Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

MASTERTHESIS

Aircraft Dent Detection Utilizing Specular Reflections and Deep Learning

submitted by

Ronja vom Schemm

University of Hamburg

MIN-Faculty

Department of Informatics

Research Group Computer Vision, CV

Course of study: Informatics

Matrikelnr.: 6561954

Examiners: Prof. Dr. Simone Frintrop
Dr. Marc Bestmann
Supervisor: Aditi Mhatre

Abstract

The aviation industry requires frequent and thorough visual inspections to find and evaluate defects, which are expensive and time-consuming. Automating parts of the inspection process using robots and deep learning has the potential to improve speed and performance. Dents are the most difficult type of defect to detect, since they usually lack distinguishing colors, and can only be seen via shadows or reflections. To make dent detection easier and more reliable for deep learning models, a capturing setup utilizing specular reflections is tested. This necessitates the creation of a new dataset of aircraft surface images, where dents are made visible with the help of specular reflections. A new annotation method that makes use of an optical tracking system to automatically create annotations was developed to create the dataset. Two different models were trained on variations of the dataset and tested to determine their ability to detect dents in specular reflection images, and their viability for use in a robotic inspection scenario. This thesis shows that both RT-DETR and YOLOv12 have excellent dent detection performance on the new dataset, are fast and accurate when processing video, and can be suitably integrated into a robotic inspection setup.

Zusammenfassung

In der Luftfahrtindustrie sind häufige und gründliche visuelle Inspektionen erforderlich, um Defekte zu finden und zu bewerten, was kostspielig und zeitaufwendig ist. Die Automatisierung von Teilen des Inspektionsprozesses mithilfe von Robotern und Deep Learning hat das Potenzial, die Geschwindigkeit und Leistung zu verbessern. Dellen sind die am schwierigsten zu erkennenden Defekte, da sie in der Regel keine charakteristischen Farben aufweisen und nur über Schatten oder Reflexionen sichtbar sind. Um die Erkennung von Dellen für Deep-Learning-Modelle einfacher und zuverlässiger zu machen, wird eine Aufnahmekonfiguration getestet, das gerichtete Reflexion ausnutzt. Dazu muss ein neuer Datensatz mit Bildern der Flugzeugoberfläche erstellt werden, in dem Dellen mithilfe von gerichteten Reflexionen sichtbar gemacht werden. Zur Erstellung des Datensatzes wurde eine neue Annotationsmethode entwickelt, die ein optisches Tracking-System nutzt, um automatisch Annotationen zu erstellen. Zwei verschiedene Modelle wurden anhand von Variationen des Datensatzes trainiert und getestet, um ihre Fähigkeit zur Erkennung von Dellen in Bildern mit gerichteten Reflexionen zu untersuchen und ihre Eignung für den Einsatz in einem Roboterinspektionsszenario zu prüfen. Diese Arbeit zeigt, dass sowohl RT-DETR als auch YOLOv12 eine hervorragende Leistung für Dellendetektion auf dem neuen Datensatz aufweisen, bei der Verarbeitung von Videos schnell und akkurat sind und sich gut in ein robotergestütztes Inspektionssystem integrieren lassen.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Detecting defects in aircraft is an important but time-consuming task that is generally still carried out by technicians visually inspecting every surface. Automating parts of the inspection process reduces the amount of time humans need to be involved and may even increase accuracy. Deep learning methods have been applied to the problem of defect detection, but particularly in the field of aviation, the absence of large publicly available datasets limits their accuracy [1]. Dents are also often described as one of the more difficult defects to detect [23] due to their lack of distinguishing colors, requiring more specific lighting conditions to be detected.

Aircraft must undergo different types of inspections at specific intervals. These inspections are frequent and must be done thoroughly, which takes a lot of time. Since these inspections are mostly done by humans, they are expensive and prone to errors; human eyes will become fatigued after prolonged monotonous work, especially in the bright lighting conditions required for this work. Using people for visual inspection also poses the risk of injury since some parts of the aircraft are very difficult for humans to reach without aides, such as the upper parts of the plane's exterior or the inside of the fuel tanks.

Automating parts of the inspection process has the potential to reduce inspection times and the amount of mistakes. Deep learning could be used to quickly and accurately detect defects on images of the aircraft; several models, such as YOLOv5 [10] and Mask R-CNN [39][2], have previously been used with success. The process could also be improved by having the camera capturing the detection images mounted on a robot. A robotic setup has the potential to capture high-quality close-up images or videos of places that are difficult for humans to reach while also providing a movable light source. Since shallow dents only become visible under certain lighting conditions, having precise control over the camera position and angle of the light source is essential. Also, robotic inspection would not suffer from fatigue or distractions, which human inspectors are prone to after prolonged monotonous work.

Most previous research aims to detect and categorize different types of defects, and an often found result is that defects like lightning strikes and dents are the most difficult to detect and localize accurately. The problem with lightning strikes [21] is that most datasets include very few examples, and most deep learning models struggle to identify categories that are rare in the training data, highlighting the need for balanced datasets [23]. Dents are difficult for a different reason; while they are not always rare, they do not visually stand out from the background, especially when shallow. Other defects, like scratches and missing paint, have a different color than the surrounding undamaged surface, making them easy to detect and also determine their boundaries. Figure 1.1 shows examples of the visibility of different defects. Additional research is still needed on the detection of shallow dents specifically, so this type of defect is the focus of this work.

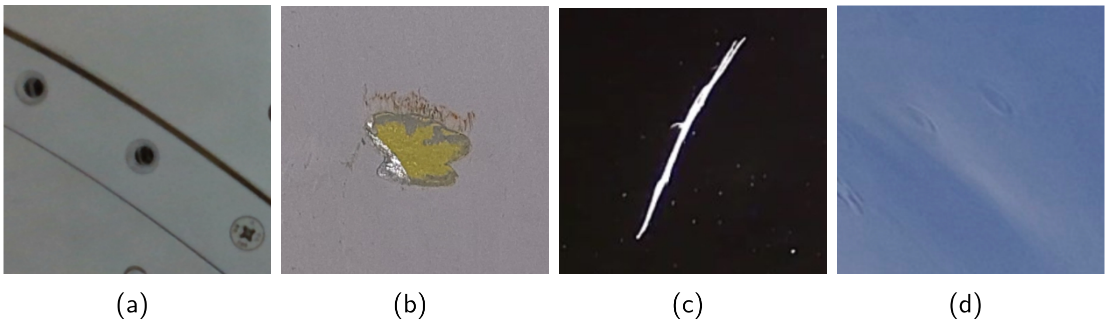


Figure 1.1: Visibility comparison of different defects (a) missing screws (b) missing paint (c) scratch (d) dent. Image (c) from [28] Image (d) from [23].

Small dents can be made more visually prominent by carefully controlling the lighting conditions [33]. Many aircraft surfaces are very shiny, causing bright specular reflections when illuminated. This can be a problem in some contexts, when these reflections hide surface details, but in the case of dent detection, they can be useful, since small deviations in the depth of the surface are very noticeable in the shape of the specular reflection. A special lighting setup taking advantage of these reflections can be used to make dents more visually apparent.

Deep learning models require large amounts of data that is specific to the desired task. The detection of defects, and specifically dents, on airplanes is a very specific task, so very few publicly available datasets exist. Existing datasets were not designed with a focus on specular reflections, so a new specialized dataset is required to train detection models for these lighting conditions.

The first contribution of this work is the creation of a new, publicly available dataset of dents on aircraft surfaces. The dataset features close-up captures of aircraft surfaces, with and without defects. The images contain specular reflections made by a light strip that make dents more visually apparent. Annotations of the dents are provided as bounding boxes. An additional set of images of dents on other non-aircraft metal plates is also provided for the creation of a larger combined dataset, which includes both images of aircraft parts and metal plates.

The second contribution is the development of a new data annotation method. An optical tracking system is used to measure the location of dents on a part and also to determine the relative position of the part and the camera, allowing the location of the dent in each image to be calculated automatically.

The third contribution is the evaluation of existing neural networks on their ability to detect dents under the specific lighting conditions of the new dataset. RT-DETR

[42] and YOLOv12 [34] are compared. They are each trained on the original dataset of airplane parts, as well as on the combined dataset to determine whether the inclusion of non-aircraft parts can improve performance. The inference speed of the models is also evaluated for their suitability for processing videos in real-time.

The final contribution is a test of the full dent detection system on a robot arm equipped with a video camera and light strip to determine its suitability for real-world applications.

The thesis was done in collaboration with the German Aerospace Center (DLR) at the Institute of Maintenance, Repair, and Overhaul, within the Robot-Assisted Inspection and Repair research group. The goal is to design an automatic inspection system that takes advantage of specular reflections to better detect defects like small indentations and provide a new dataset to train the system.

1.1 Organization

The thesis is split into 7 chapters. The first chapter introduces the topic of detecting defects on aircraft, and highlights the need for more research on dents. It gives an overview of the contributions and structure of the thesis. The second chapter provides background information on the various relevant subjects. Chapter 3 gives an overview of related works on defect detection in aviation as well as other industries. Chapter 4 covers the methods used during the thesis, including details about the dataset creation process. Chapter 5 provides information on what is contained in the final dataset as well as the deep learning results. Chapter 6 analyzes and discusses these results. The final chapter summarizes the findings of the thesis.

Chapter 2

Background

This chapter gives and overview on relevant topics, such as lighting and how it can be used to make dents more visible, optical tracking systems, like the one used for the new annotation method, and deep learning models, methods, and metrics.

2.1 Lighting and Reflections

The appearance of an object depends on the material properties and geometry of the surface and the specific lighting conditions. When light hits the surface of an object, portions of it will be absorbed, reflected, or, in the case of transparent or translucent objects, also refracted. In the case of opaque objects, which are relevant here, only the reflective and absorptive properties apply. There are two types of reflections, diffuse and specular.

Diffuse reflections scatter light equally in all directions, appearing equally bright regardless of viewing direction. The total amount of light diffusely reflected depends on how much of each wavelength of light is absorbed by the material. For example, a material that absorbs mostly yellow wavelengths will reflect all colors but yellow and appear blue.

The intensity of reflected light also depends on the angle between the incoming direction of light and the surface normal, obeying Lambert's Cosine Law: [5]

$$I = I_0 \cos(\theta)$$

Light incoming at oblique angles hits a larger surface area than light traveling along the surface normal, resulting in less illumination. Figure 2.1 shows the same light source illuminating an object at different angles. The area being illuminated is shown in orange, and is smallest when the angle θ between the incoming light and the surface normal is $0°$.

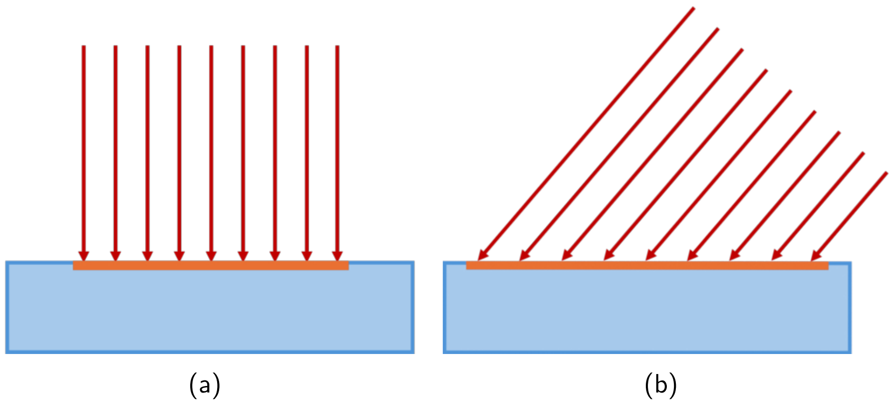


Figure 2.1: Light traveling (a) along the surface normal (b) at an angle

This characteristic makes bumps or divots in an otherwise even surface have darker and lighter areas, making them visually distinct from flat surfaces because the surface normal is oriented in different directions. A particularly extreme concavity or convexity lit by very oblique light rays may even have a shadowed area where no light rays are reflected. Figure 2.2 shows how the orientation of the surface normal in different areas results in variations in illumination. The areas labeled A have similarly oriented surface normals and therefore have diffuse reflections of equal brightness. Area B has a surface normal closer to the direction of the incoming light, making the reflections here brighter. In area C, the angle between incoming light and the surface normal is even larger than in areas A, resulting in less bright reflections.

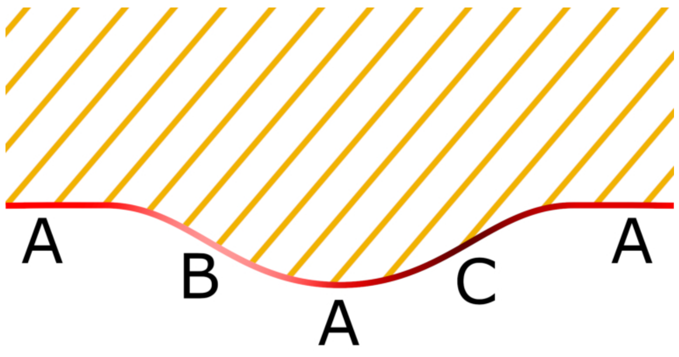


Figure 2.2: Brightness differences due to orientation of surface normal. Area B appears brighter due to its surface normal being closest to direction of incoming light. Area C appears darkest due to oblique angle of incoming light. Areas marked A have similar brightness to each other.

Specular reflections depend on the direction of incoming light relative to the surface normal. The incoming light v_i is reflected in the direction rotated by $180°$ around the surface normal $\hat{n}$ [30](see Figure 2.3). Mathematically, the specular reflection direction $\hat{s}_i$ is calculated as:

$$\hat{s}_i = v_{\|} - v_{\perp} = \left(2\hat{n}\hat{n}^T - I\right) v_i$$

When an even surface has a dent or bump, where the surface normal is oriented in a different direction, the specular reflection direction will be different in these areas, making these reflections useful for detecting surface anomalies.

An ideal specular reflector would reflect light only along the specular reflection direction. Most surfaces do not behave like ideal mirrors, because small irregularities in the surface normal will reflect light in slightly different directions, blurring the reflection [5]. The amount of light reflected in a given view direction $\hat{v}_r$ depends on the angle θ between this direction and the specular direction $\hat{s}_i$.

The Phong model expresses this relationship as: [30]

$$f(\theta; \lambda) = k_s(\lambda) \cos^{k_e}(\theta)$$

The exponent k_e describes the specularity of the surface, with larger k_e resulting in sharper reflections, and smaller values producing softer edges. $k_s(\lambda)$ is the specular reflection distribution, which describes how much of each wavelength is reflected. Most materials have a uniform distribution resulting in a white specular reflection [30].

In most situations there is not just one light source but also some amount of ambient illumination. This is usually caused by diffuse reflections from various nearby surfaces like walls and floors, but could also be from the sky, which is caused by the scattering of sunlight in the atmosphere [30].

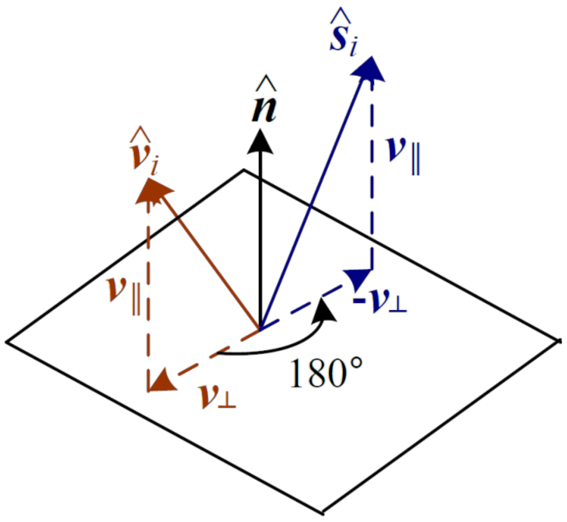


Figure 2.3: Specular reflection direction $\hat{s}_i$. Image from [31]

2.2 Coordinate Systems and Camera Calibration

2.2.1 Coordinate Systems

The pose of an object in 3-dimensional space has 6 degrees of freedom, 3 for position and 3 for orientation. These characteristics are defined relative to a base world coordinate system. For example, the world coordinate system may be a Cartesian coordinate system aligned to the floor and walls of a room, with the z-axis pointing up towards the ceiling. The coordinate system of a camera is generally defined with the origin at the camera's optical center and the z-axis corresponding to the optical axis. A point that was defined or measured in the world coordinate system can be transformed into the camera's coordinate system if the transformation between the camera and world systems is known.

