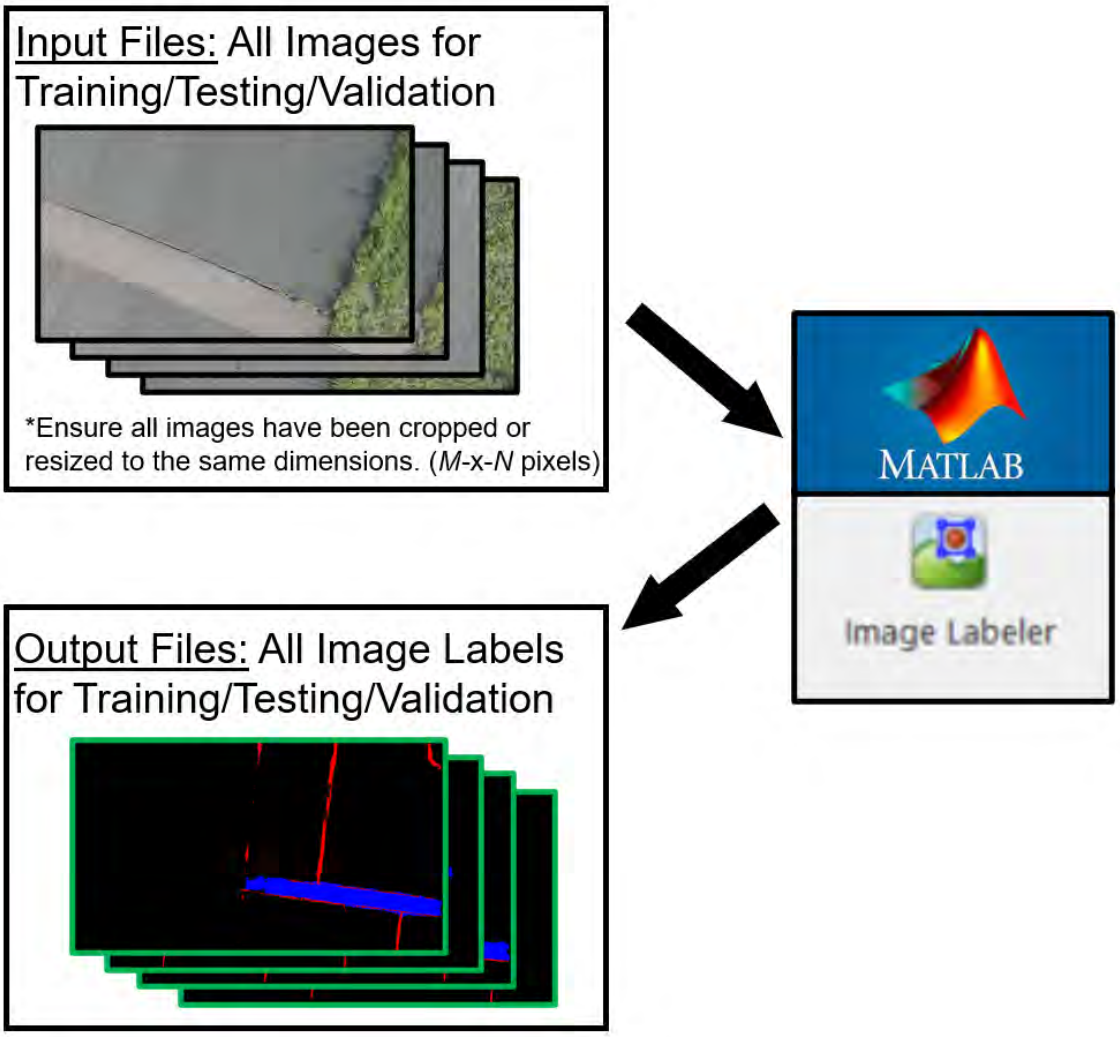


Figure B-1: The image labeler application receives the image data as inputs and provides a graphical user interface that assists in generating ground truth images that can be used for training a CNN.

Step 1

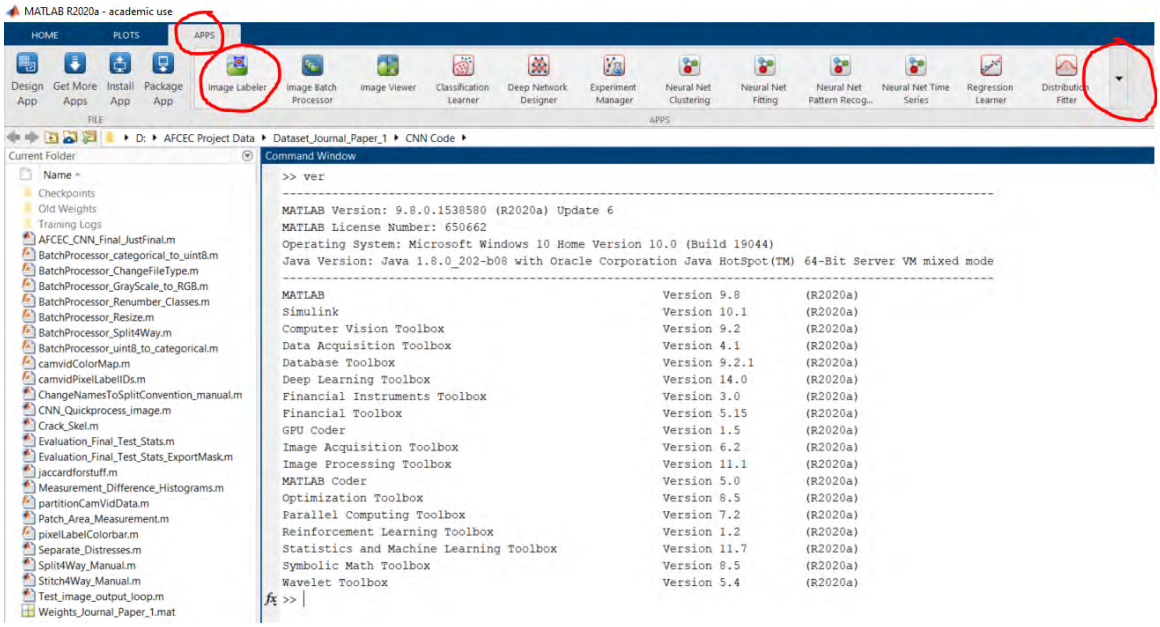


Figure B-2: Start by opening up an instance of Matlab. Click on the "Apps" tab, and then click on the drop-down menu of apps and open "Image Labeler" as shown below. In this picture "Image Labeler" is in the favorites bar, so it shows up in the window. If you cannot find the application, type "ver" in the command line to check what Matlab toolboxes are installed. Although every Matlab package (or "toolbox") listed above may not be essential, the list in the image above has been tested to run this project without error.