The image coordinate system is a 2D coordinate system of the image plane, where the origin is in the center. The 3D coordinates of a point in the camera coordinate system can be projected to the 2D image coordinate system using the camera's intrinsic parameters.

A point in the image coordinate system can only be projected to a ray in the camera coordinate system, since there is no distance information.

The coordinate system used to store images as digital files consists of discrete pixels, with the origin usually being in the top left corner of the image, requiring an additional conversion from the image coordinate system to the pixel coordinate system [41].

2.2.2 Camera Calibration

In order to transform between the world and image coordinate systems, the camera must be calibrated. Calibration involves intrinsic and extrinsic parameters. The extrinsic parameters depend on the camera's current pose, the intrinsics are fixed based on how the camera was manufactured.

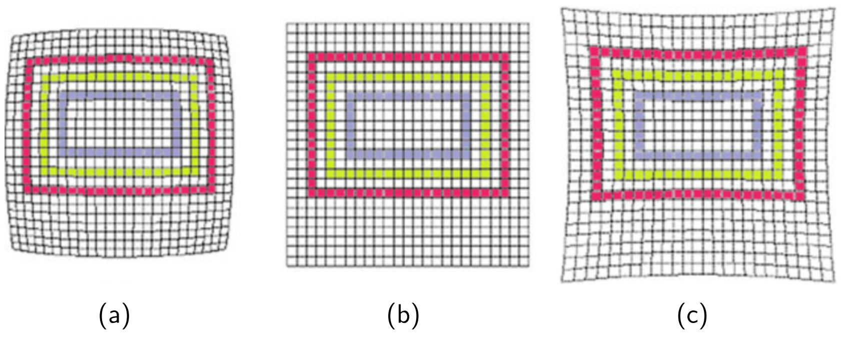


Figure 2.4: (a) Image distorted by barrel distortion (b) Undistorted image (c) Image distorted by pincushion distortion. Image adapted from [15]

Real cameras have some amount of lens distortion that affects how points in space are projected onto the image plane. The two main types are radial and tangential distortion.

Radial distortions are symmetrical around the optical axis of the camera and tend to be more severe farther away from the center of the lens. Negative radial distortions, called barrel distortions, shift points closer to the center, causing straight lines near the image edges to bulge outward, and right angles to become obtuse [15]. Positive radial distortions, or pincushion distortions, shift image points outward, resulting in inward-curving lines, and more acute angles. Figure 2.4 shows examples of these.

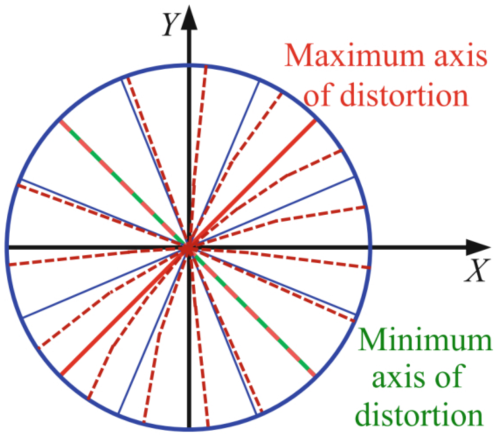


Figure 2.5: Tangential distortion. Solid blue and black lines represent the actual position of a line in space, dashed red lines the distorted position of these lines in an image. Figure from [41].

Tangential distortions shift image points in a tangential direction, as illustrated in Figure 2.5. This distortion will be greatest along a specific axis, shown in red, and least along the axis perpendicular to the maximum axis, shown as the red and green dashed line. Tangential distortions tend to be much smaller than radial distortions [41]. The radial distortion in x and y direction [3] can be approximated as:

$$x_{dr} = x\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \tag{2.1}$$

$$y_{dr} = y\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \tag{2.2}$$

and tangential distortion as:

$$x_{dt} = x + \left(2p_1 xy + p_2\left(r^2 + 2x^2\right)\right) \tag{2.3}$$

$$y_{dt} = y + \left(p_1\left(r^2 + 2y^2\right) + 2p_2 xy\right) \tag{2.4}$$

The camera calibration procedure determines the distortion coefficients k_1,k_2,k_3,p_1, and p_2, as well as the focal length (f_x, f_y) and optical centers (c_x, c_y). A calibration target with a known pattern of points is needed, such as a checkerboard pattern of equally sized squares. When the exact size of the squares is known, all vertices can then be calculated as points in 3D space. The calibration procedure requires at least 6 known points [41], but in practice many more points are used by taking multiple photos of the calibration target at different positions and orientations, which itself has more than 6 points.

2.2.3 Optical Tracking System

The film and video game industries have long made use of optical tracking systems to capture the 3-dimensional layout and movements of objects using markers attached in specific locations. Once the cameras are calibrated, their positions and lens distortions are known, making it possible to determine the 3D position of any markers visible to at least two cameras.

Different types of markers exist, the simplest being the passive marker. Passive markers are small spheres covered in retroreflective tape. These markers reflect light originating near the cameras and appear as bright circles in the camera images. Being spherical, it is not possible to determine the orientation of the markers, only the position, giving them 3 degrees of freedom. An object with a set of at least 4 markers with fixed relative positions can be combined into a 'target', allowing not just the position but also the orientation of the object to be tracked, giving the target 6 degrees of freedom. A requirement for the relative location of the markers is that the pairwise distances between all markers of a target are different, so there will be no ambiguity in the orientation of the target. Figure 2.6 shows one of the targets used. A tracking system will output the position of single markers and the position and orientation of targets relative to a coordinate system defined during calibration. A target may also be localized relative to another target instead of the global coordinate system.

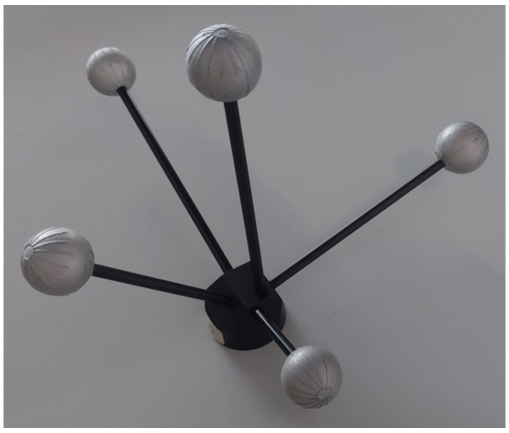


Figure 2.6: Target consisting of 5 retroreflective markers.

There is also a special type of target called a measurement tool. This object generally has an elongated shape with a small tip at one end. The measurement tool must be calibrated to determine the precise location of the tool tip relative to the attached markers. This is done by keeping the tip stationary on a hard surface while pivoting the body of the tool in different directions. If the tool tip has a ball at the end, its diameter is also needed for the calculation. Once calibrated, the position and orientation of the tool's tip can also be output by the system, allowing the user to determine the location of other objects or features by placing the tool tip on them.

2.3 Neural Networks

Deep learning methods excel at a wide variety of computer vision tasks. Unlike traditional methods, they do not require hand-crafted features or domain knowledge, but they are reliant on having large datasets for training, and specialized tasks require specialized datasets.

2.3.1 Computer Vision

Computer vision is the use of electronic systems to perceive and interpret a scene given some visual information. This information can take the form of RGB images or videos, but also radar, lidar, sonar, etc.... The information extracted can range from identifying and localizing objects to describing features or actions. Common computer vision tasks include classification, detection, tracking, and reconstruction. These tasks also have applications in a wide variety of industries.

Object detection is the task of detecting instances of certain object classes in images or videos. This requires determining both the object class and localizing it with a bounding box. There are two main categories of object detection models: one-stage and two-stage. Two-stage models determine candidate regions first and then classify and refine those regions, while one-stage models do both localization and classification together, which is often faster [41].

Semantic segmentation localizes objects by assigning each pixel in an image to a class. This allows for a more precise determination of the object outline than detection with bounding boxes. Instance segmentation also differentiates between multiple objects of the same class. Figure 2.7 (a) shows an image with a bounding box localizing a cat and (b) depicts segmentation of the same image, assigning every pixel to either the 'cat' or 'background' category.

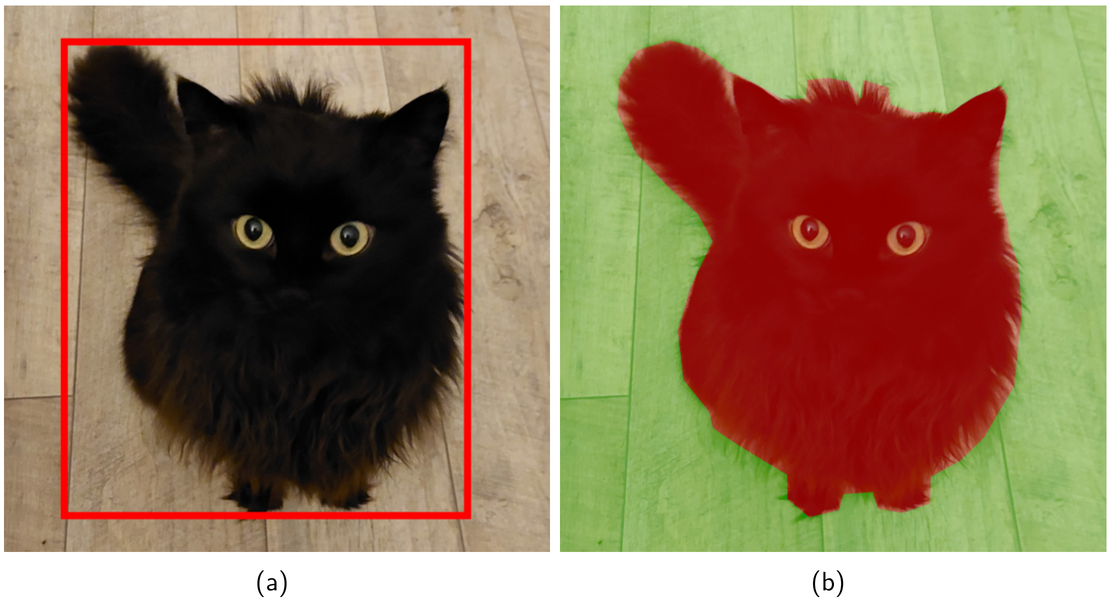


Figure 2.7: Detection and segmentation of an image of a cat.

2.3.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are often used for a variety of computer vision tasks [24]. CNNs contain convolutional layers, responsible for extracting features, and pooling layers that reduce the dimensionality of the extracted feature map [20]. Figure 2.8 shows the architecture of a generic CNN used for classification. The alternating convolution and pooling layers help the network learn features of the input image, while a separate classification head, usually consisting of one or more fully connected layers, learns to associate these features with the desired output classes. Convolutional layers are generally much more sparsely connected than fully connected layers, reducing the number of computations. Processing a fully connected layer with an input size of m and an output size of n has a runtime of $\mathcal{O}(m \times n)$. If there are only k sparse connections, the runtime will be only $\mathcal{O}(k \times n)$ [6], which is significantly lower, since k is usually much smaller than m. Convolutional layers make use of the spatial information contained in images and videos and can be used to extract features at a variety of scales. This allows CNNs to learn about both small details, as well as the larger spatial context and relationships of those features.

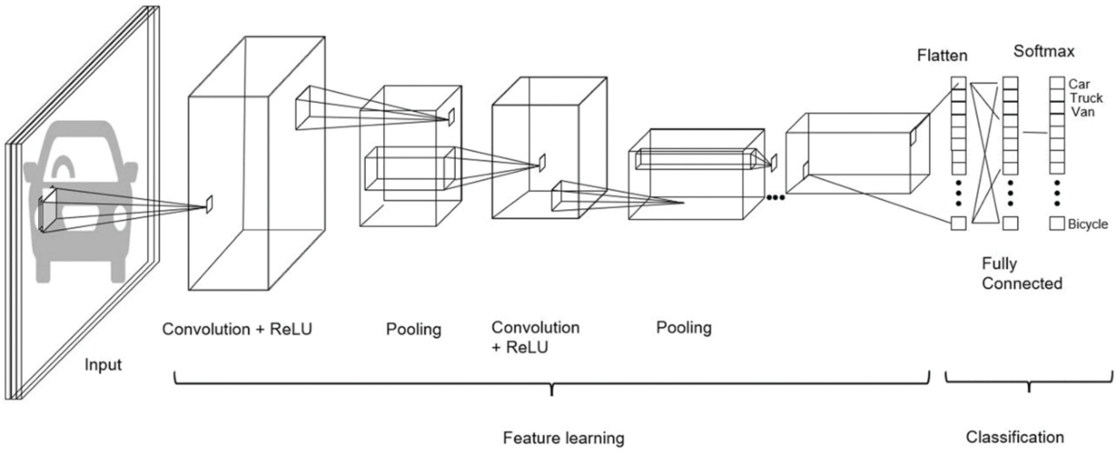


Figure 2.8: Architecture of a generic CNN used for classification. Image from [20].

2.3.3 Transformers

Transformers were introduced by Vaswani et al. in 2017[35]. They were originally designed for natural language processing tasks, such as machine translation, but have since been adapted for other tasks, notably computer vision, showing promising results.

Transformers process inputs as a sequence of tokens; in the case of natural language text, the input is usually a calculated embedding of a word or word fragment, and if the input is an image, the tokens are often small patches of the image. Transformers make use of multi-head self-attention, which determines the importance of each input to every other part of the input sequence. This allows interactions between distant parts of the input sequence to be considered, without any bias towards nearby tokens. Positional encodings are usually added to the input in contexts where location information is relevant, such as the position of a word in a sentence or position of a feature in an image. These position encodings may be predefined or learned. The transformer architecture consists of several stacked transformer layers, each of which contains multi-head self-attention and a fully connected feed-forward network. These layers also make use of residual connections, as depicted in Figure 2.9.

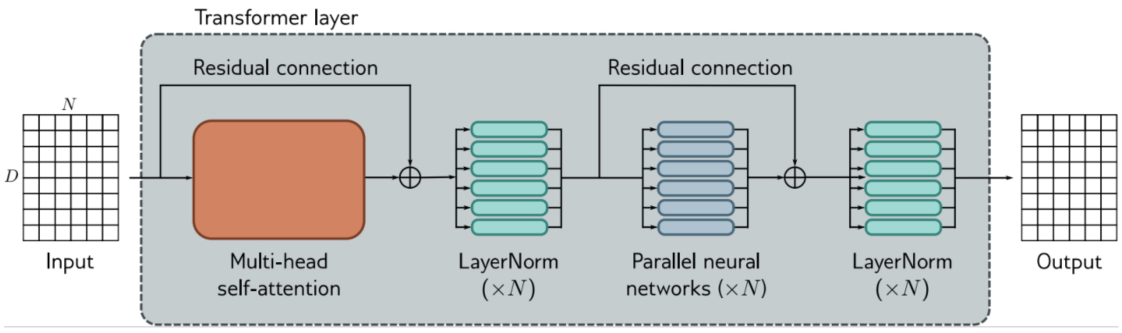


Figure 2.9: A transformer layer with multi-head self-attention, a fully connected neural network, and residual connections. LayerNorm is also often added. Image from [24].

One of the advantages of transformers is their ability to process inputs of varying sizes, although long sequences become computationally expensive due to the $\mathcal{O}(n^2)$ runtime of the attention mechanism. Transformers are capable of outperforming CNNs at computer vision tasks, but only when a sufficiently large amount of training data is available [24]. This high performance also comes at the cost of long training times and high computational cost on large inputs.

2.3.4 Training

Performance Metrics

The performance of a model can be evaluated with a variety of metrics. In a binary classification task, for example detecting the presence or absence of a class like dents, there are four possible outcomes, shown in Figure 2.10.

A true positive (TP) occurs when the model predicts a positive that matches the ground truth; in this case, it detects a dent, and there is actually a dent. A false positive (FP) happens when the model makes a positive prediction that does not match the ground truth, like detecting a dent when there is none. A false negative (FN) occurs when the model predicts a negative for a class, but the ground truth is positive. In this example it does not detect a dent even though there actually is one. A true negative (TN) is when both the prediction and ground truth are negative. The model does not detect a dent, and there is no dent.

A confusion matrix can be used to visualize the frequency of these four cases. When the task involves multiple classes, the confusion matrix displays the discrepancies between the predicted and ground truth classes, showing which classes are more likely to be 'confused' with each other by the model.

Accuracy is defined as the ratio between the number of correct predictions and the

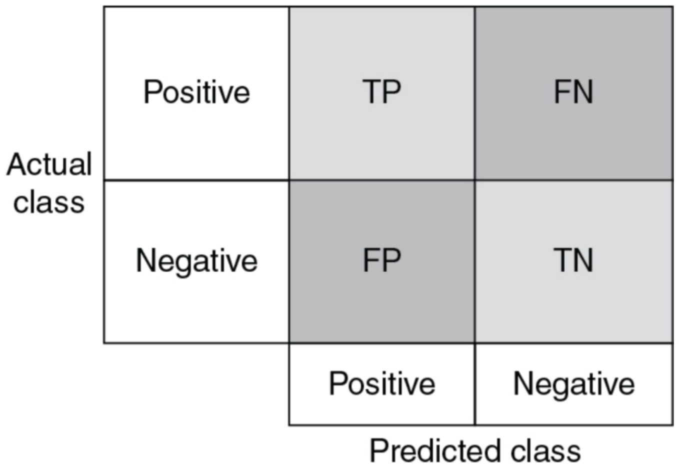


Figure 2.10: Confusion matrix for binary classification from [20]

number of total predictions. It gives an overall measure of the performance.

$$A = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.5}$$

Precision is the percentage of true positives out of all positives. This represents the probability that a given prediction is actually correct. A high precision model will produce few false positives

$$P = \frac{TP}{FP + TP} \tag{2.6}$$

Recall is the ratio between true positives and the sum of true positives and false negatives. This represents the percentage of class instances that were correctly identified. A model with high recall produces few false negatives.

$$R = \frac{TP}{TP + FN} \tag{2.7}$$

Intersection over Union (IoU) is often used to measure the accuracy of detection and segmentation models. IoU measures the similarity between two areas, such as a predicted bounding box and a ground truth bounding box, using the overlap. IoU is defined as the intersection of two areas A and B divided by the union of the two areas:

$$IoU(A, B) = \frac{|A \cap B|}{|A \cup B} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{2.8}$$

The higher the IoU score is, the more overlap the two areas have. IoU is often used as part of a loss function to determine the accuracy of predictions during training. IoU can also be used during post-processing of predictions to detect duplicate predictions of the same area. Non-maximum supression (NMS) removes duplicates of the same class above a certain IoU threshold, keeping only the highest confidence prediction. IoU is also used to define how similar a prediction must be to a ground truth in order to be considered a positive detection.

When there is no overlap between two bounding boxes, the IoU score is 0 regardless of how far away they are from each other. During training, improvements in location that do not produce an overlap are not reflected in IoU Loss, providing no information for the model to learn from. To combat this, GIoU was developed, which takes the distance into account. GIoU is defined as:

$$GIoU = IoU - \frac{|C \backslash (A \cup B)|}{|C|} = \frac{|A \cap B|}{|A \cup B|} - \frac{|C \backslash (A \cup B)|}{|C|} \tag{2.9}$$

Here, C is the smallest convex hull of A and B. This additional term gets smaller the closer A and B are to each other, providing feedback about location accuracy even when there is no overlap between A and B.

Optimizers

Optimizers are responsible for updating model weights during training in a way that minimizes the loss function, improving model performance.

Stochastic Gradient Descent (SGD) updates weights based on an individual training sample or a small batch of samples, [24] unlike gradient descent, which uses the entire dataset to calculate the gradient. This also introduces some randomness, which helps avoid local minima, improving performance. Momentum can also be added to SGD; this keeps a weighted average of previous gradients, which helps reduce oscillations and improves convergence speed.

Adaptive Moment Estimation (Adam) [14] combines momentum and the weighted moving average of squared gradients from RMSprop. The learning rate is adapted for each parameter.

Adam with Weight Decay Correction (AdamW) [19] is a modification of Adam that decouples weight decay from the gradient-based update, making weight decay more consistent.

Rectified Adam (RAdam) [18] is a variant of Adam with an added rectification term that reduces the large variance of the adaptive learning rate during early training.

Helpful Techniques

A. Early Stopping

Large models have the capacity to overfit to the training data. This happens when the training error continues to decrease, but performance on the validation set starts to drop. To avoid overfitting, the configuration that performs best on the validation set must be saved. When validation performance ceases to improve for a predefined number of epochs, training can be ended early and the best model returned [6].

B. Data Augmentation

Data augmentation is often used to increase model performance when the amount of training data is limited or lacks diversity. Data augmentation artificially increases the number of available training samples by applying one or more operations to the existing data. Commonly used data augmentations for images include rotating, flipping, shifting, or cropping the image; shifting the hue, saturation, or brightness; doing shear or perspective transformations; and mosaic combination of multiple images [20]. Increasing the amount and variety of training samples generally improves performance. Additional training data can also be created by using generative models trained on the dataset.

C. Transfer Learning

When only a small amount of data is available for a specific primary task, transfer learning can be used to improve model performance. In transfer learning, the model is first trained on a different but related task, for which ample training data exists. The model can then be fine-tuned by training on the dataset for the primary task. This allows the model to learn general properties from a large dataset and then learn the more specific details of the more specialized dataset.

The MS COCO [17] dataset is commonly used to pretrain models for visual tasks. MS COCO stands for **M**icro**S**oft **C**ommon **O**bjects in **CO**ntext. Unlike older datasets, which were dominated by images with few objects, mainly in 'canonical' positions, COCO features objects in more natural contexts and orientations, like cats lying on a keyboard or multiple airplanes at an airport (see Figure 2.11). On average, images contain 3.5 different categories and 7.7 object instances.

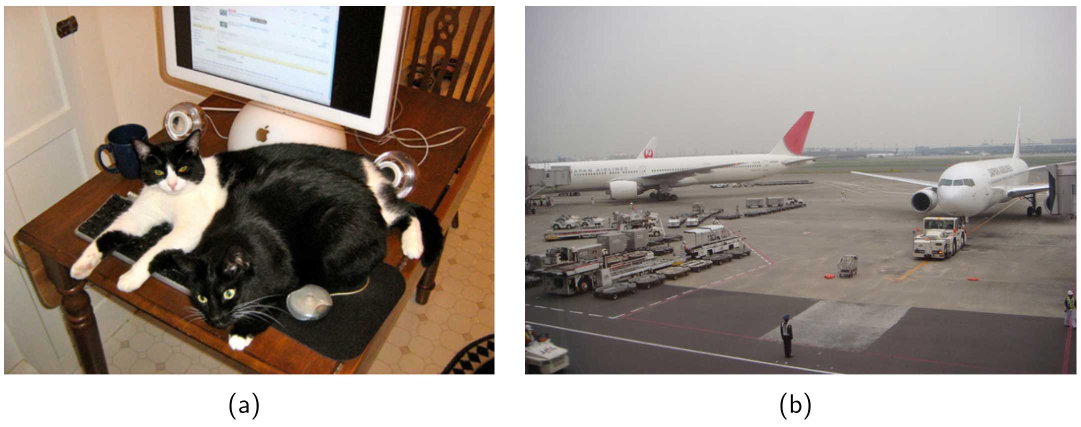


Figure 2.11: Example images from the COCO dataset (a) cats on a keyboard and (b) airplanes surrounded by other vehicles and people

Chapter 3

Related Works

Many deep learning approaches for detecting and localizing defects on aircraft have been tested in recent years, but research in other industries may also provide useful insights.

3.1 Defect Detection on Aircraft

3.1.1 Detection

The recent work by Huang et al. [10] proposes ASD-YOLO, an improvement of YOLOv5 [12], for the localization and classification of different aircraft defects.

Two different datasets are used in this work. The first dataset, ASD-942, consists of 314 images from a public dataset[1] that were extended to 942 via data augmentations. The defects in this dataset are dents, cracks, missing screws, peeling paint, and scratches. The second newly created dataset, ASD-505, consists of 505 images; 312 of the images were captured from a distance, the other 193 were close-up captures. The defects in this dataset are missing screws, missing caps, and lost tools.

ASD-YOLO uses a deformable convolution module and a global attention mechanism to improve feature extraction. A contextual enhancement module is used to improve feature representations of small defects. Additionally, an exponential sliding average weight function is introduced to reduce problems caused by class imbalance in the dataset. Multiple models were tested, and their proposed model performed best overall, followed by the original YOLOv5. The results also show that specifically for the category of dents, YOLOv8 [13] had the highest average precision of 77%.

Analysis of the different defect categories showed that scratches and missing screws had the highest false negative rates, likely due to scratches being only slightly visible, and screws being very small. This highlights the problem that many systems, even with their improvements, have in detecting small or subtle defects.

In their most recent work, Plastropoulos et al. [23] create a relatively large dataset containing different defect types but focus their work on detecting the most problematic

[1]https://universe.roboflow.com/ddiisc/aircraft_skin_defects

category, which is dents. The dataset consists of 1518 images with 6816 annotations of dents, missing paint, scratches, screws, and repairs. Dents are the least common defect (8%) in the dataset. They also experimented with different image acquisition techniques for creating the dataset, like handheld and drone-mounted cameras. They learned that defects like dents are barely visible in drone footage, and that handheld cameras allow the user to adjust the angle to make defects more visible.

Their experiments on many different models showed that EfficientDet D1 [32] performed best on this dataset. This model had a recall of 44% and a precision of 71% for dents. This is the second-lowest precision result out of the 5 classes. This is attributed to the small number of annotations in the dataset as well as dents having less visual information in images than other classes. Their exchange with experienced technicians also confirmed that this is a very difficult task, even for professionals, that often requires using a flashlight or changing the viewing angle to reveal discontinuities in reflections.

These findings highlight the importance of designing an image-capturing setup specifically for making dents more visible with light and specific viewing angles, and the need for a larger balanced dataset featuring many dents.

3.1.2 Segmentation

There are also some segmentation-based approaches to aircraft defect detection, mostly utilizing Mask R-CNN [8].

Yasuda et al. [39] created a dataset of 13 images with about 200 annotations of dents, dings, scratches, lightning strikes, missing fasteners, and corrosion. Their preliminary results show the viability of the model even on such a small dataset. The authors also highlight the need for integration of such a defect detection approach into broader aircraft inspection procedures.

Bouarfa et al. [2] created a dataset of 56 images segmented into dents and background. The model was initially trained for 15 epochs with the ResNet layers fixed, then evaluated, trained for 5 more epochs with the ResNet layers unfrozen, and final performance evaluated. Due to the small dataset size, they made use of 10-fold cross-validation and achieved 69% precision and 57% recall with Mask R-CNN. The interim evaluation showed that the final training with unfrozen layers greatly improved performance by 15% precision and 11% recall.

They found that the binary segmentation of just dents and background sometimes causes things like reflections, rivets, and raindrops to be misidentified as dents, and suggest adding additional classes for these commonly mistaken features in future work. They also suggest using a Generative Adversarial Network (GAN) [7] to expand the dataset size.

The authors discuss the higher computational cost of a segmentation model, like Mask R-CNN, over detection models like YOLO, and how the long training duration impacted their training to 15 epochs when using k-fold cross-validation. The tedious and time-consuming nature of creating segmentation masks is also noted as a drawback of their segmentation-based approach to dent detection.

3.1.3 Findings

Even though a lot of researchers have created or compiled datasets of defects, only one of the mentioned datasets is available publicly, so there continues to be a demand for datasets like these to be published. Each of the mentioned datasets is relatively small on its own, which limits the performance of various models trained on them. More datasets being published, even small ones, would allow datasets to be combined into a larger, diverse collection that meets the needs of deep learning models. The above works demonstrate that dents continue to be difficult to detect, both because they are often misclassified as other defects or features, and because they are often only visible from certain angles or with proper lighting conditions.

The datasets used in previous works often feature images taken from a distance, and the dents featured are often large and easy to detect. Figure 3.1 shows example images from some of the papers described above. There is a need for more images of smaller, shallow dents, since there is less representation of these available, and more research is needed to develop methods for making these dents easier to detect.

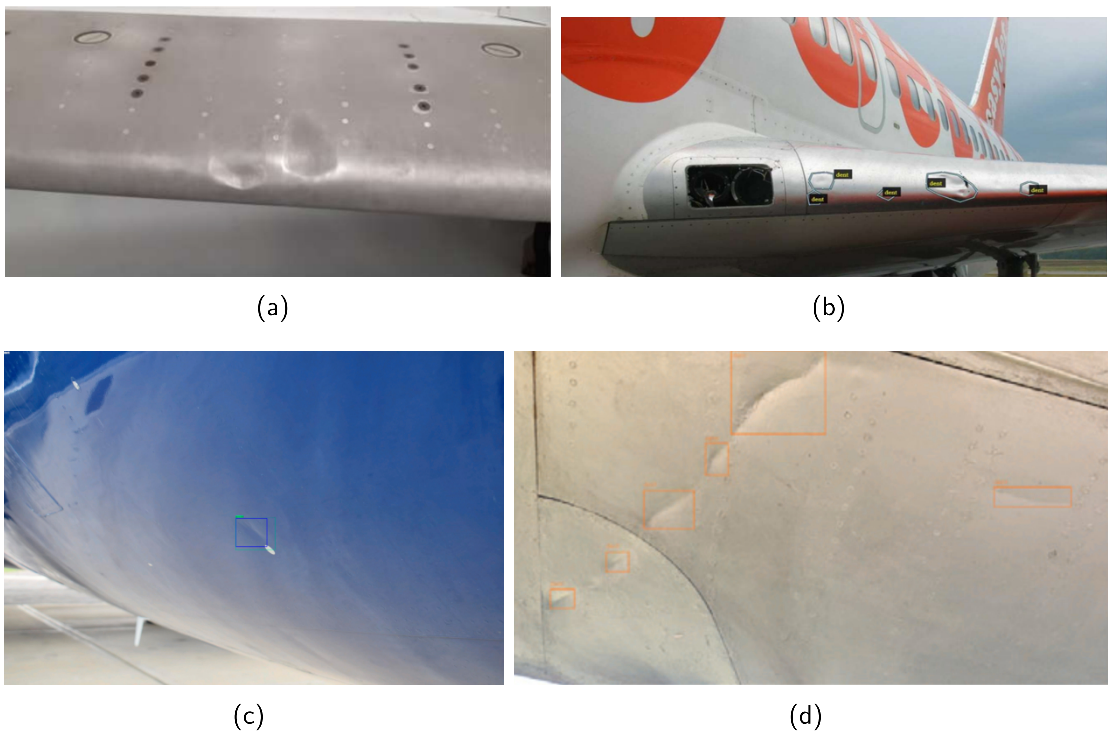


Figure 3.1: Aircraft dent images used in other works. Images from (a) and (b) Bouarfa et al. [2] (c) Plastropoulos et al. [23] (d) Huang et al. [10]

The works on segmentation have created much smaller datasets, because the annotation of segmentations is much more time-consuming than the annotation of bounding boxes. Segmentation models also require a long time to train, even on these small datasets, which can limit the number and duration of experiments when there are time constraints.

3.2 Defect Detection in Other Industries

Since the number of papers on the detection of aircraft defects is small, and not all include dents as defects, the detection of dents and similar defects in other industries was also studied, with a focus on works utilizing specialized lighting setups.

Different defects occur during the production process of curved glass, but the most difficult to detect here are also small, shallow dents, which were studied by Wang et al. [37]. The authors note how dents on glass have very low contrast with the background, which also has a non-uniform gray color caused by the curvature of the glass. This description is very similar to how shallow dents appear on aircraft surfaces. The authors create a dataset of 360 images of defective and non-defective samples. Figure 3.2 shows some of the dents from the dataset.

A pruned version of DenseNet-121 [11] is used to classify images as defective or acceptable. Due to the small dataset size, 5-fold cross-validation is used, and the impact of data augmentation tested. They were able to increase performance from 85% to 100% using crops, flips, and scaling as data augmentation.