Step 2

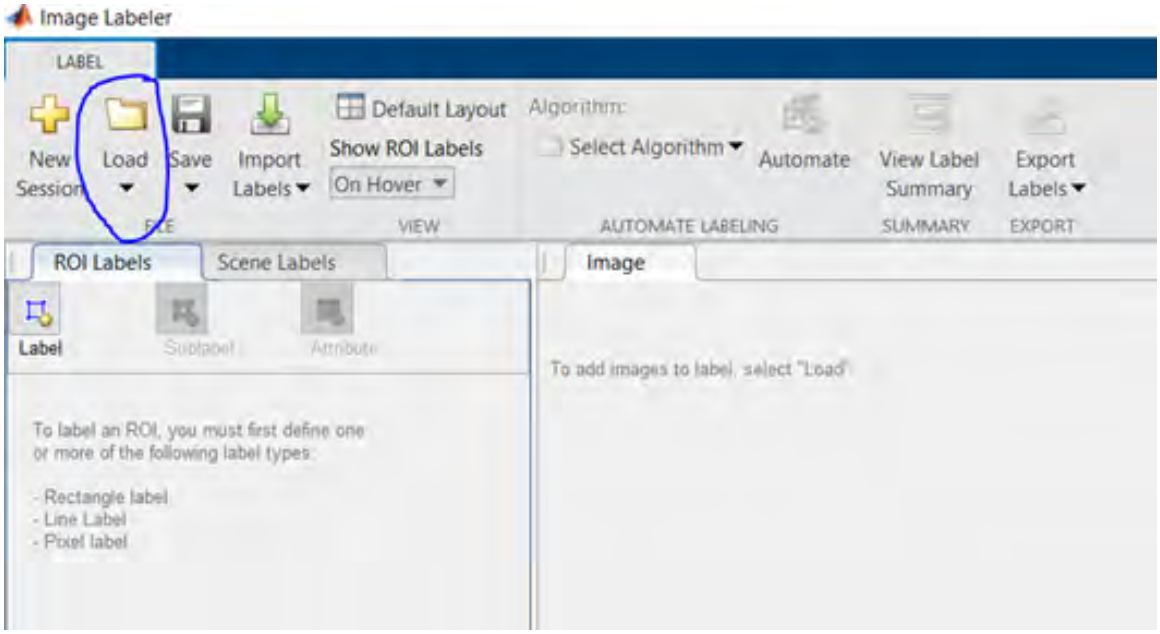


Figure B-3: Once you have the "Image Labeler" app opened you will need to load the images you are looking to label or load an existing session. If you have an existing session simply click on "Load", then click "Session" and select the session file you have and you can begin working. If you are starting a new labeling session with a new data set click on "Load" and then click "Add images from folder". A pop up file selector will appear, select all files you wish to include in this session. The easiest way to do this is to click once on the first image to highlight it, then scroll to the last image and SHIFT+RIGHT CLICK the last image, highlighting all images in the folder. Now all the images should be loaded into your session.

Step 3

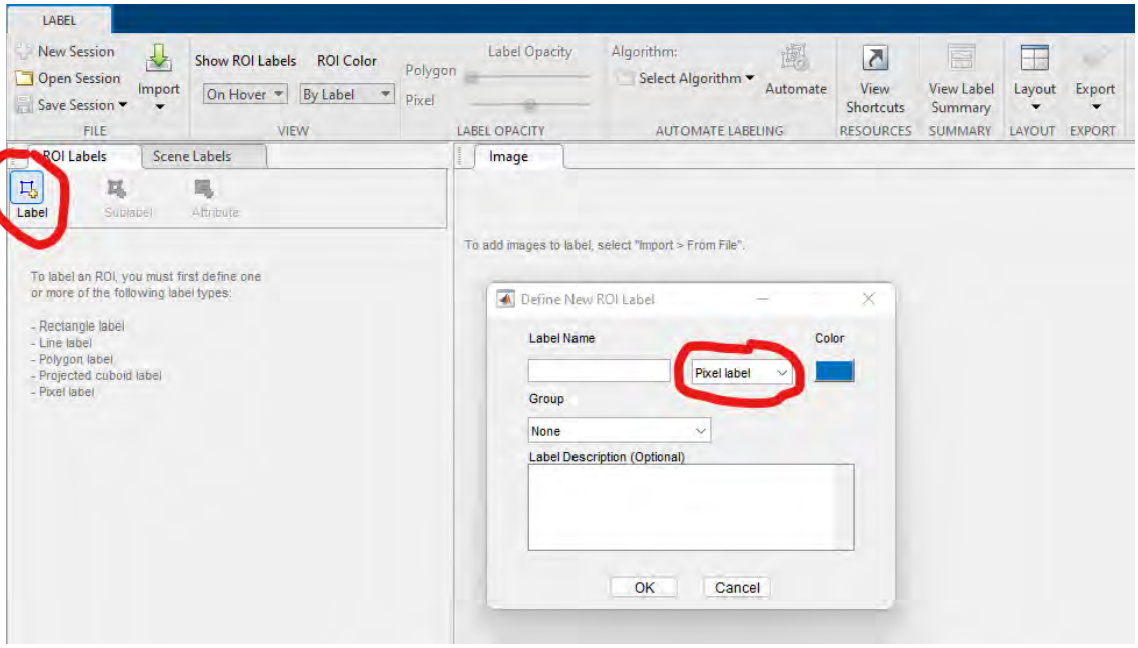


Figure B-4: If you have an existing label.mat file with your classification labels, go to the "Load" drop-down menu again, click on "Label Definitions" and select a label file to use, then proceed to the next step. If you don't have a label file yet and need to create new labels, hit the "Label" button in the "ROI Labels" tab on the left side of the screen. A pop up menu will populate. In it select "Pixel Label" from the drop-down menu circled bellow.

Step 4

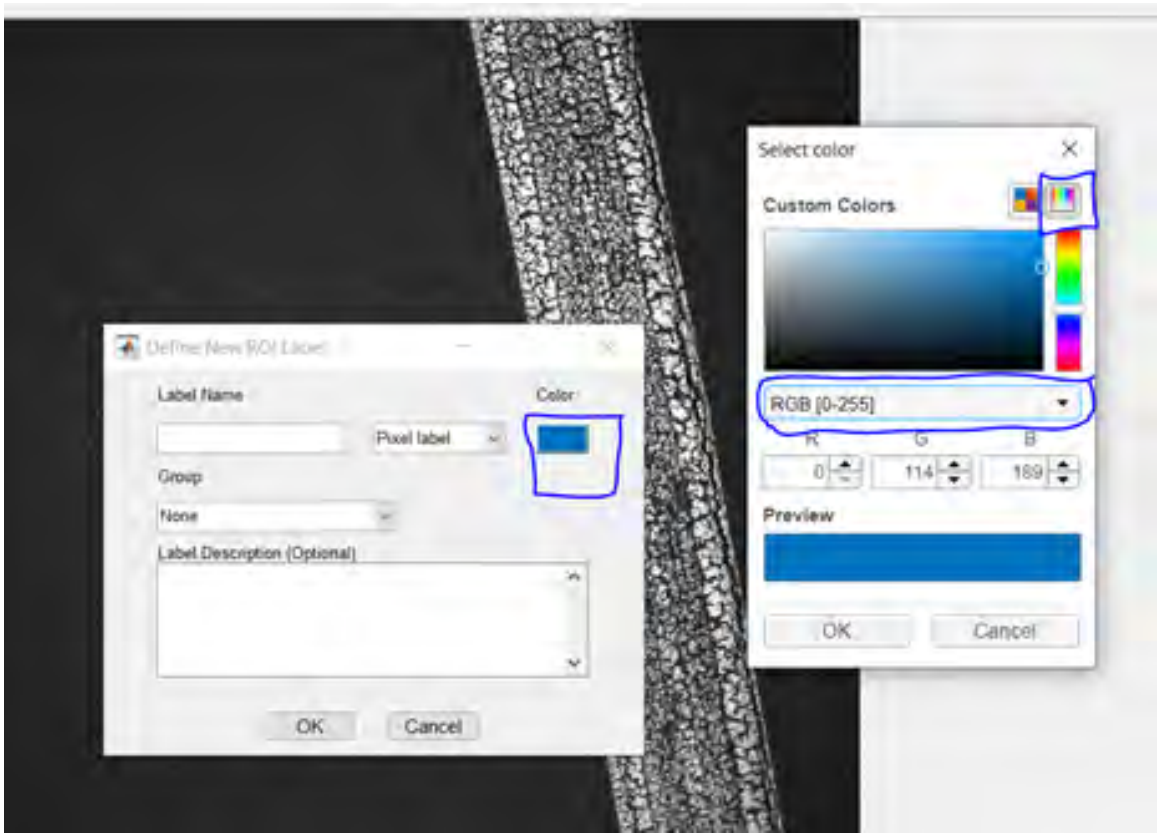


Figure B-5: Click on the box Colored box below the label "Color" and select the right color pallet button on the new popup menu and make sure the drop-down menu is "RGB [0-255]". Once you have labels created or loaded in the colors you want for each of your categories, save your "session" by going to the "Save" tab. Now, whenever you open up the app again, you can just load this "session" and pick up exactly where you left off. If you have an existing session it is important you do not alter the input and output image file names and locations on your computer, otherwise you will receive an error when trying to re-open your session and you may have to start over.

Step 5

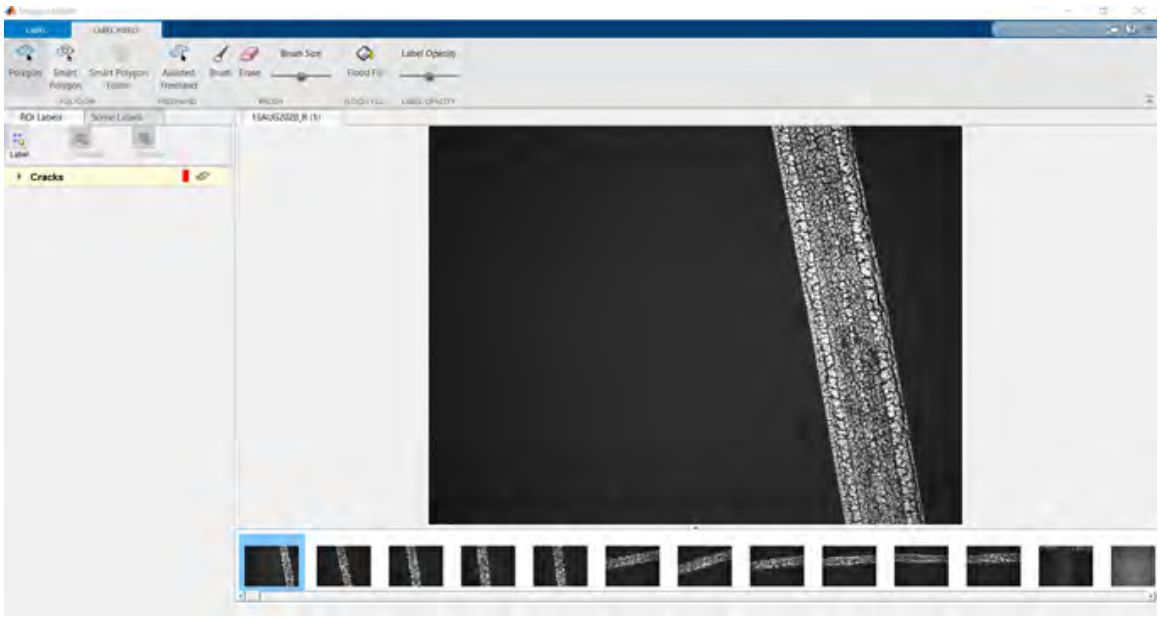


Figure B-6: If you have labels loaded or created, your screen will now have a new tab at the very top called "Label Pixels" so you can begin labeling your images. You can use Polygons, Smart Polygons, Brushes, and Different algorithms to try to quickly label each pixel in your image of the class you want. Play around with it and see what is fast and accurate for what you are trying to do. For cracks I have found the only reliable way is to use the paintbrush and do each pixel by hand. Do not forget to save your session often. Be as accurate as you can if your access to data is limited. I find each image usually takes anywhere from a few minutes to a half hour to do well depending on how much is there in the image to label. Try adjusting the size of your brush if you need to change your precision.