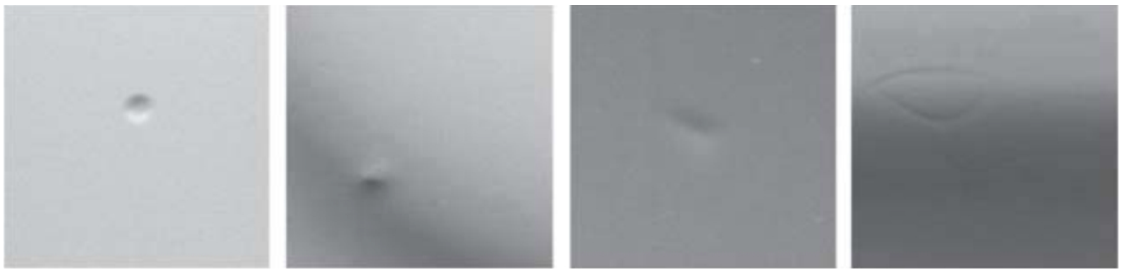


Figure 3.2: Small dents on glass. Image from [37].

Shi et al. [27] detected defects in the polarizer film used in LCD panels. Since some concave or convex defects are only visible under certain lighting conditions, they developed a structured light capturing setup to make these defects visible.

Their setup, shown in Figure 3.3, uses a grate placed over the light source to generate a binary stripe pattern that makes these defects detectable. The lighting setup and camera are positioned over the platform that manufactured polarizers are transported on. The authors note that using a grate to produce the strip pattern is more reliable than creating stripes virtually by using a monitor. The tiny convex or concave impurities in the polarizer appear much like dents on metallic surfaces under these conditions.

The large captured images were processed into small image patches (50x50) and labeled as dent or normal. They tested 8 different CNN models that all performed well on the training and validation sets, but only 3 performed well on the test set. GoogLeNet [29], ResNet [9], and DenseNet [11] generalize better to the differences in the test set.

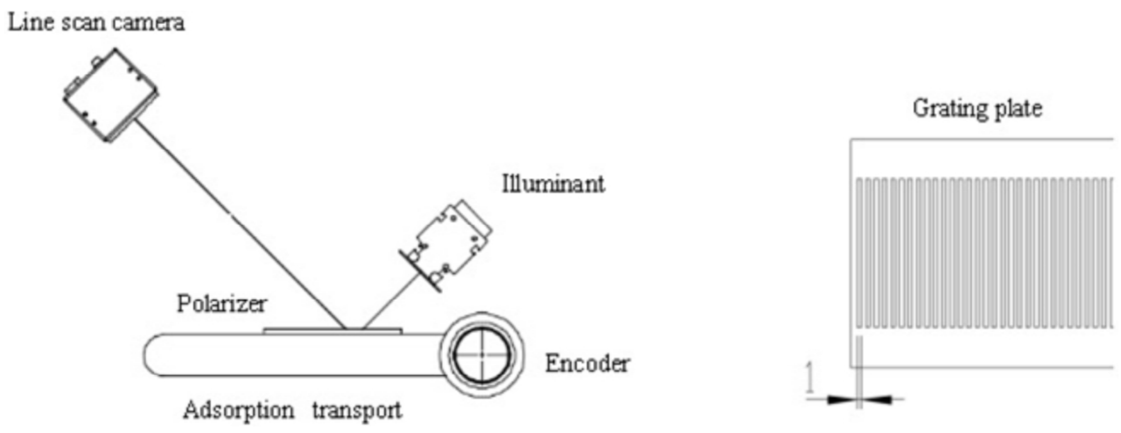


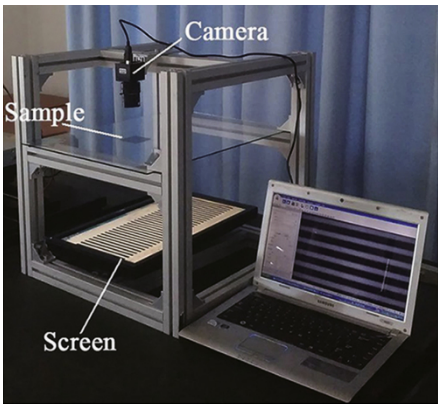
Figure 3.3: Grate lighting setup for polarizer defects. Image from [27].

This discrepancy between validation and test performance was likely caused by a difference between their training, validation, and test sets, because polarizers from different factories were used in these sets. Even though the image-capturing setup was the same, the different manufacturing processes caused slight variations in transparency and refractive index, which introduced slight systemic differences in the datasets. This was confirmed by splitting the test set into two test sets from the two different manufacturers. These two test sets showed drastically different performance for most models, and consistently showed higher performance on one of the test sets.

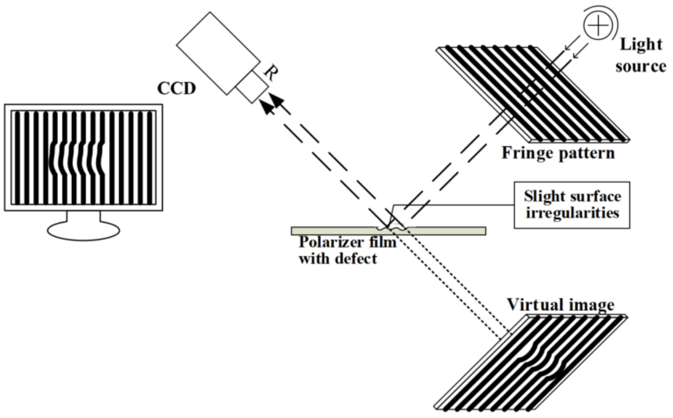
They also utilized a Deep Convolutional Generative Adversarial Network (DCGAN) [25] to generate more samples for training on a combination of the original dataset and varying number of synthetic samples. All three models improved with training on the larger datasets, and ResNet showed the best performance, with 95% accuracy. With the expanded dataset, the performance discrepancy on the two test datasets was also greatly reduced.

Other works also used light stripes on polarizers to make defects more visible but applied traditional methods for classification. Lai et al. [38] made defects like dents, bubbles, impurities, stains, and scratches more visible by placing an LCD screen underneath the polarizer sample, letting the light shine through, as depicted in Figure 3.4 (a). The authors also emphasize the importance of automating this type of inspection, due to the amount of time needed for inspecting each part, and errors caused by the subjective assessment or fatigue of even experienced inspectors. They also experimented with different-sized stripe patterns, and concluded that the optimal width of the light stripes should be 2-3 times the size of the defect. Yen et al. [40] used a similar setup, but the display with the stripe pattern was positioned above the sample and angled so the light would be reflected into the camera, as depicted in Figure 3.4 (b).

Park et al. [22] also developed a system using light stripes to detect dents on manufactured car parts. A striped cover was placed over an LED light source to produce the stripe pattern, much like the method by Shi et al. Images were sampled from video clips



(a)



(b)

Figure 3.4: Striped lighting setups for polarizer defects. Image (a) from [38] (b) from [40].

of a car fender with dents ranging in size from 0.5-10 mm. The images were processed into small patches (32x32), depicted in Figure 3.5, to serve as input for the R-CNN to carry out binary classification. To localize a dent in the whole image, a heatmap was created using the last convolution layer, and a bounding box was estimated. This method achieved 98.5% classification accuracy of patches.

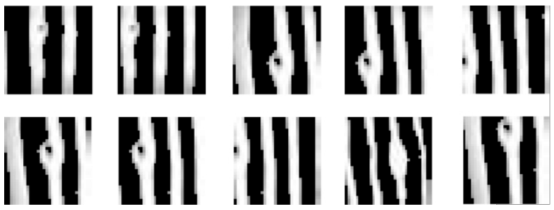
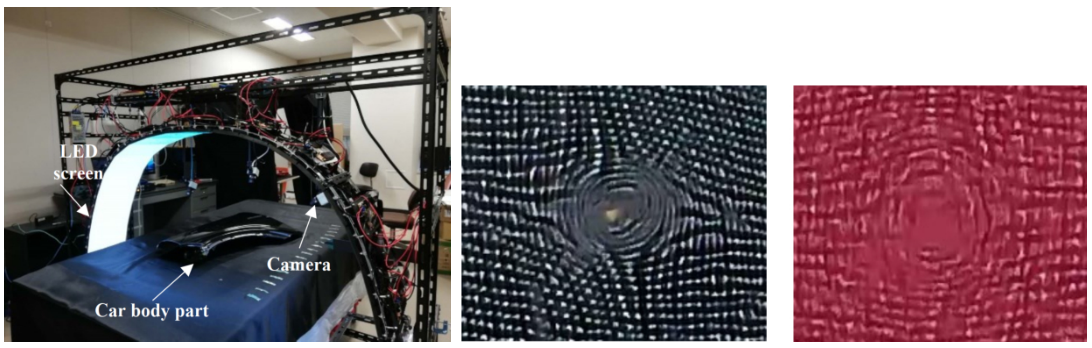


Figure 3.5: Image patches of dents under striped lighting. Image from [22].

Sikun et al.[28] used dual images for the damage detection on car parts. They used structured light and captured images with the camera focused on the reflection for dent detection, and captured images using homogeneous light with the camera focused on the surface to detect scratches. The structured light setup consists of multiple combined LED screens that project a dot pattern. The screens curve around the inspection area, as depicted in Figure 3.6 (a). Image (b) shows how the dot pattern warps around dents. Mask R-CNN was used for dent detection, while scratches were detected separately with traditional methods. They note that car parts with a higher reflectivity have a higher detection rate because the structured light is reflected more clearly.

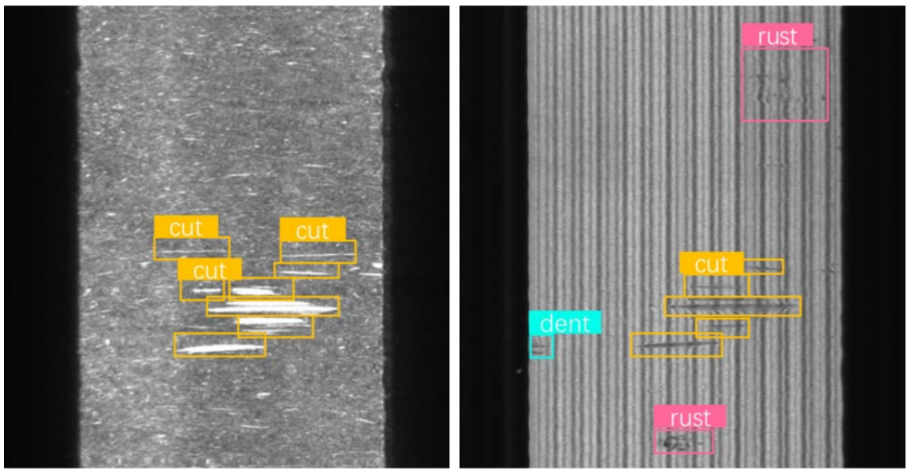
Tao et al.[33] also used two image-capturing setups for the detection of different damage types on the surface of drum-shaped rollers. The parts were lit with a fine striped pattern to make rust and shallow dents more visible, and again with a line light source, which made abrasions and cuts more visible. Figure 3.7 shows how more cuts



(a) (b)

Figure 3.6: (a) Structured lighting setup for car defects. (b) Dot pattern displaced by dents. Images from [28]

are detected under line light, and dents and rust only become visible under structured light. Their results show that the available YOLO models performed better than other architectures like EfficientDet and Faster R-CNN.



(a) (b)

Figure 3.7: Defects under different lighting options (a) line light (b) stripe pattern. Image from [22].

These works show how different types of structured light can be used to make very small dents and defects more visible in other industries. These methods can also be applied to aircraft dents.

3.3 Overview

Table 3.1 summarizes the methods and datasets described in this section.

Year	Authors	Architecture[2]	Task[3]	Application	Defects	Dataset Size[4]	Public Dataset	Structured Light
2024	Huang	YOLOv5*	C,D	aircraft	**dents**,others others	314+a 505	✓	
2024	Plastropoulos	EfficientDet, +	C,D	aircraft	**dents**,others	1518+a		
2021	Yasuda	Mask R-CNN	D,S	aircraft	**dents**,others	13		
2020	Bouarfa	Mask R-CNN	D,S	aircraft	**dents**	56		
2019	Wang	DenseNet-121*	C	glass	**dents**	360+a		
2024	Shi	ResNet, +	C	polarizer	**dents**	14503		✓
2016	Lai	traditional	D	polarizer	defects	4x150		✓
2015	Yen	traditional	D	polarizer	bumps	2		✓
2020	Park	R-CNN*	C,D	car	**dents**	8017		✓
2023	Sikun	Mask R-CNN	D,S	car	**dents**,others	350+a		✓
2022	Tao	YOLO*	C,D	drum rollers	**dents**,others	3332+a		✓

Table 3.1: Overview of defect detection papers.

[2]When multiple models were tested, only the best-performing one is listed, followed by a '+'. An asterisk denotes that the tested model was inspired by or is an extension of the listed model.
[3]Tasks: C=Classification, D=Detection, S=Segmentation
[4]Dataset Size: '+a' denotes that the authors explicitly mentioned using data augmentation.

Chapter 4

Methodology

The proposed automated dent detection system consists of three components: a camera and lighting setup capable of capturing images where dents are visible, a software system capable of detecting dents in the input images, and a large dataset specialized for this task to train the deep learning model on. Since there are very few public datasets of aircraft defects, and no dataset specifically focused on specular reflections of aircraft dents exists, a new dataset must be created. This chapter describes the methods used to create the new dent dataset, as well as the detection models used.

4.1 Dataset Creation

Deep learning models require large datasets that are both specific to the task and varied enough to represent real-world conditions. Ideally the dataset would feature only real airplane parts, with dents that were 'naturally' acquired during use, to most closely resemble real inspection scenarios. When only a few parts with a few dents are available the number of photos that can be extracted may not be enough to train a ML model capable of generalizing. To increase the available amount of data, additional dents were created in airplane parts, and another set of dented metal plates was created.

The dataset should include dents of varying sizes and depths, as well as other features commonly found on the aircraft skin, such as screws, rivets, and joints. Other defects like scratches or missing paint and screws may also be included if available. How specular reflections interact with these features should also be captured. Images should be captured at varying distances with varying lighting conditions. To annotate the images for use in deep learning, a new method using an optical tracking system was devised to automatically annotate the location of dents.

4.1.1 Workflow

The process of creating the dataset of dent images consists of 7 steps, that must be repeated for each part. Figure 4.1 shows the workflow of this process. First, the part is assessed for existing dents. Then, additional dents are made, if needed. Next, the locations of the dents on their respective part are measured with the help of an optical tracking system and saved. After these initial preparation steps, photos can be taken and automatically annotated. Now the lighting conditions are prepared for the next series of images, followed by carefully positioning the part and camera. A photo is then taken and automatically annotated. These steps of adjusting the positions and capturing an image are repeated many times. After the part has been sufficiently photographed, the lighting is adjusted to a different setup, and the positioning and capturing is repeated. Once the images of the part have been captured, the annotations are reviewed and some unwanted images discarded.

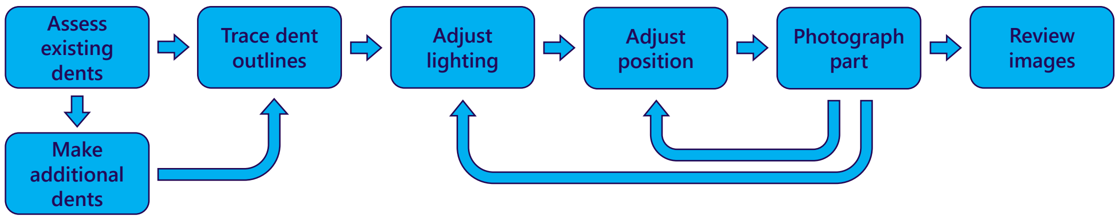


Figure 4.1: Dataset creation workflow.

4.1.2 Dent Creation

The 6 available aircraft parts shown in Figure 4.2 were cleaned, examined, and evaluated for existing dents and damage. The part taken from the left wing of a Boeing B737 is mainly white in color with some black lines and text, and is referred to as Part A in the following. Part B is a rectangular white fuselage section from an Airbus A340. It has the characters "3B-" on it in black, as well as some thin black and red lines that appear to have been added with markers. Part C is a large rectangular section of Airbus A330 fuselage. It has a white, highly specular surface and appears to have had some writing and arrows applied with a black marker. Parts D, E, and F are slat sections from an Airbus A240, each with a white side and a light gray side.