Step 6

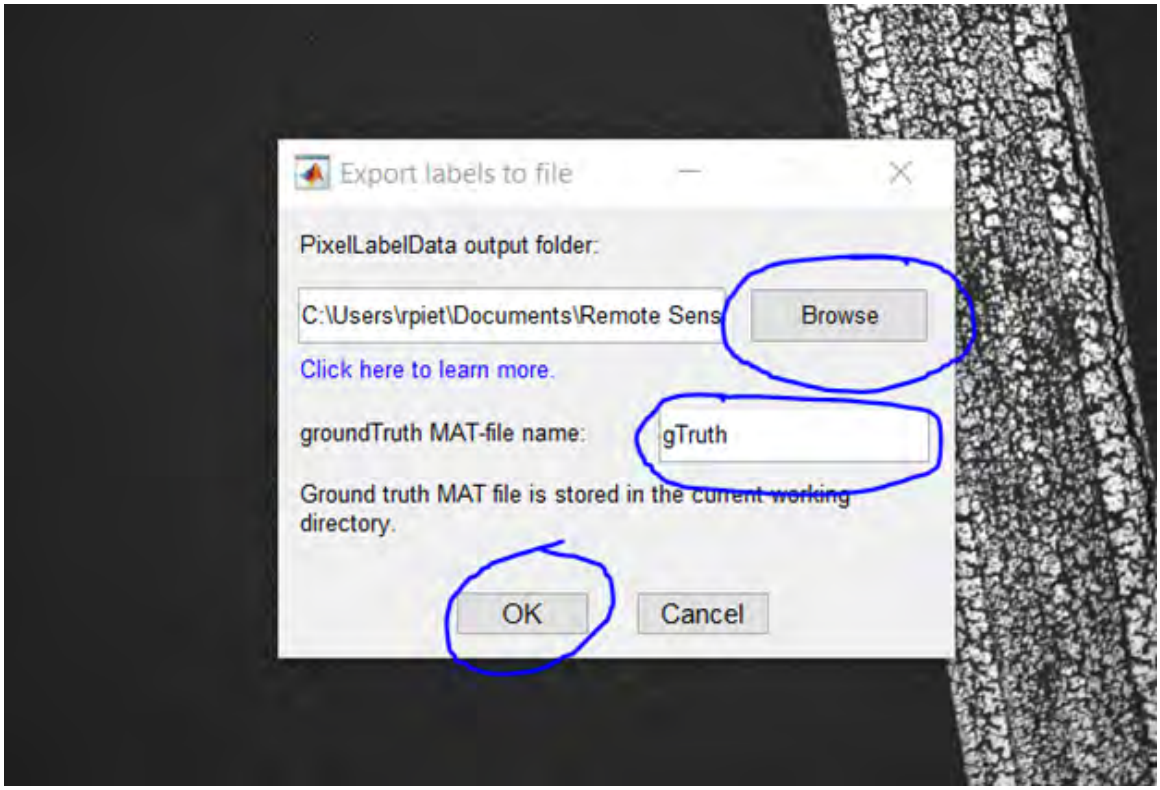


Figure B-7: When you are done labeling your entire set, make a new folder in your computer to store these labeled images, also called masks or ground truth data. Click on "Export Labels" and then "To file". Hit "Browse" and select the folder you just made to deposit these labeled images into. Change the name of the .MAT file that will be created to go along with these data if you need to. Hit "OK" and the app will save all ground truth images into the file you specified and will save the "gTruth.mat" file into your current directory opened in MATLAB.

Appendix C

Matlab Code

This appendix contains four code files. The first three blocks of code are required to train and test the neural networks used in this research. The last block of code is used to extract physical measurement estimations from the neural networks outputs. The commenting explains how the code works and what most of the functions do. When a comment begins with "specify" this generally indicates the subsequent line is an input that can be changed to fit the individual needs of the project. Some of the code is robust and can easily be applied elsewhere, other parts are hyper-specialized to this project's data and application. The comments provided, as well as the official Matlab documentation for the functions used, should provide most of the answers; however, if anything is unclear please email me personally at rpieters@mit.edu for assistance. The Matlab version and toolboxes used to successfully run this code are listed in Figure C-1. Other combinations of packages and versions may be compatible but are not tested.

```
>> ver
-----------------------------------------------------------------------------------------------------
MATLAB Version: 9.8.0.1538580 (R2020a) Update 6
MATLAB License Number: 650662
Operating System: Microsoft Windows 10 Home Version 10.0 (Build 19044)
Java Version: Java 1.8.0_202-b08 with Oracle Corporation Java HotSpot(TM) 64-Bit Server VM mixed mode
-----------------------------------------------------------------------------------------------------
MATLAB                                                Version 9.8         (R2020a)
Simulink                                              Version 10.1        (R2020a)
Computer Vision Toolbox                               Version 9.2         (R2020a)
Data Acquisition Toolbox                              Version 4.1         (R2020a)
Database Toolbox                                      Version 9.2.1       (R2020a)
Deep Learning Toolbox                                 Version 14.0        (R2020a)
Financial Instruments Toolbox                         Version 3.0         (R2020a)
Financial Toolbox                                     Version 5.15        (R2020a)
GPU Coder                                             Version 1.5         (R2020a)
Image Acquisition Toolbox                             Version 6.2         (R2020a)
Image Processing Toolbox                              Version 11.1        (R2020a)
MATLAB Coder                                          Version 5.0         (R2020a)
Optimization Toolbox                                  Version 8.5         (R2020a)
Parallel Computing Toolbox                            Version 7.2         (R2020a)
Reinforcement Learning Toolbox                        Version 1.2         (R2020a)
Statistics and Machine Learning Toolbox               Version 11.7        (R2020a)
Symbolic Math Toolbox                                 Version 8.5         (R2020a)
Wavelet Toolbox                                       Version 5.4         (R2020a)
```

Figure C-1: The code in this section has been tested and verified using this Matlab version and these toolboxes.

C.1 Matlab Code: CNN Main Code

This block of code takes image and label data inputs and trains a CNN to perform semantic segmentation. To run it requires that all input images already share (or be either cropped or resized to share) the same matrix dimensions (MxN pixels). It also requires that for each image, its corresponding ground truth image label shares the same file name. This code also requires the function defined in Section C.2.

```
%% Import input data and initialize
%Specify the file folder path containing your image data.
imgdirectory = fullfile(...
    'D:\AFCEC Project Data\Dataset_Journal_Paper_1\Images');

%Specify the file folder path containing your label data. Be sure the
%labels share the same file name as the image the correspond to.
labeldirectory = fullfile(...
    'D:\AFCEC Project Data\Dataset_Journal_Paper_1\Labels');

%Specify the minibatch size you wish to run your code with. This will be
```

```
%limited by the computer hardware you have available, but bigger is better
batchsize=4;

%Specify the maximum number of epochs you wish to conduct during training
numepochs=5;

%Specify the file path you wish to save checkpoint files during training
checkpointpath = ...
    'D:\AFCEC Project Data\Dataset_Journal_Paper_1\CNN Code\Checkpoints';

%Specify the network image size. This is typically the same as the traing
%image sizes.
imageSize = [1080 1920 3];

%This function creates an image datastore object in your matlab workspace
imgdatastore = imageDatastore(imgdirectory);

%Specify the classes or categories you are detecting
classes = [
           "Background"
           "Cracks"
           "Patches"
          ];

%Specify what numeric values are in the pixels of your ground truth image
%labels for each of the corresponding categories. Notice that the location
%of the number must correspond to the "classes" matrix above. So the first
%entry means that any pixels in the ground truth image label that match the
%value [000,000,000] will be associated with the category known as
%"background". If this is unknown you can use the "Image Viewer" app in
%matlab to look at what the pixel values are in an image. This can be
%changed to add new categories for detection if desired.
labelIDs = { ...
    [000, 000, 000] % "Background"
    [001, 001, 001] % "Cracks"
    [002, 002, 002] % "Patches"
    };

%This function creates a pixel label datastore object in your matlab
%workspace
pixeldatastore = pixelLabelDatastore(labeldirectory,classes,labelIDs);


%This section of code will display an example of the picture and its ground
%truth
I = readimage(imgdatastore,2);
I = histeq(I);
C = readimage(pixeldatastore,2);

%The cmap defines what color each category will be displayed as. The Red
%channel is the first entry, the Green channel is the second, and the blue
%channel is the third. Similar to the "labelIDs" variable, the position of
%each category corresponds to the order defined in the "classes" matrix.
%The first entry here designates what color the "background" pixels will be
```

```
%colored as completely black, [000, 000, 000].
cmap = [
    000 000 000   % Background [R,G,B]
    001 000 000   % Crack
    000 000 001   % Patch
    ];
B = labeloverlay(I,C,'ColorMap',cmap);
imshow(B)
pixelLabelColorbar(cmap,classes);

tbl = countEachLabel(pixeldatastore);

%This produces a graph summarizing the representation of each
%classification category in the data provided.
frequency = tbl.PixelCount/sum(tbl.PixelCount);
bar(1:numel(classes),frequency)
xticks(1:numel(classes))
xticklabels(tbl.Name)
xtickangle(45)
ylabel('Frequency')

%This code divides the total input data randomly into independent training,
%validation, and test data sets. The "partitionData" function is a special
%function for this application. To avoid error please ensure
%"partitionData.m" is a file in your Matlab path and current folder so the
%code is able to locate and call this function. Please specify how you want
%your data partitioned between training, validation, and testing by
%changing this function.
[imdsTrain, imdsVal, imdsTest, pxdsTrain, pxdsVal, pxdsTest] =...
    partitionData(imgdatastore,pixeldatastore,labelIDs);
numTrainingImages = numel(imdsTrain.Files);
numValImages = numel(imdsVal.Files);
numTestingImages = numel(imdsTest.Files);

%% Create the CNN
% This determines the number of classes or detection categories from the
% matrix "classes"
numClasses = numel(classes);

%This creates DeepLabv3+. DeepLabv3+ requires a network backbone as the
%last input argument. In this case that is 'inceptionresnetv2'. Other
%options include 'resnet18', 'resnet50', 'mobilnetv2', and 'xception'. If
%the Deep Learning Toolbox Model for Inception-ResNet-v2 Network support
%package is not installed, then the function provides a link to the
%required support package in the Add-On Explorer. To install the support
%package, click the link, and then click 'Install'. Check that the
%installation is successful by typing 'inceptionresnetv2' at the command
%line. If the required support package is installed, then the function
%returns a 'DAGNetwork' object.
lgraph = deeplabv3plusLayers(imageSize, numClasses, 'inceptionresnetv2');

%This balances classes using class weighting, so classes without as much
%representation are still important for detection performance.
imageFreq = tbl.PixelCount ./ tbl.ImagePixelCount;
```

```
classWeights = median(imageFreq) ./ imageFreq
pxLayer = pixelClassificationLayer(...
    'Name','labels','Classes',tbl.Name,'ClassWeights',classWeights);
lgraph = replaceLayer(lgraph,"classification",pxLayer);

%This defines validation data.
pximdsVal = pixelLabelImageDatastore(imdsVal,pxdsVal);

%Specify the training options and hyperparameter values you wish to use for
%your network training. Please consult the documentation for
%'trainingOptions' to learn more.
options = trainingOptions('adam', ...
    'InitialLearnRate',1e-5, ...
    'ValidationData',pximdsVal,...
    'MaxEpochs',numepochs, ...
    'MiniBatchSize',batchsize, ...
    'Shuffle','every-epoch', ...
    'CheckpointPath', checkpointpath, ...
    'ExecutionEnvironment', 'gpu', ...
    'VerboseFrequency',20,...
    'Plots','training-progress');

%Specify the Data Augmentation you wish to perform on your training data.
%Please read more about what options are available in the documentation for
%'imageDataAugmenter'
augmenter = imageDataAugmenter(...
    'RandXTranslation',[-100 100],...
    'RandYTranslation',[-100 100],...
    'RandRotation',[-20 20]);

%% Start training
%This creates a new pixel image label datastore that has the data
%augmentation applied to it.
pximds = pixelLabelImageDatastore(imdsTrain,pxdsTrain, ...
    'DataAugmentation',augmenter);

%Specify 'true' if you wish to train a new weight file for the neural
%network. Specify 'false' if you wish to import an existing weight file. If
%importing an existing weight file, either make sure that file is in
%Matlab's current folder and working directory, or specify the full file
%name and location. The new weight file generated from training the network
%will be saved in the current directory under the string name specified
%next to the 'save' function. In this example that is
%'Weights_Journa_Paper_1'
doTraining = true;
if doTraining
    [Weights_Journal_Paper_1, info] = trainNetwork(pximds,lgraph,options);
    save Weights_Journal_Paper_1
    net = Weights_Journal_Paper_1;
    %If you decide to change the name of the output weight file, be sure to
    %change it in each place it says 'Weights_Journal_Paper_1'
else
    data = load('trainedresnet50_1.mat');
    net = data.trainedresnet50_1;
```

```
    %If you decide to import an existing weight file, be sure its name is
    %specified correctly in both places where it currently says
    %'trainedresnet50_1.mat
end

%This tests the resutling trained network on one image and displays the
%result
I = readimage(imdsTest,1);
C = semanticseg(I, net);
B = labeloverlay(I,C,'Colormap',cmap,'Transparency',0.4);
figure('Name','Prediction Over Image')
imshow(B)
pixelLabelColorbar(cmap, classes);

%This displays the expected (groundTruth) vs actual display for one example
%image
expectedResult = readimage(pxdsTest,1);
actual = uint8(C);
expected = uint8(expectedResult);
figure('Name','Prediction Over Ground Truth')
imshowpair(actual, expected)

%This calculates the IoU metric for the example
iou = jaccard(C,expectedResult);

%This evaluates the trained network using the test data
pxdsResults = semanticseg(imdsTest,net, ...
    'MiniBatchSize',4, ...
    'WriteLocation',tempdir, ...
    'Verbose',false);
table(classes,iou)
metrics = evaluateSemanticSegmentation(...
    pxdsResults,pxdsTest,'Verbose',false);
metrics.DataSetMetrics
metrics.ClassMetrics
```

C.2 Matlab Code: "partitionData.m" Function

This block of code randomly divides the input data for the code in Section C.1 into training, validation, and testing data.

```
function [imdsTrain, imdsVal, imdsTest, pxdsTrain, pxdsVal, pxdsTest] =...
    partitionData(imds,pxds,labelIDs)
% Partition data by randomly selecting 80% of the data for training.
% The rest is used for testing.

% Specify initial random state for example reproducibility.
rng(0);
```

```
numFiles = numel(imds.Files);
shuffledIndices = randperm(numFiles);

%Specify the percentage of the data used for training. In this case it is
% 0.8 of the images for training.
numTrain = round(0.8 * numFiles);
trainingIdx = shuffledIndices(1:numTrain);

%Specify the percentage of the data used for validation. In this case it is
% 0.1 of the images for training.
numVal = round(0.1 * numFiles);
valIdx = shuffledIndices(numTrain+1:numTrain+numVal);

%This uses the rest of the available data for testing.
testIdx = shuffledIndices(numTrain+numVal+1:end);

% Create image datastores for training and test.
trainingImages = imds.Files(trainingIdx);
valImages = imds.Files(valIdx);
testImages = imds.Files(testIdx);

imdsTrain = imageDatastore(trainingImages);
imdsVal = imageDatastore(valImages);
imdsTest = imageDatastore(testImages);

% Extract class and label IDs info.
classes = pxds.ClassNames;

% Create pixel label datastores for training and test.
trainingLabels = pxds.Files(trainingIdx);
valLabels = pxds.Files(valIdx);
testLabels = pxds.Files(testIdx);

pxdsTrain = pixelLabelDatastore(trainingLabels, classes, labelIDs);
pxdsVal = pixelLabelDatastore(valLabels, classes, labelIDs);
pxdsTest = pixelLabelDatastore(testLabels, classes, labelIDs);
end
```