Part A is the only part with dents of known origin; these were artificially created during previous experiments at the DLR [16]. The majority of these dents were created by an impact gun, the others by a large kettlebell weight. Some of the impact sites were excluded due to not having a visible indentation in addition to the surface damage. All dents, as well as other defects, are on the top surface of the wing section. Parts B and C have dents of unknown origin. Parts D, E, and F appear to have many dents caused by hail on the white side, as well as a few dents of other origins on the gray side, but the details could not be verified.

Part name	Type	Colors	Dent origins
Part A	Boeing B737 left wing	white, black	impact gun, kettlebell
Part B	Airbus A340 fuselage	white, black, red	unknown, hammer
Part C	Airbus A330 fuselage	white, black	unknown, hammer
Parts D,E,F	Airbus A340 slat	white, light gray	hail, unknown

Table 4.1: Plane parts and their dents.

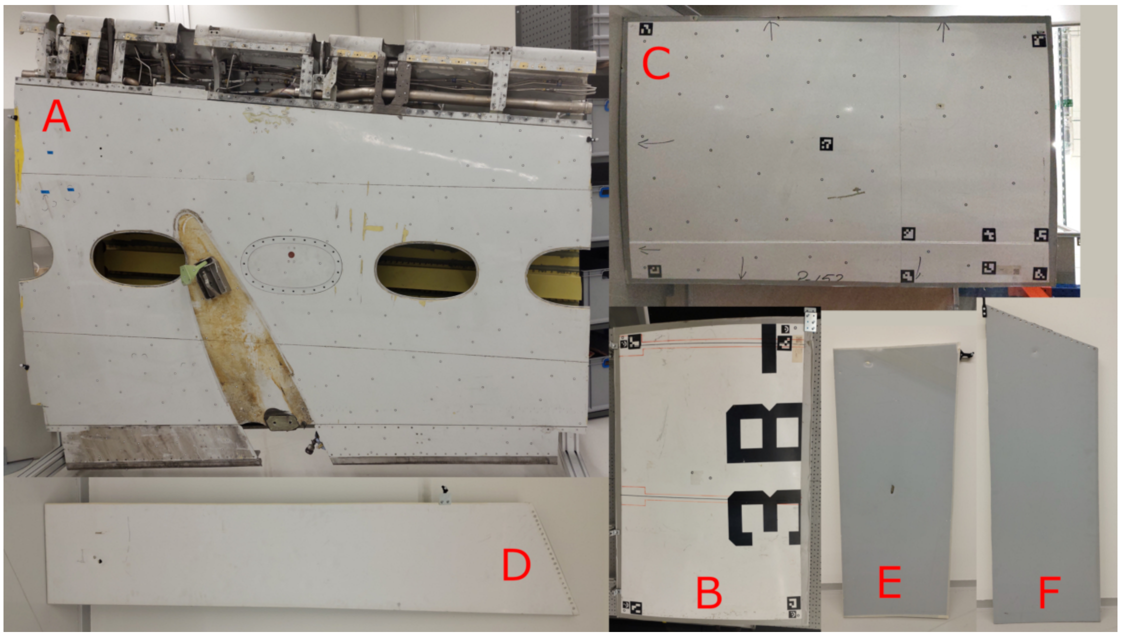


Figure 4.2: The 6 plane parts used.

Initially, only parts A, B, and C were available, which had the fewest dents, so additional dents were added to parts B and C. The panels were laid flat on the floor, and some layers of cloth were placed on the impact site that was struck with a hammer. Many of the existing dents had additional damage that may make them easier to detect, so more dents without damage were desired. The cloth padding helped prevent the paint from being damaged at the impact site, resulting in a dent with no other damage. The edges of the panels were held down when struck to reduce vibrations, making the material easier to dent. Table 4.1 gives an overview of the parts used.

Additionally, dents were made in 5 metal plates to increase the number and variety of available dents. The aluminum and steel plates had thicknesses ranging from 0.5 mm to 2.0 mm, and were painted with colors commonly found on the exterior of commercial airplanes. The available aircraft parts were mostly white in color, with some black lines or text, so more variety in color and designs was needed to make the dataset more representative of real-world conditions. Some dents were made directly with a hammer, some were made by holding a cap nut against the surface and striking it with a hammer. The cap nut was used increase the diversity of dents by modeling an impact with a more rounded object, since both ends of the hammer used have relatively sharp edges. The plates were also stabilized to reduce vibrations, most by being clamped between two wooden frames with circular cutouts. Figure 4.3 shows the painted plates and Table 4.2 lists the specifics.

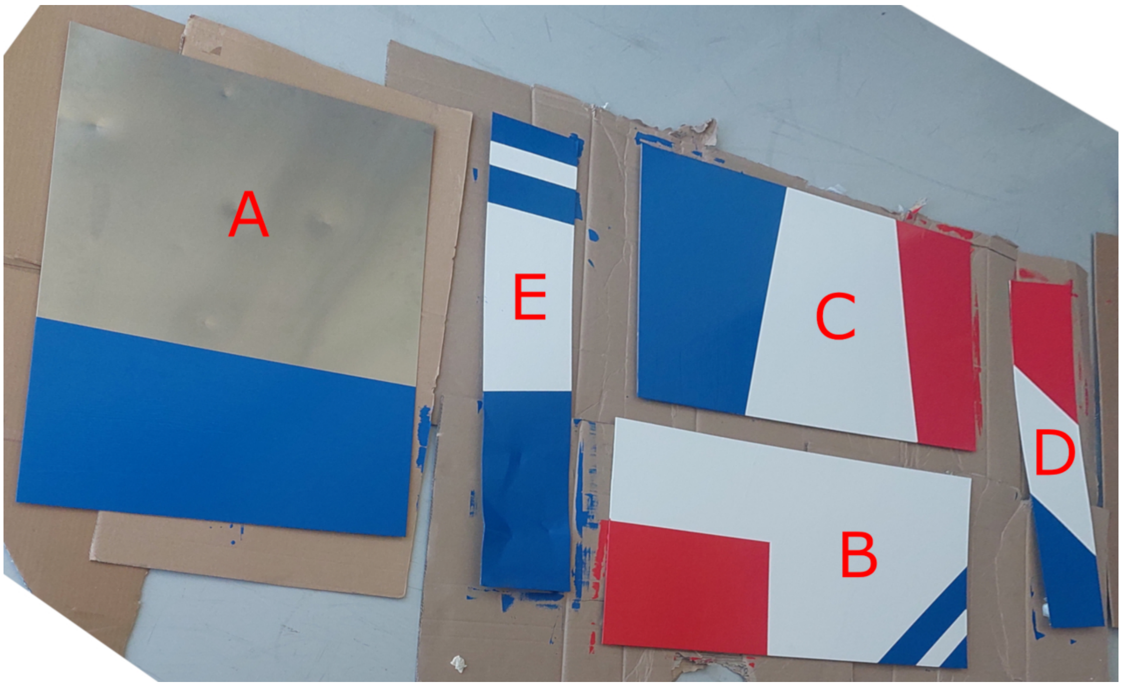


Figure 4.3: Painted metal plates

Part name	Material	Thickness	Paint
Plate A	aluminum	2 mm	blue, unpainted
Plate B	steel	1 mm	red, white, blue
Plate C	aluminum	1.5 mm	red, white, blue
Plate D	steel	0.5 mm	red, white, blue
Plate E	aluminum	0.5 mm	white, blue

Table 4.2: Characteristics of the metal plates

4.1.3 Capturing Setup

Large and deep dents are very noticeable and would be found quickly by any inspection staff, while finding smaller, less apparent dents is much more time-consuming, so automating this step has more potential for reducing inspection times. These dents are difficult to see or capture in photographs because they are only visible in some lighting conditions. Integrating a light source into the capture setup is therefore essential.

There are two main ways of using light to make dents more visible. The first is having a light source at a very oblique angle to the surface, creating additional diffuse reflections, which make dents appear darker on the side close to the light source and lighter on the side further away. Figure 4.4 (a) shows this case. Here, the direction of incoming light is from the right, making the right side of the dent appears darker due to the large angle between its surface normal and the direction of incoming light. The left side of the dent appears brighter because the angle between the incoming light and the surface normals in this area is smaller. This lighting setup has the potential to make dents in a large area more visible, but would require careful control of the position of the light source. The light source would need to be placed close to the aircraft surface and oriented in the correct direction to achieve the oblique angle required for detecting shallow dents. However, positioning equipment very close to the inspection target increases the chance of collision and damage, especially for an automatic system.

The second option makes use of specular reflections, which are the bright white reflections that appear on shiny or metallic surfaces. The direction of these reflections is also dependent on the surface normal as well as the viewing direction. These reflections have the shape of the light source: a round flashlight will cause a round specular reflection, a long light strip will cause a long line-shaped one. When the inspection surface has a deformation, the shape of the specular reflection in this area will be distorted. For example, when a line-shaped specular reflection is near a dent on an otherwise flat surface, the line will appear to bend around the dent, as depicted in Figure 4.4 (b). This option was chosen because even very subtle deformations will cause deviations in the specular reflection shape, making it a good option for detecting shallow dents.

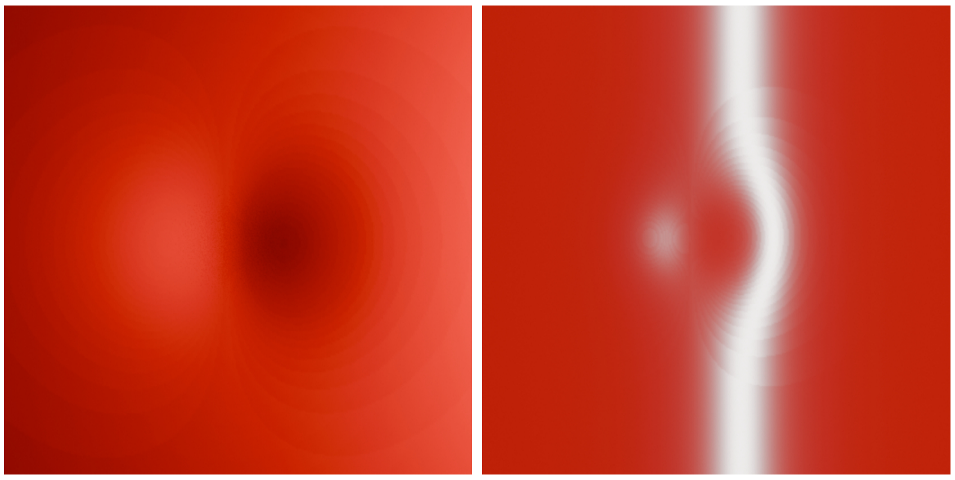


Figure 4.4: Lighting options with (a) diffuse reflections caused by an obliquely angled light source (b) specular reflections caused by an elongated light source near the camera.

Some images were captured with only this light source, some additionally had other light sources, such as ceiling lights and direct or indirect sunlight. Including a variety of other light sources is intended to model a wider variety of real-world conditions.

Detection will only be possible when the dent is very close to the specular reflection in the camera view, so the light source should be elongated enough to cover most of the height or width of the image. The light source should also consist of one or more very narrow bands, since a small deviation in a narrow line is more obvious than in a wider line, at least to the human observer. In principle, a single line of light that is moved across a given area while images are being captured is sufficient to detect dents in that area, but using multiple parallel lines would speed up the process, since fewer images are needed to cover the area. A single line-shaped light was chosen here as a proof of concept, but the use of multiple parallel lines may provide additional information helpful to a neural network and should be explored in the future.

A special LED light strip[1] that produces a continuous line of light was selected because there are no visible gaps between the individual LED lights, producing a continuous line-shaped specular reflection. The light strip was cut to a length of 70 cm and attached to a rigid piece of metal.

The light strip and camera were mounted on a metal base, which can be attached to a tripod or robot arm, as needed. While capturing photographs for the dataset, the camera was mounted on a tripod that could be adjusted in height and angle, or on a small metal base when capturing images near the floor. To photograph different areas, the inspection parts were attached vertically to a rolling platform. Figure 4.5 depicts one of the setups used to capture images.

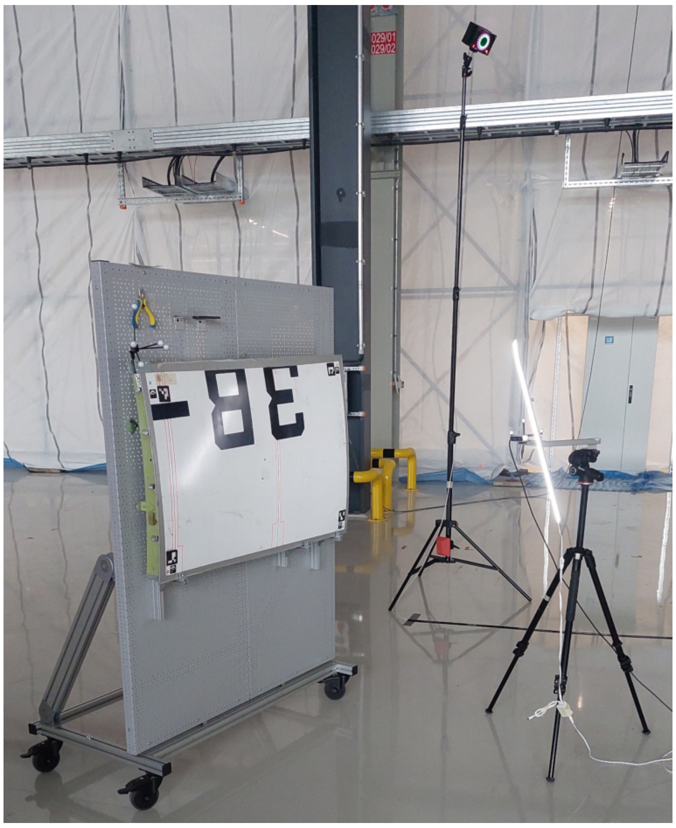


Figure 4.5: Setup used to capture images in the hangar at the DLR.

Two different cameras were used to capture images: a Logitech C505e HD webcam [2] and an Intel RealSense D435i depth camera [3]. The RealSense camera was used for its ability to take more close-up captures; no depth information was used.

[1]https://de.paulmann.com/p/maxled-1000-led-strip-basisset-cob-1-5m-tunable-white-15-5w-1200lm-m-tunable-white-40va/71114

[2]https://www.logitech.com/en-us/products/webcams/c505e-business-webcam.960-001385.html

[3]https://www.intel.com/content/www/us/en/products/sku/190004/intel-realsense-depth-camera-d435i/specifications.html

4.1.4 Tracking-based Annotation

Because the visibility of dents in captured images is highly dependent on the exact lighting conditions and angles between the camera and the dented surface, it is difficult to annotate such images at a later time; the exact size and shape of a dent are difficult to estimate when the only visible indicator is a curved line of reflected light. To combat this difficulty and produce a dataset with annotations of consistent size and location, a new tracking-based annotation method was developed and tested. The method makes use of an optical tracking system and retroreflective markers to localize the camera, aircraft part, and outlines of dents.

Tracking System and Targets

In this work, two different ARTTrack[4] systems were used. The first system consists of 8 cameras mounted to a rack suspended from the ceiling of a laboratory at the DLR. This room was fitted with blackout curtains and external blinds, allowing for control of external light sources. The second tracking system, consisting of 4 cameras, was newly set up in the hangar to allow for a much larger tracking volume, which was needed to accommodate the size of the wing section. In this location, external light sources from the windows, ceiling lights, and adjacent rooms could not be blocked, and more objects are visible in the background.

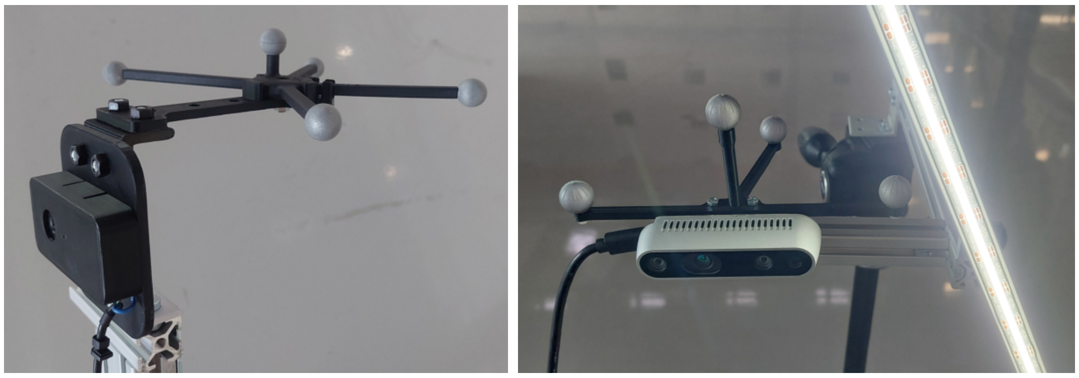


Figure 4.6: Targets attached to the cameras (a) Logitech (b) Intel RealSense.