C.3 Matlab Code: Evaluate on Test Data

This block of code takes new test data (images and their labels) as inputs and uses an already trained neural network weight file to evaluate the performance of the CNN.

```
%Specify the location of the images you wish to test
TestImageLocation = fullfile(...
    'D:\AFCEC Project Data\Dataset_Journal_Paper_1\Test\Images');

%Specify the location of the labels that correspond to the images you wish
```

```
%to test. Be sure the labels share the same file name as the image the
%correspond to.
TestLabelLocation = fullfile(...
    'D:\AFCEC Project Data\Dataset_Journal_Paper_1\Test\Labels');

%Specify the location you wish the new detection results to be exported to.
ExportLocation = fullfile(...
    'D:\AFCEC Project Data\Dataset_Journal_Paper_1\Results');

%This part of the code will prompt use input. Please select from your
%computer a the .mat file containing the CNN "weights" you wish to use to
%perform the detection. This file is generated and exported by running the
%"CNN_Main_Code.m" or can be downloaded from another source as long as it
%is compatible with this network.
[filename, path] = uigetfile;
CNN_File = fullfile(path, filename);
CNN_Weights = load(CNN_File);
[path,net_name,ext] = fileparts(CNN_File);
net = CNN_Weights.(net_name);

%Specify the classes or categories you are detecting. Please ensure the
%"classes", "labelIDs", and "cmap" matrices match the same ones used to
%generate the "weight" file selected that was produced by CNN_Main_Code.m
classes = [
           "Background"
           "Cracks"
           "Patches"
          ];

%Specify what numeric values are in the pixels of your ground truth image
%labels for each of the corresponding categories. Notice that the location
%of the number must correspond to the "classes" matrix above. So the first
%entry means that any pixels in the ground truth image label that match the
%value [000,000,000] will be associated with the category known as
%"background". If this is unknown you can use the "Image Viewer" app in
%matlab to look at what the pixel values are in an image. This can be
%changed to add new categories for detection if desired.
labelIDs = { ...
    [000, 000, 000] % "Background"
    [1, 1, 1] % "Cracks"
    [2, 2, 2] % "Patch"
    };

%The cmap defines what color each category will be displayed as. The Red
%channel is the first entry, the Green channel is the second, and the blue
%channel is the third. Similar to the "labelIDs" variable, the position of
%each category corresponds to the order defined in the "classes" matrix.
%The first entry here designates what color the "background" pixels will be
%colored as completely black, [000, 000, 000].
cmap = [
    000 000 000   % Background [R,G,B]
    1 000 000   % Crack
    000 000 1   % Patch
    ];
```

```
imdsTest_Images = imageDatastore(TestImageLocation);
imdsTest_Labels = pixelLabelDatastore(TestLabelLocation,classes,labelIDs);

%This evaluates the trained network using the weights file you selected
%from your computer
pxdsResults = semanticseg(imdsTest_Images,net, ...
    'MiniBatchSize',1, ...
    'WriteLocation',tempdir, ...
    'Verbose',false);
metrics = evaluateSemanticSegmentation(...
    pxdsResults,imdsTest_Labels,'Verbose',false);
metrics.DataSetMetrics
metrics.ClassMetrics
%This exports all labeled images
[NumImages,nothing] = size(imdsTest_Images.Files);
for i =1:NumImages
I = readimage(imdsTest_Images,i);
C = readimage(pxdsResults,i);
B = labeloverlay(I,C,'Colormap',cmap,'Transparency',0.4);
imwrite(B,fullfile(ExportLocation,sprintf('Final_%d.tif',i)))
end
```

C.4 Matlab Code: Physical Measurement Estimations

This block of code takes the detection outputs of the CNN and uses an estimation for drone altitude and the camera's field of view of to calculate the physical dimensions of the pavement damages in each sample unit.

```
%% Initialize
%This initializes matrices to write data into during the loops
crack_matrix = zeros(1,32);
patch_matrix = zeros(1,32);
crack_matrix_severity = {};
patch_matrix_severity = {};

%This loop runs through the 32 test image for each of the 32 sample units.
%If there are a different number of images, set "i" equal to a different
%range.
for i = 1:32
    %% Patch Import
    %Specify the file location path where your detection outputs are
    %located. This code is sensative to these images and requires the
    %patches are pure blue, [000,000,255], cracks are pure red,
    %[255,000,000], and the background is black, [000,000,000]
    folder = 'D:\AFCEC Project Data\Dataset_Journal_Paper_1\Results';

    %Specify the file name of each of your inputs. This code requires you
```

```
%name your files in a consistent manner changing only with number.
img_name_patch = sprintf('%d_patch.tif',i);
img_patch = imread(fullfile(folder,img_name_patch));
img_bw_patch = img_patch(:,:,3);

%% Patch Measurement
img_num_patch = double(img_bw_patch);
img_num_patch(img_num_patch>0) = 1;
patch_count = sum(sum(img_num_patch));

%Specify pixel dimension values using the imager FoV and
%estimated flight altitude.
pixel_area = .0262*.0271; %(inputs in ft, output in sqft)
patch_area = patch_count*pixel_area; %(sqft)
patch_matrix(1,i) = patch_area;

%All patches are assumed 'medium' severity
patch_matrix_severity{i} = 'medium';

%% Crack Full Import
%Specify the file location path where your detection outputs are
%located. This code is sensative to these images and requires the
%patches are pure blue, [000,000,255], cracks are pure red,
%[255,000,000], and the background is black, [000,000,000]
folder = 'D:\AFCEC Project Data\Dataset_Journal_Paper_1\Results';

%Specify the file name of each of your inputs. This code requires you
%name your files in a consistent manner changing only with number.
img_name_crack = sprintf('%d_crack.tif',i);
img_crackfull = imread(fullfile(folder,img_name_crack));
img_crackfull = double(img_crackfull);
img_crackfull = img_crackfull(:,:,1);
img_crackfull(img_crackfull>0) = 1;
crackfull_count = sum(sum(img_crackfull));
%% Crack Skel Import
%Batch process images with the 'bwskel' function to create traces of
%the crack distresses.
%Specify the file location path of the crack trace images
folder = ...
    'D:\AFCEC Project Data\Dataset_Journal_Paper_1\Results\CrackSkel';

%Specify the file name of each of your inputs. This code requires you
%name your files in a consistent manner changing only with number.
img_name_crack = sprintf('%d_crackskel.tif',i);
img_crack = imread(fullfile(folder,img_name_crack));
img_crack = double(img_crack);
img_crack(img_crack>0) = 1;
%% Crack Skel Measurement
crack_count = sum(sum(img_crack(:,:,1)));

%Specify pixel dimension values using the imager FoV and
%estimated flight altitude.
pixel_length = (.0262+.0271)/2; %(in ft)
crack_length = crack_count*pixel_length; %(ft)
```

```
    crack_matrix(1,i) = crack_length;
    %% Crack Severity Determination
    crackratio = crackfull_count/crack_count;
    if crackratio < 2
        crack_matrix_severity{i} = 'low';
    elseif (crackratio ≥ 2) && (crackratio < 7)
        crack_matrix_severity{i} = 'medium';
    elseif crackratio ≥ 7
        crack_matrix_severity{i} = 'high';
    end
end
%Displays the final measurement result for each sample unit
crack_matrix
crack_matrix_severity
patch_matrix
patch_matrix_severity
```

Bibliography

Sruthy Agnisarman, Snowil Lopes, Kapil Chalil Madathil, Kalyan Piratla, and Anand Gramopadhye. A survey of automation-enabled human-in-the-loop systems for infrastructure visual inspection. *Automation in Construction*, 97:52–76, 2019.