The tracking systems can be used to track the location of individual retroreflective markers, or the location and orientation of a "target" consisting of multiple markers at fixed positions. Three targets were used here, one attached to the camera, one attached to the metal plate or plane part to be photographed, and a measurement tool target. Figure 4.6 shows how targets were attached to each of the cameras used. The targets are positioned behind the camera itself to make room for the light strip. The targets are oriented so the attachment structure does not obscure the view of the tracking cameras, which are located near the ceiling. The measurement tool (Figure 4.7) is unique, because the tracking system can be calibrated to output the exact location of

[4]https://ar-tracking.com/en/product-program/arttrack5

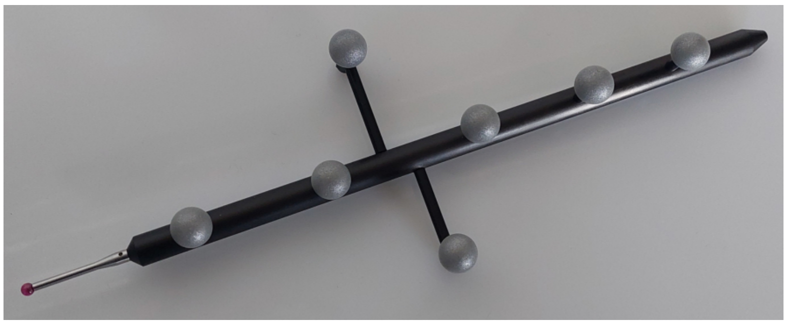


Figure 4.7: Measurement tool with attached targets.

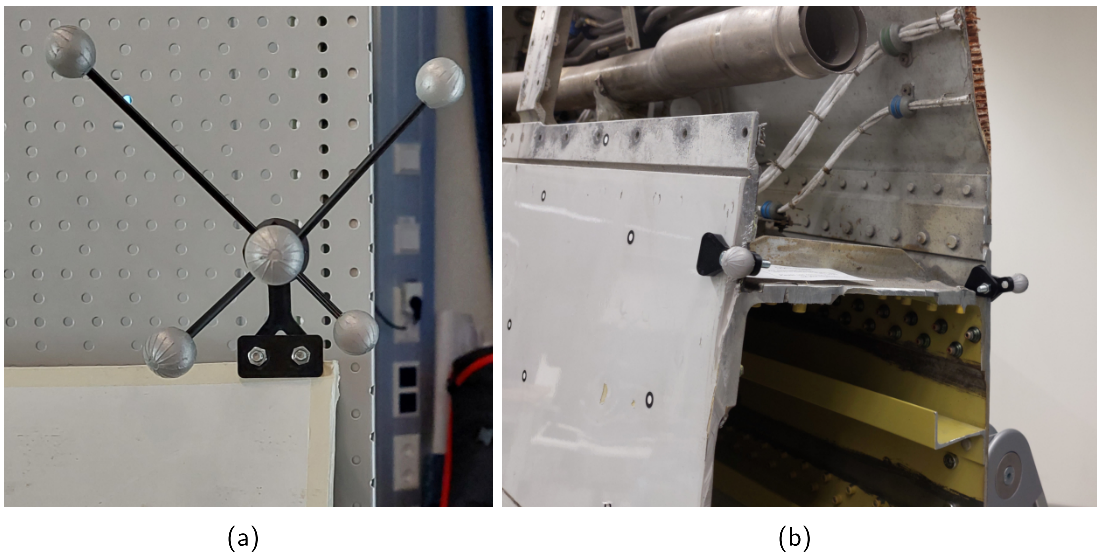


Figure 4.8: (a) Prefabricated target and (b) 2 of the individual markers attached to plane parts.

the tip of this tool; other targets are usually calibrated to have their location defined as one of the retroreflective markers. This means the measurement tool can be used to determine the point in space where the tool tip is located, making it possible to trace the outline of a dent with the tool to measure its location in space. A few different targets were attached to the metal plates and plane parts such as the target in Figure 4.8 (a). On the larger plane parts, using only the prefabricated targets would result in localization errors being larger for measurements taken farther away from the target. To reduce these errors, an additional retroreflective marker was attached to the part at the edge farthest away from the primary target. This set of markers can be recalibrated as a new target and used as normal. Having larger distances between the markers of a target increases the accuracy of the orientation of the part. On the largest plane part, Part A, only individual markers were attached, as shown in Figure 4.8 (b).

Coordinate Systems

If the relative location of the dent and the camera are known at the time an image is captured, this information can be used to project the coordinates of the dent from the 3D world coordinate system to the 2D image coordinates, providing annotation data. To calculate this relative position, the transformations between the coordinate systems of the three tracked targets must be known, as well as the transformation between the camera target's coordinate system $\mathrm{CS}_{\mathrm{camera}}$ and the camera's optical coordinate system $\mathrm{CS}_{\mathrm{optical}}$. The camera's intrinsic parameters are also required to accurately project to 2D image coordinates.

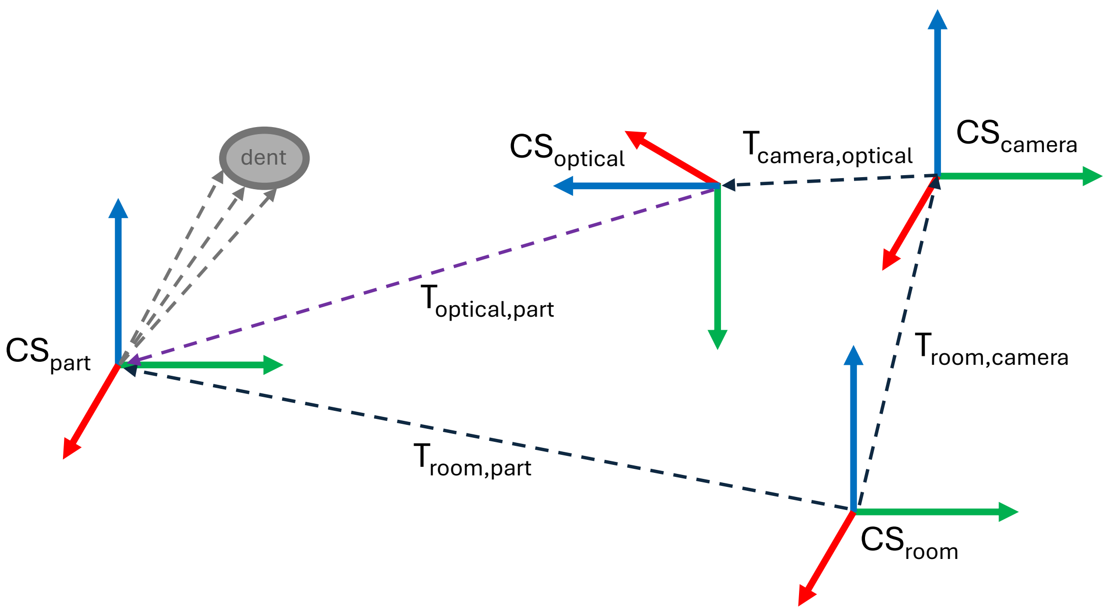


Figure 4.9: Schematic overview of the transformations between different coordinate systems used to determine the location of a dent.

Figure 4.9 shows a schematic overview of the relations between the different coordinate systems used. The tracking system was set to output the locations of the camera and part targets relative to its base coordinate system $\mathrm{CS}_{\mathrm{room}}$ and the measurement tool's location relative to the part target's coordinate system $\mathrm{CS}_{\mathrm{part}}$. This means any measurements made with the tool are defined in $\mathrm{CS}_{\mathrm{part}}$, and must be transformed to the desired coordinate system. This choice was made so any measurements made by the tool are defined in relation to the part, meaning the part could be freely moved around. The transformation from the tracking system's base coordinate system $\mathrm{CS}_{\mathrm{room}}$ to the part target's coordinate system $\mathrm{CS}_{\mathrm{part}}$ is $\mathrm{T}_{\mathrm{room,part}}$, and the transformation from $\mathrm{CS}_{\mathrm{room}}$ to the camera target's coordinate system is $\mathrm{T}_{\mathrm{room,camera}}$. With these locations known, the transformation directly from the camera's target to the part's target can be calculated as $\mathrm{T}_{\mathrm{camera,part}} = \mathrm{T}^{-1}_{\mathrm{room,camera}}\mathrm{T}_{\mathrm{room,part}}$. The last transformation needed is from $\mathrm{CS}_{\mathrm{camera}}$ to the camera's optical coordinate system $\mathrm{CS}_{\mathrm{optical}}$, which is $\mathrm{T}_{\mathrm{camera,optical}}$. This can be determined using a similar procedure used for calibrating the camera, which will be explained later. Once all transformations are known, the direct transformation from the optical coordinates to the part coordinates can be calculated as:

$$T_{\text{optical,part}} = T^{-1}_{\text{camera,optical}} T^{-1}_{\text{room,camera}} T_{\text{room,part}} \tag{4.1}$$

This transformation can be applied to the measured points to provide their locations in the camera's optical coordinate system. From here, the points can be projected to the 2-dimensional image coordinate system using the camera's intrinsic parameters, which are determined during camera calibration.

Camera Localization

The transformation $T_{\text{camera,optical}}$ from the camera tracking target's coordinate system CS_{camera} to the camera's optical coordinate system CS_{optical} is required for the annotation method. This can be determined by using an object whose position and orientation are known in both coordinate systems. The coordinates of an object in the camera's optical coordinate system can be determined through a procedure also used to calibrate cameras. Taking photos of an object with known relative points can be used to determine the relative position of the object and the camera's optical system. A checkerboard pattern is often used for this purpose; if the exact size of one square is known, then the relative locations of all the vertices where corners meet are known. A photograph of the checkerboard pattern can be used to calculate $T_{\text{optical,board2}}$.

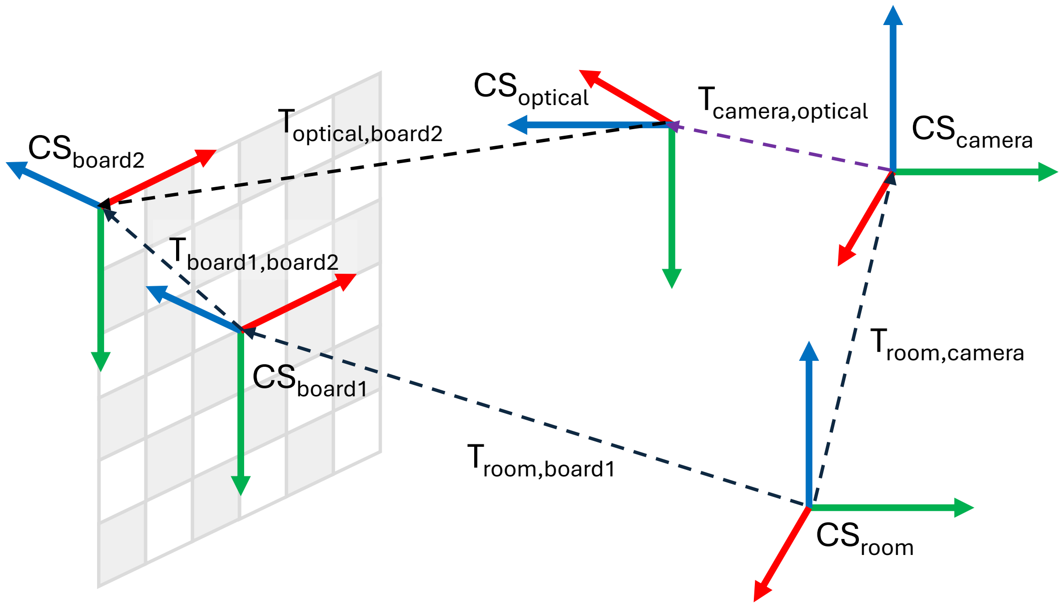


Figure 4.10: Schematic overview of the transformations between different coordinate systems used to determine the offset between CS_{camera} and CS_{optical}.

The tracking system can be used to determine the location of the checkerboard in CS_{camera}. This requires a checkerboard with tracking markers attached and the position of the checkerboard vertices relative to the markers. The coordinate system defined by the tracking markers is CS_{board1}. The algorithm used to calculate $T_{\text{optical,board2}}$ returns the location and orientation of the checkerboard pattern with the origin at one of the outer corners, so the additional transformation between the coordinate system defined by the tracking markers CS_{board1} and the coordinate system CS_{board2} calculated by the checkerboard photograph is also needed. The relevant transformations are shown in Figure 4.10. In this work two different checkerboards were used, the one shown in Figure 4.11 came with a file defining the exact coordinates of the vertices relative to the integrated tracking markers, the other was calibrated manually with the help of the measurement tool. No significant difference in accuracy could be determined between the two boards.

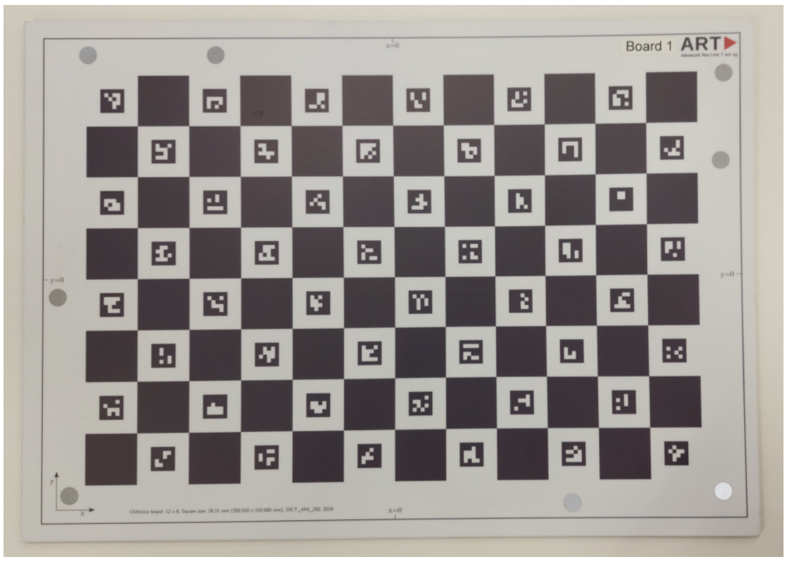


Figure 4.11: ChArUco board with 8 integrated retroreflective markers.

The tracking system was used to determine the exact coordinates of all the targets at the time the photo of the checkerboard was taken, allowing $T_{\text{camera,optical}}$ to be calculated as:

$$T_{\text{camera,optical}} = T^{-1}_{\text{room,camera}} T_{\text{room,board1}} T_{\text{board1,board2}} T^{-1}_{\text{optical,board2}} \tag{4.2}$$

$T_{\text{camera,optical}}$ is static as long as the same camera is used and the attached tracking target remains unchanged. Once calculated, it was saved to a file and later used during the capture of dataset images.

Dent measurement

In order to measure the locations of dents, the part and the light source were positioned so the dent would be visible from the viewpoint of the person, and the targets are visible from multiple of the tracking system's cameras. When targets are visible from only two cameras or are far away, the tracking system's accuracy decreases, so careful positioning of the part and the person are required so as not to obscure the tracking system's view of the targets.

A program was written to receive the location of the measurement tool's tip and save the coordinates to a file. The tool was carefully positioned at one point on the outline of the dent, and a key pressed to start the capture. After a small delay, the coordinates started being received, and the tool was moved to trace the outline of the dent. Once the initial starting position of the tool was reached, the capture would be terminated by another key press. Due to the tool being held still at the beginning and end of the tracing procedure, additional unnecessary points are captured at the beginning and end of the file. These points are also noisier due to the user having to move and press a key. Since no markings were made to indicate the exact starting position, the end of a trace will sometimes overshoot the beginning. These additional points would cause an unintended deviation in the shape of the outline, as well as artifacts when converting the shape to a mask, so they were removed, resulting in a smoother outline. Figure 4.12 shows the measured coordinates of a dent on plane part B with the ends already trimmed.

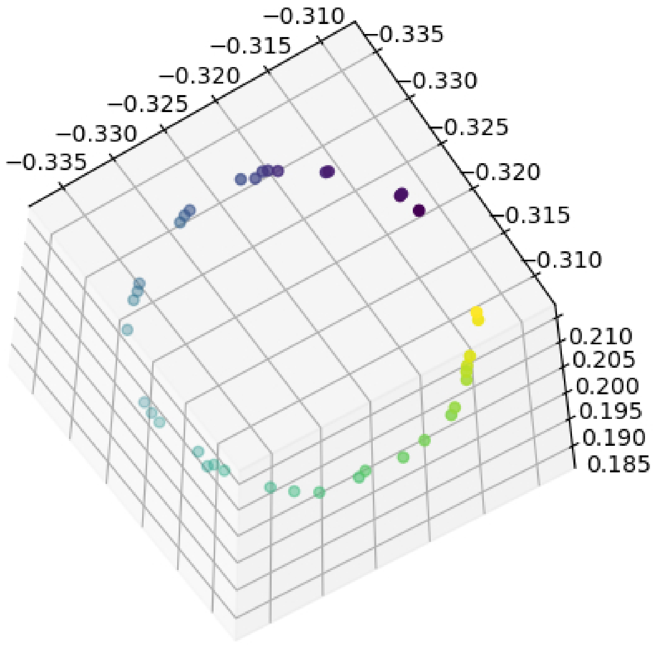


Figure 4.12: Traced outline of dent 2 of plane part B.

Data Annotation

The images for the dataset were captured inside the volume of the tracking system. All objects were stationary at the time a photo is taken so the location of all targets could be determined and the relevant transformation $T_{\text{optical,part}}$ was calculated and saved. Next the coordinates of all dents measured on the current part are transformed from the original coordinate system CS_{part} to the cameras optical coordinate system CS_{optical}. Next the coordinates of each dent are projected to 2D image coordinates and the visibility of the pixels within the image is determined. In the next steps, dents that are only visible at the very edge of the image are excluded, because there is rarely any visible indication of a dent, when it is largely out of view. Initially the threshold for visibility was set to 20 pixels from the image border, after review of the resulting annotations, this threshold was increased to 40 pixels.

The image coordinates of the dent were then used to determine the normalized bounding box coordinates. Since a more precise outline of the dent was already calculated at this point, segmentation masks were also created for each dent. Because the outline of the dent consists of sparse pixels, polygon lines were drawn between the pixels, and the resulting polygon filled with white pixels. The resulting segmentation masks were not used in this work, but are available for all images. Figure 4.13 shows a cropped image of a dent on plane part D with the bounding box annotation in blue and the segmentation mask overlaid in red.

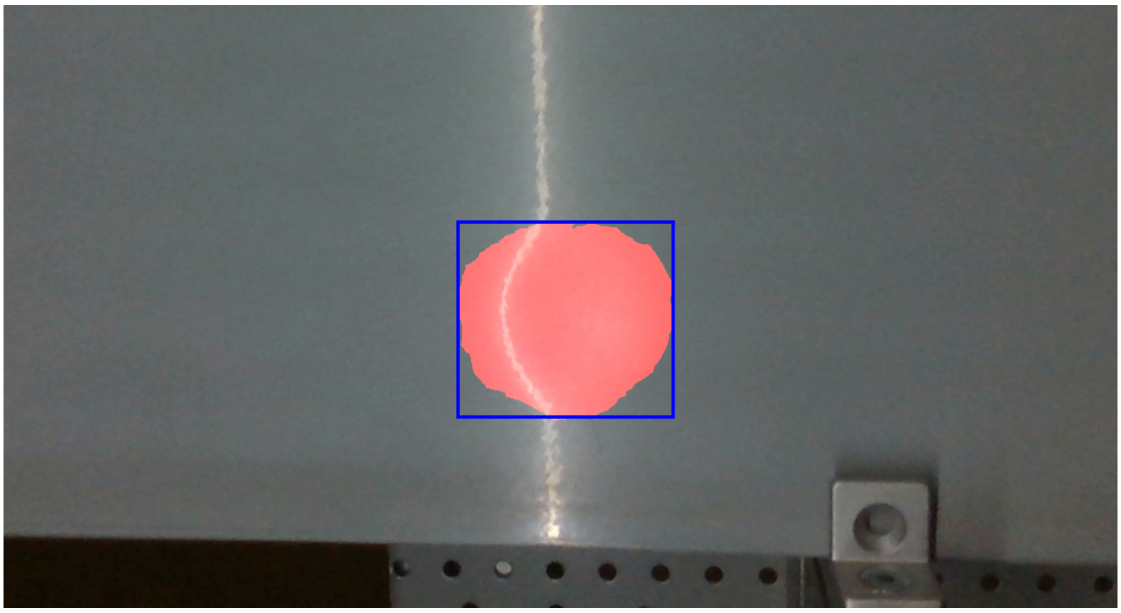


Figure 4.13: Cropped image of a dent on Plane Part D with the bounding box annotation in blue and the segmentation mask overlaid in red.

Review

The captured image annotations were later reviewed for annotation quality and visibility of dents. Some images with annotations that were barely visible along the edge were deleted or had the annotation removed when there was no indication of a dent visible. Some images were removed due to annotated dents not being visible at all, despite not being near the image boundary; this happens when the dent is far away from any specular reflections. As a result, only annotations that are visible to a human and could have been annotated manually are kept. Some images were also removed due to poor annotation quality; these are images where the localization accuracy was very low, resulting in the annotation being offset from the actual location of the dent. Images with medium localization accuracy were kept in the dataset. This decision sacrifices some accuracy in bounding box locations in order to keep the dataset large enough for deep learning. Keeping these images also makes it possible to manually reannotate them later, if needed. Some images were also discarded due to being accidental duplicates or nearly identical.

4.2 Deep Learning

The deep learning component of the robotic inspection setup must be fast enough to process the images coming from a video feed in real time. Most cameras will capture video at 24 or 30 frames per second (fps), so being able to process at this speed would be ideal. A setup with a slower processing speed would also be possible, but only at the cost of also reducing the movement speed of the camera. Small dents may only be visible from a small range of camera positions, so it is important to capture images at very small position intervals. Capturing and processing video at a lower frame rate would mean the speed of the camera movement must also be lowered, which would extend the amount of time needed to fully inspect a large area. There is always a tradeoff between accuracy and speed when it comes to model selection, but new models keep improving accuracy at real-time speed.

Most of the dents in this work have the general shape of a Gaussian distribution and transition smoothly into the surrounding flat surface. This means they do not have clearly defined boundaries, only the general shape and size are visible in an image. For this reason, a bounding box detection approach was chosen for this work instead of segmentation, which is more appropriate for features with clearly visible boundaries.

Due to the need for high speed processing, and the long history of YOLO models performing well at this speed, the newest model of the series, YOLOv12, was the primary model chosen for this work. Due to time constraints, only one other model could be tested, and RT-DETR was chosen, due to its very different architecture.

4.2.1 YOLOv12

The YOLO series of models is well known for its fast speed, and new models are frequently released that further improve accuracy while maintaining speed. YOLO stands for **Y**ou **O**nly **L**ook **O**nce [26], meaning it localizes and classifies objects at the same time, unlike two-stage detectors that handle these properties separately.

The newest model, YOLOv12, was released by Tian et al. [34] in 2025. This model relies on attention to improve accuracy while attempting to keep the computational costs associated with traditional attention low.

They introduce an area attention module, which has a large receptive field but a low computational cost. Area attention divides the feature map into 4 parallel bands, making the receptive field and computational cost one-fourth of the original. The computational complexity is still $\mathcal{O}(n^2)$, but the reduction by a factor of four is significant enough for real-time requirements as long as the image width n is fixed at 640 pixels. Increasing the input size would greatly affect model speed.

To improve optimization, they introduced **R**esidual **E**fficient **L**ayer **A**ggregation **N**etworks (R-ELAN), which are based on the ELAN used in YOLOv7 [36]. This adds a residual shortcut and a scaling factor. Other architectural changes were also made, like introducing flash attention, removing positional encoding, and reducing the number of stacked blocks in the last stage of the backbone.

5 different variants of the model are available and the largest variant with the highest performance, YOLOv12-X, was chosen for this work.

4.2.2 RT-DETR

Another approach to object detection are transformer-based models. The original **DE**tection **TR**ansformer (DETR) was released in 2020 by Carion et al. [4]. The transformers used in DETR variants allow for a high accuracy, but at the cost of speed, making them unsuitable for real-time processing. In 2023, Zhao et al. [42] released **R**eal-**T**ime **DE**tection **TR**ansformer (RT-DETR), which had speed and accuracy competitive with YOLO models.

The DETR architecture, shown in Figure 4.14, consists of a CNN backbone for feature extraction, a transformer-based encoder-decoder, and a feed-forward network (FFN) for prediction. The decoder transforms a set of N embeddings, called object queries, into output embeddings, which are decoded into box coordinates and class labels by the FFN. They use a loss function that employs bipartite matching between predictions and ground truth objects. A matching algorithm calculates the optimal matching between the set of predictions and the set of ground truth objects by calculating the matching cost of combinations using the class labels and box coordinates. The matching function, as well as the loss function, uses the bounding box loss $\mathcal{L}_{box}$ to evaluate the quality of the bounding box. $\mathcal{L}_{\text{box}}$ is a combination of ℓ^1 loss and generalized IoU loss $\mathcal{L}_{\text{iou}}$ and is defined as:

$$\mathcal{L}_{\text{box}}\left(b_i, \hat{b}_{\sigma(i)}\right) = \lambda_{\text{iou}}\mathcal{L}_{\text{iou}}\left(b_i - \hat{b}_{\sigma(i)}\right) + \lambda_{\ell^1}\left|\left|b_i - \hat{b}_{\sigma(i)}\right|\right|_1 \tag{4.3}$$

Here, $\hat{b}_{\sigma(i)}$ are the box coordinates of prediction i, and b_i are the ground truth box coordinates of the object matched with prediction i.

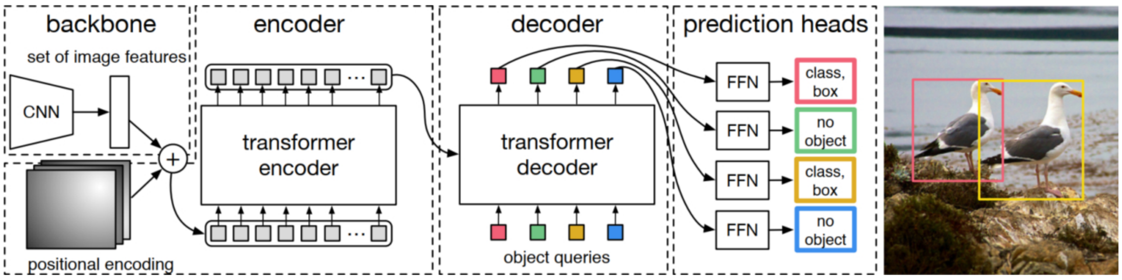


Figure 4.14: DETR architecture from [4]

Building on DETR, Zhao et al. [42] introduced RT-DETR, which has an efficient hybrid encoder and uncertainty-minimal query selection. The RT-DETR architecture, shown in Figure 4.15, consists of a ResNet backbone, the efficient hybrid encoder, and a transformer decoder. The last 3 stages of the backbone are used as inputs for the encoder. The encoder produces queries through decoupled cross-scale feature fusion and intra-scale feature interaction. Uncertainty-minimal query selection is employed to select a specific number of queries to be passed to the decoder. The decoder with auxiliary prediction heads then produces detection categories and bounding boxes. The hybrid encoder consists of two parts: the **A**ttention-based **I**ntra-scale **F**eature **I**nteraction Module (AIFI) and the **C**NN-based **C**ross-scale **F**eature **F**usion Module (CCFF). AIFI is only applied to the last stage, S5, of the backbone, which represents high-level features, thus significantly reducing the computational cost of intra-scale interactions.

The paper shows that performing intra-scale interactions only on stage S5, rather than on all three stages, greatly improves speed and also slightly increases accuracy. CCFF performs the cross-scale feature fusion. Uncertainty-minimal query selection prevents features with low localization confidence from being selected for the set of queries passed to the decoder, increasing performance.

The RT-DETR architecture also allows for later speed tuning without requiring the model to be retrained by adjusting the number of decoder layers. Experiments show that removing a few layers from the end of the decoder only slightly reduces accuracy, while greatly increasing inference speed.

Two different variants of RT-DETR are available; the larger variant RT-DETR-X with a ResNet101 backbone, was selected for this work.

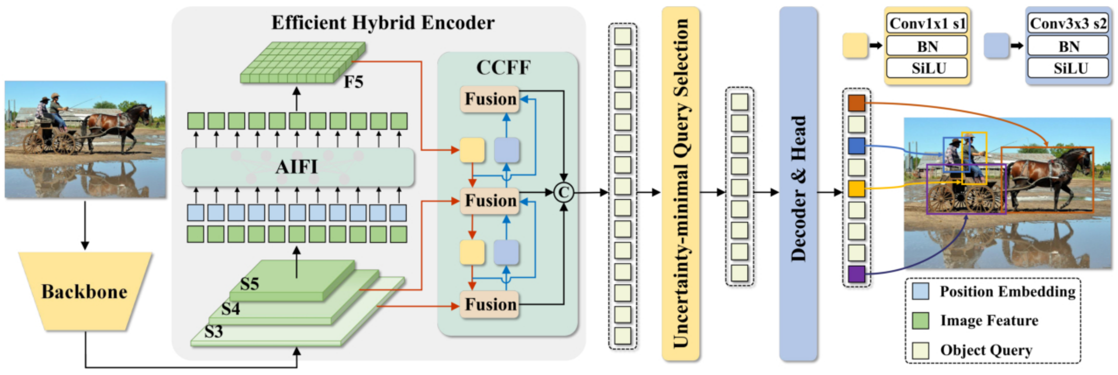


Figure 4.15: RT-DETR architecture from [42].

4.2.3 Experiments

Training

Both models are pretrained on MS COCO val2017 [17]. All images are scaled to a size of 640x640 pixels during data-loading. A batch size of 4 was used for both models. Hyperparameters were tuned for each model to determine the best-performing configuration. AdamW[19] optimizer was used for RT-DETR, RAdam [18] for YOLOv12. The initial learning rate was set to 1e-5 for both models. The performance would plateau within 350 epochs for YOLOv12 and 600 epochs for RT-DETR.

The best version of each model was determined by its fitness on the validation set; fitness was defined as the average of its accuracy, precision, AP_{50}, and $AP_{50\text{-}95}$.

Training was done using an NVIDIA GeForce RTX 4070 SUPER GPU with 12 GB of memory. RT-DETR was trained for a maximum of 600 epochs, which took about 21 hours on dataset A and 43 hours on Dataset B. YOLO was trained for a maximum of 350 epochs, which took about 14 hours on dataset A and 27 hours on Dataset B.

Evaluations

Since neural networks generally require large amounts of training data, it may be beneficial to have a larger dataset, even if the additional images are slightly different. To test this theory, each model is trained on both Dataset A and Dataset B to determine the impact of training on additional images of non-aircraft parts. Since Dataset B functions like a data augmentation on Dataset A, training on these datasets with and without data augmentation was also compared. Each model was trained on the following:

Dataset A	no data augmentation
Dataset A	with data augmentation
Dataset B	no data augmentation
Dataset B	with data augmentation

The performance of each model is evaluated using accuracy, precision, AP_{50}, and $AP_{50\text{-}95}$ on the test set. Additionally, incorrect detections made by the best-performing models are reviewed to qualitatively examine the mistakes.

Speed

To determine the viability of the models for detecting defects on a video feed, each model is tested on the same prerecorded video to determine its speed when processing sequential images.

Robotic Inspection Test

The feasibility of a robotic inspection setup is tested by mounting the camera and light strip on a UR10e [5] robot arm, and having it inspect the entire surface of a plane part. The video is streamed to a computer running the detection model, saving the original video and the video with annotated detections. Both models are tested.