Dragos Andrei and Mehrnoosh Arabestani. Astm d6433 pavement condition index variability study.

Maziar Arjomandi, Shane Agostino, Matthew Mammone, Matthieu Nelson, and Tong Zhou. Classification of unmanned aerial vehicles. *Report for Mechanical Engineering class, University of Adelaide, Adelaide, Australia*, 2006.

ASTM. Astm d5340, standard test method for airport pavement condition index surveys. *ASTM International*, 2020.

ATAG. Economic growth, Sep 2020. URL `https://aviationbenefits.org/economic-growth`.

Christoph Balada, Markus Eisenbach, and Horst-Michael Gross. Evaluation of transfer learning for visual road condition assessment. In *International Conference on Artificial Neural Networks*, pages 540–551. Springer, 2021.

Yoshua Bengio, Ian Goodfellow, and Aaron Courville. *Deep learning*, volume 1. MIT press Cambridge, MA, USA, 2017.

Susan M Bogus, Jongchul Song, Raymond Waggerman, and Lary R Lenke. Rank correlation method for evaluating manual pavement distress data variability. *Journal of Infrastructure Systems*, 16(1):66–72, 2010.

Louisa Brooke-Holland. Unmanned aerial vehicles (drones): an introduction. *House of Commons Library: London, UK*, 2012.

Sylvie Chambon and Jean-Marc Moliard. Automatic road pavement assessment with image processing: review and comparison. *International Journal of Geophysics*, 2011, 2011.

Susanne Chan, Pavement Design Engineer, P Eng, Sam Cui, and Stephen Lee. Transition from manual to automated pavement distress data collection and performance modelling in the pavement management system. In *TAC 2016: Efficient Transportation-Managing the Demand-2016 Conference and Exhibition of the Transportation Association of Canada*, 2016.

Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

Francois Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

116th Congress. National defense authorization act for fiscal year 2021, March 2020. URL https://www.govtrack.us/congress/bills/116/hr6395.

CSU. Contact. URL http://www.paver.colostate.edu/.

Anik Das, Md Nasim Khan, Saurav Barua, and Md Mizanur Rahman. An evaluation of life cycle cost analysis of airport pavement.

Iddo Drori. Course notes in introduction to machine learning, November 2020.

A Ellenberg, A Kontsos, Franklin Moon, and I Bartoli. Bridge related damage quantification using unmanned aerial vehicle imagery. *Structural Control and Health Monitoring*, 23(9):1168–1179, 2016.

FAA. *Airport Pavement Design and Evaluation*, volume 150/5320-6E of *Advisory Circular*. 2009.

FAA. *Airport Pavement Management Program (PMP)*, volume 150/5380-6 of *Advisory Circular*. 2014a.

FAA. *Airport Pavement Management Program (PMP)*, volume 150/5380-7B of *Advisory Circular*. 2014b.

FAA. *Airport Pavement Design and Evaluation*, volume 150/5320-6F of *Advisory Circular*. 2016.

FAA. *Airport Pavement Design and Evaluation*, volume 150/5320-6G of *Advisory Circular*. June 2021a.

FAA. Secondary navigation, Sep 2021b. URL https://www.faa.gov/uas/.

Steven J Fenves. Artificial intelligence-based methods for infrastructure evaluation and repair. *Annals of the New York Academy of Sciences*, 431:182–193, 1984.

David Ford. Ape ensures mission-ready airfields, Jan 2020. URL https://www.afcec.af.mil/News/Article-Display/Article/2065076/ape-ensures-mission-ready-airfields/.

Paul Freeman. Aardvark field, colorado springs, co, Nov 2017. URL http://www.airfields-freeman.com/CO/Airfields_CO_ColoradoSprings.htm.

G Gibson, R Milnes, M Morris, Hill N, and Simbolotti G. Aviation infrastructure, Aug 2011. URL https://iea-etsap.org/E-TechDS/PDF/T16_Aviation_Infrastructure_v4%20Final.pdf.

Google. Aardvark airfield, usaf, October 2021. URL https://earth.google.com/web/@39.03477675,-104.84813069,2033.01758638a,1583.47207945d,35y,0h,0t,0r.

Kasthurirangan Gopalakrishnan, Siddhartha K Khaitan, Alok Choudhary, and Ankit Agrawal. Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection. *Construction and building materials*, 157:322–330, 2017.

J Gu, Z Wang, J Kuen, L Ma, A Shahroudy, B Shuai, T Liu, X Wang, and G Wang. Recent advances in convolutional neural networks. eprint. *arXiv preprint arXiv:1512.07108*, 2015.

Jaroslav J Hajek. *Common airport pavement maintenance practices*, volume 22. Transportation Research Board, 2011.

Mostafa Hassanalian and Abdessattar Abdelkefi. Classifications, applications, and design challenges of drones: A review. *Progress in Aerospace Sciences*, 91:99–131, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

James V. Henrickson, Cameron Rogers, Han-Hsun Lu, John Valasek, and Yeyin Shi. Infrastructure assessment with small unmanned aircraft systems. *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2016. doi: 10.1109/icuas.2016.7502652.

Martin Herold and Dar Roberts. Spectral characteristics of asphalt road aging and deterioration: implications for remote-sensing applications. *Applied optics*, 44(20):4327–4334, 2005.

Martin Herold, Dar Roberts, Val Noronha, and Omar Smadi. Imaging spectrometry and asphalt road surveys. *Transportation Research Part C: Emerging Technologies*, 16(2): 153–166, 2008.

Laura Inzerillo, Gaetano Di Mino, and Ronald Roberts. Image-based 3d reconstruction using traditional and uav datasets for analysis of road pavement distress. *Automation in Construction*, 96:457–469, 2018.

Muhammad Irfan, Muhammad Bilal Khurshid, Shahid Iqbal, and Abid Khan. Framework for airfield pavements management—an approach based on cost-effectiveness analysis. *European Transport Research Review*, 7(2):13, 2015.

Veronica Kemeny. Afcec airfield pavement evaluation team wins one air force award, Aug 2018. URL https://www.af.mil/News/Article-Display/Article/1613125/afcecs-airfield-pavement-evaluation-team-wins-one-air-force-award/.

Sungjin Kim, Javier Irizarry, and Ruth Kanfer. Multilevel goal model for decision-making in uas visual inspections in construction and infrastructure projects. *Journal of Management in Engineering*, 36(4):04020036, 2020.

Starr D Kohn and Mohamed Y Shahin. Evaluation of the pavement condition index for use on porous friction surfaces. Technical report, CONSTRUCTION ENGINEERING RESEARCH LAB (ARMY) CHAMPAIGN IL, 1984.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25: 1097–1105, 2012.

David Lattanzi and Gregory Miller. Review of robotic infrastructure inspection systems. *Journal of Infrastructure Systems*, 23(3):04017004, 2017.

Man Liu, Bolin Fu, Shuyu Xie, Hongchang He, Feiwu Lan, Yuyang Li, Peiqing Lou, and Donglin Fan. Comparison of multi-source satellite images for classifying marsh vegetation using deeplabv3 plus deep learning algorithm. *Ecological Indicators*, 125:107562, 2021a. ISSN 1470-160X. doi: https://doi.org/10.1016/j.ecolind.2021.107562. URL https://www.sciencedirect.com/science/article/pii/S1470160X21002272.

Man Liu, Bolin Fu, Shuyu Xie, Hongchang He, Feiwu Lan, Yuyang Li, Peiqing Lou, and Donglin Fan. Comparison of multi-source satellite images for classifying marsh vegetation using deeplabv3 plus deep learning algorithm. *Ecological Indicators*, 125:107562, 2021b.

Rajib B Mallick and Tahar El-Korchi. *Pavement Engineering: Principles and Practice*. CRC Press, New York, 2018.

Ciro Manzo, Alessandro Mei, Rosamaria Salvatori, Cristiana Bassani, and Alessia Allegrini. Spectral modelling used to identify the aggregates index of asphalted surfaces and sensitivity analysis. *Construction and Building Materials*, 61:147–155, 2014.

Mathworks. Documentation: trainingoptions, 2021a. URL https://www.mathworks.com/help/deeplearning/ref/trainingoptions.html?s_tid=srchtitle_trainingoptions_1.

Mathworks. Documentation: bwskel, 2021b. URL https://www.mathworks.com/help/images/ref/bwskel.html;jsessionid=29c08cc6597dbb44d66613c49498.

Alessandro Mei, Ciro Manzo, Cristiana Bassani, Rosamaria Salvatori, and Alessia Allegrini. Bitumen removal determination on asphalt pavement using digital imaging processing and spectral analysis. *Open Journal of Applied Sciences*, 2014, 2014.

Christodoulos Mettas, Kyriacos Themistocleous, Kyriacos Neocleous, Andreas Christofe, Kypros Pilakoutas, and Diofantos Hadjimitsis. Monitoring asphalt pavement damages using remote sensing techniques. In *Third international conference on remote sensing and geoinformation of the environment (RSCy2015)*, volume 9535, page 95350S. International Society for Optics and Photonics, 2015.

Christodoulos Mettas, Athos Agapiou, Kyriacos Themistocleous, Kyriacos Neocleous, and Diofantos G Hadjimitsis. Detection of asphalt pavement cracks using remote sensing techniques. In *Remote Sensing Technologies and Applications in Urban Environments*, volume 10008, page 100080W. International Society for Optics and Photonics, 2016.

Abdul Wahid Mohammed. *Developing Quick Scanning Technique Using Satellite Images for Pavement Distress Identification*. PhD thesis, Texas A&M University-Kingsville, 2017.

NAPA. Napa fast facts - national asphalt pavement association, Nov 2014. URL `https://www.asphaltpavement.org`.

Meenal V Narkhede, Prashant P Bartakke, and Mukul S Sutaone. A review on weight initialization strategies for neural networks. *Artificial intelligence review*, pages 1–32, 2021.

Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. Activation functions: Comparison of trends in practice and research for deep learning. *arXiv preprint arXiv:1811.03378*, 2018.

Abdulkadir Ozden, Ardeshir Faghri, Mingxin Li, and Kaz Tabrizi. Evaluation of synthetic aperture radar satellite remote sensing for pavement and infrastructure monitoring. *Procedia Engineering*, 145:752–759, 2016.

Yifan Pan, Xianfeng Zhang, Guido Cervone, and Liping Yang. Detection of asphalt pavement potholes and cracks based on the unmanned aerial vehicle multispectral imagery. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 11 (10):3701–3712, 2018.

Anand Prakash, Brij N Sharma, and Thomas J Kazmierowski. Investigation into observational variations in pavement condition survey. In *Proceedings of the 3rd International Conference on Managing Pavements*, volume 2, pages 290–301. Citeseer, 1994.

Geoffrey M Rowe, Jay N Meegoda, Andris A Jumikis, Mark J Sharrock, Nishantha Bandara, and Hiroshan Hettiarachchi. Detection of segregation in asphalt pavement materials using the aran profile system. *Northeast asphalt user, Producer Group Newport Marriott, Newport, Rhode Island*, 30, 2002.

Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 3 edition, 2010.

Mohamed Y Shahin, Michael I Darter, and Starr D Kohn. Development of a pavement maintenance management system. volume ii. airfield pavement distress identification manual. Technical report, 1976.

Monte Symons. Life-cycle cost analysis for airport pavements, 2010.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

Marco-Michael Temme and Christian Trempler. Introduction of unmanned aerial vehicles into airport ground operations. 2017.

Yi-Chang Tsai, Vivek Kaul, and Russell M Mersereau. Critical assessment of pavement distress segmentation methods. *Journal of transportation engineering*, 136(1):11–19, 2010.

USAFCE and Cooper. *AFI 32-1041: Pavement Evaluation Program*. 2017.

USN. volume 1021/2 of *U.S. Naval Facilities Engineering Command Military Handbook*. 1988.

Russell T. Vought. Guidelines and discount rates for benefit-cost analysis of federal programs. *Washington, DC: US Office of Management and Budget*, OMB Circular No. A-94, December 2019.

Ming Wang, Ralf Birken, and Salar Shahini Shamsabadi. Implementation of a multi-modal mobile sensor system for surface and subsurface assessment of roadways. In *Smart Sensor Phenomena, Technology, Networks, and Systems Integration 2015*, volume 9436, page 943607. International Society for Optics and Photonics, 2015.

Roland Weibel and R John Hansman. Safety considerations for operation of different classes of uavs in the nas. In *AIAA 3rd" Unmanned Unlimited" Technical Conference, Workshop and Exhibit*, page 6421, 2004.

B Zakora and A Molodchick. Classification of uav (unmanned aerial vehicle), 2014.

Kaige Zhang, HD Cheng, and Boyu Zhang. Unified approach to pavement crack and sealed crack detection using preclassification based on transfer learning. *Journal of Computing in Civil Engineering*, 32(2):04018001, 2018.

Aleksandar Zlateski, Ronnachai Jaroensri, Prafull Sharma, and Frédo Durand. On the importance of label quality for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.