[5]https://www.universal-robots.com/products/ur10e/

Chapter 5

Results

5.1 Dataset

A total of 6,295 images were suitable for inclusion in the dataset, 3224 were of plane parts and 3071 of metal plates. These two types of images were kept separate and split into subsets to allow different training and validation datasets to be created. The images are available at this LINK [1].

Overview

Of the 3224 images of plane parts, 42% (1360) contain dents. There are 1493 instances of dents, meaning some images contain multiple dents. Of the 3071 images of metal plates, 42% (1290) contain dents; there are 2524 instances of dents. Figure 5.1 shows how many dents per image are in both portions of the dataset. Notably, the images of metal plates contain more dents on average as well as a higher maximum number of dents (3 vs. 7). The differences in the dent distributions are mainly due to the fact that the majority of the images of metal plates were taken with the Logitech C505e HD webcam, which must be positioned further away to be in focus, resulting in a larger area being captured in the images. The images taken later, including all images of plane parts, were taken with the RealSense D435i depth camera, which is able to stay in focus at much closer distances. This allowed images to be captured with more variation in the distance between the camera and the part.

Figure 5.2 gives an overview of the annotations of aircraft images. Plot (a) shows the position of the annotation centers, which are clustered near the image center. Plot (b) shows the annotation size relative to the image size. This shows that most annotations have similar length and width, which is to be expected when most dents are circular or only slightly elliptical in shape.

[1]https://mega.nz/folder/PsJVILRY#KZQ2Q4ORJbkR2GgjkNrseA

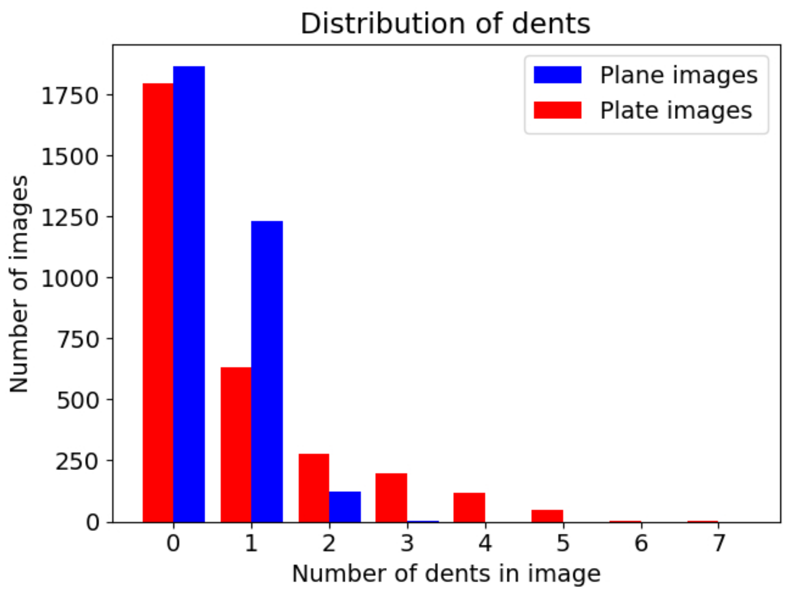


Figure 5.1: Comparison of the dent distribution of plane and metal plate images.

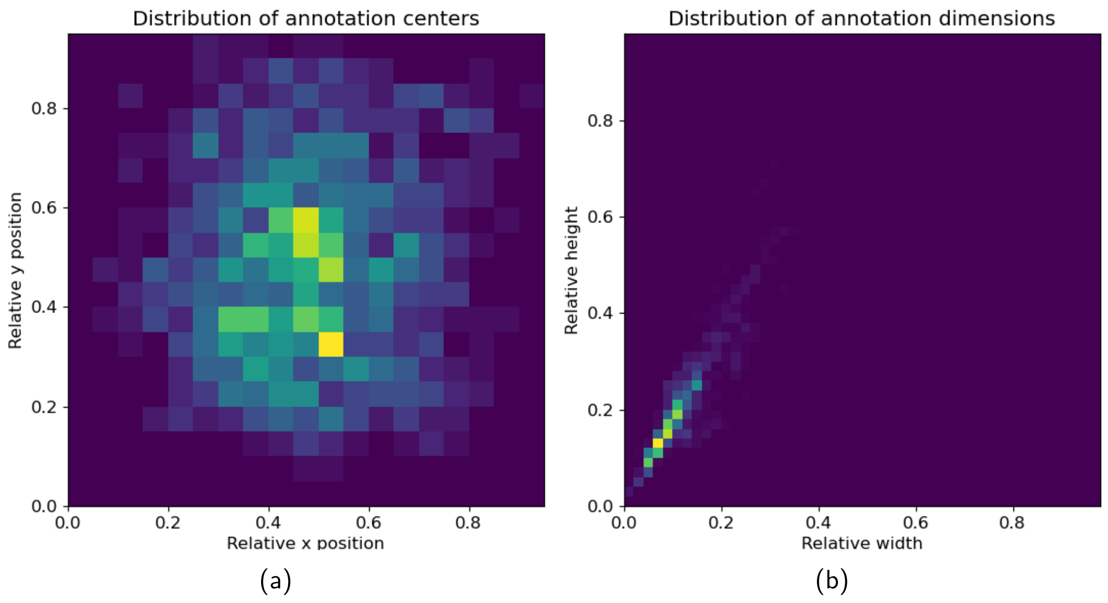


Figure 5.2: Overview of the distribution of annotations of plane images. Figure (a) shows the position of the bounding box centers. Figure B shows the distribution of annotation widths and heights relative to the image size.

Lighting

The dataset contains images with a variety of lighting conditions. The brightness of the light strip and ceiling lights was varied, and windows were sometime covered and uncovered. Figure 5.3 shows an example of the difference between images of the same dent taken with a single light source vs. multiple. Image (a) was taken with only the light strip, all other lights were turned off, and windows were covered. This results in more contrast between the specular reflection and the surrounding areas, and also areas further away from the light source being darker. Image (b) was captured in the afternoon with uncovered windows, resulting in a large amount of ambient light from the windows, that is additionally reflected by the white walls of the room. This results in greater and more even illumination of the entire part, and slightly less contrast with the specular reflection of the light strip. In this particular image, the part was also positioned in a way that shows specular reflections of the windows. The straight lines and right angles that are often present in the reflections of windows can reveal dents in the same way the reflection of the light strip does: the otherwise straight edges of the windows will bend away from indentations. Dents that are too far away from the light strip's reflection to be seen, may still be detectable when near window reflections.

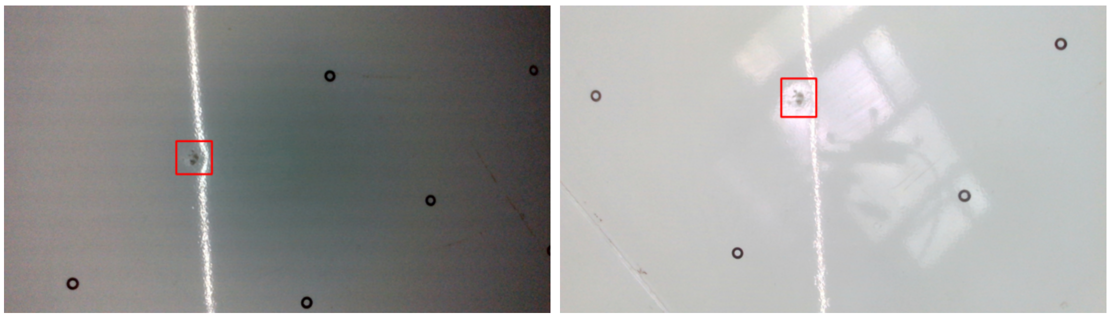

(a) (b)

Figure 5.3: The same dent lit by (a) only the light strip (b) additionally indirect sunlight

Positioning

Images were captured at different distances between the camera and the part. Even when all light sources remain at the same intensity, an image captured at a greater distance will be darker than a close-up capture because the primary light source, the line-shaped light, is also further away from the part, resulting in less intense illumination; Figure 5.4 shows the difference between photos taken with the camera near the part (a) and further away (b).

The parts themselves could not be easily rotated, so the camera was tilted in different orientations throughout the capturing process to acquire images with different orientations of the dents and other features. The line light was originally mounted in a fixed vertical position, so the orientation of its specular reflection is also vertical in the majority of images. Later, the light was mounted differently to allow it to be rotated slightly, allowing images with different orientations of its reflection to be captured,

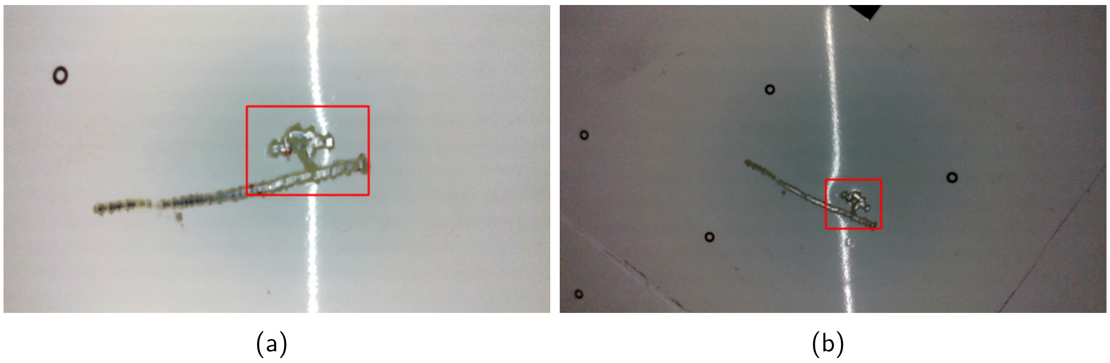


Figure 5.4: The same dent at different distances. The camera was near the part during the capture of photo (a) and further away for photo (b), resulting in a darker image.

increasing the diversity of the dataset. Figure 5.5 shows the same dent captured with different camera and light orientations. When two dents are very close to each other, it was always attempted to capture images containing only one dent as well as images with both dents visible. Images with multiple dents required adjusting the position and orientation of the camera so that all dents in the image are visible through specular reflections.

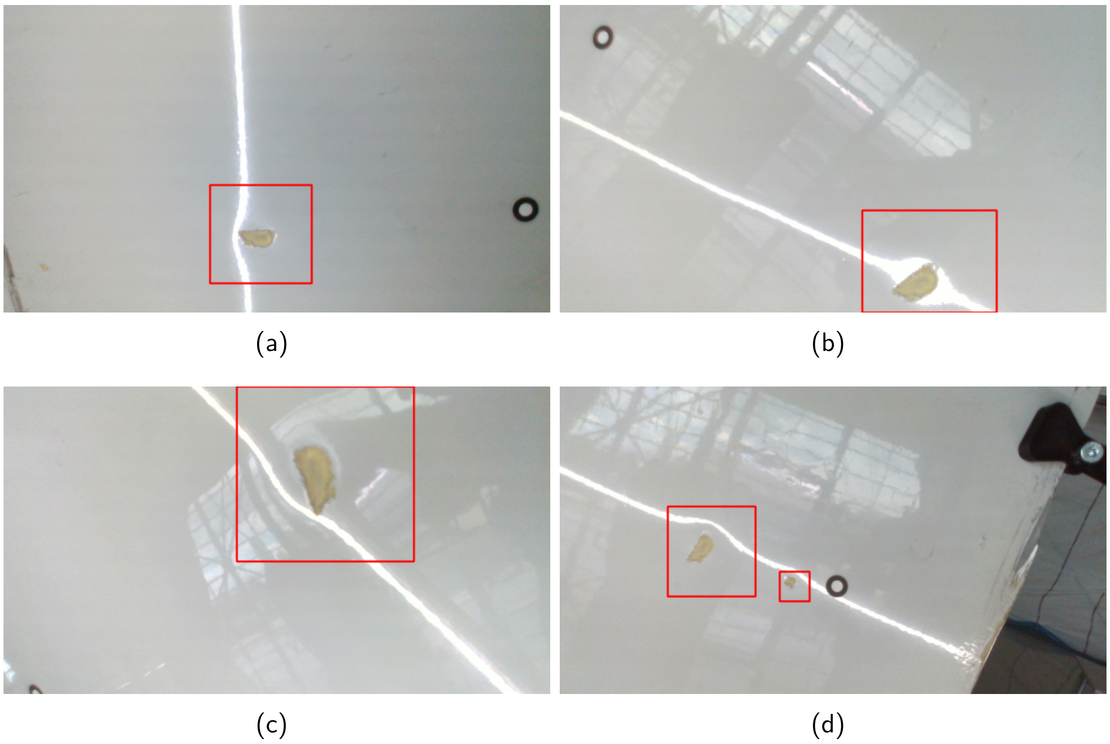


Figure 5.5: The same dent in different orientations.

Features

The dataset contains many images of features commonly found on aircraft. Normal features like screws, rivets, joints, vortex generators, and access panels are included. Many images of these were taken, including images depicting how specular reflections appear near and on these features. The main surface of the aircraft skin is often slightly warped near the edges of panels or near attached parts, resulting in a deviation in the shape of the specular reflection. Rivets can also cause ring-shaped indentations in the paint that are visible in specular reflections. Capturing many examples of these is important so neural networks can learn to differentiate between deviations caused by dents and deviations caused by other features, reducing the likelihood of false positives. Other defects like missing paint and missing screws, as well as some dirt, were also present on some of the parts and captured. Figure 5.6 depicts examples of some of the features present in the dataset.

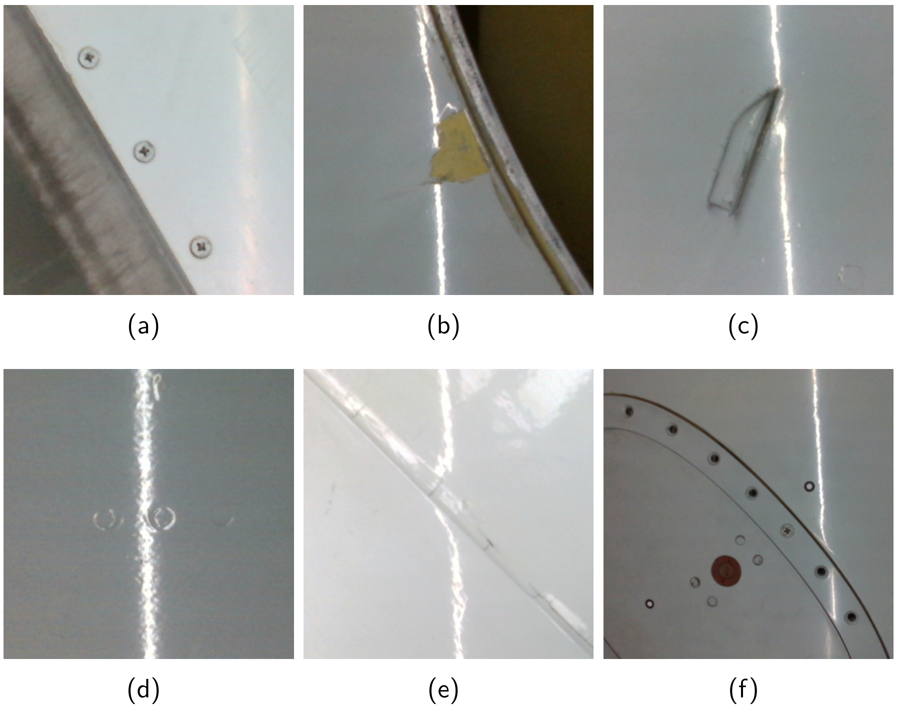


Figure 5.6: Features present in the dataset: (a) screws (b) missing paint (c) vortex generator (d) line of rivets (e) joint in the skin (f) access panel with missing screws. Images are cropped to show detail.

There is some variety in the appearance of dents depending on how much additional damage was caused by the impact. Figure 5.7 shows some of the visual features of dents in the dataset. Image (a) shows a dent with severe damage that removed all layers of paint and also scratched the metal underneath. Image (b) shows a dent where the upper layers of paint are missing, revealing the yellow paint underneath. Images (c) and (d) show more minor scratch-like damage to the glossy coating and upper layer of paint that does not reveal other paint colors underneath. Some dents, like the one depicted in image (e), have some dirt on the impact site, while others show no dirt or damage at all, like in image (f). The vast majority of dents on the plane parts are round in shape, but some, like the one in image (c), are more elongated. Despite the small number of available parts, a decent variety in the size and appearance of dents is present in the dataset.

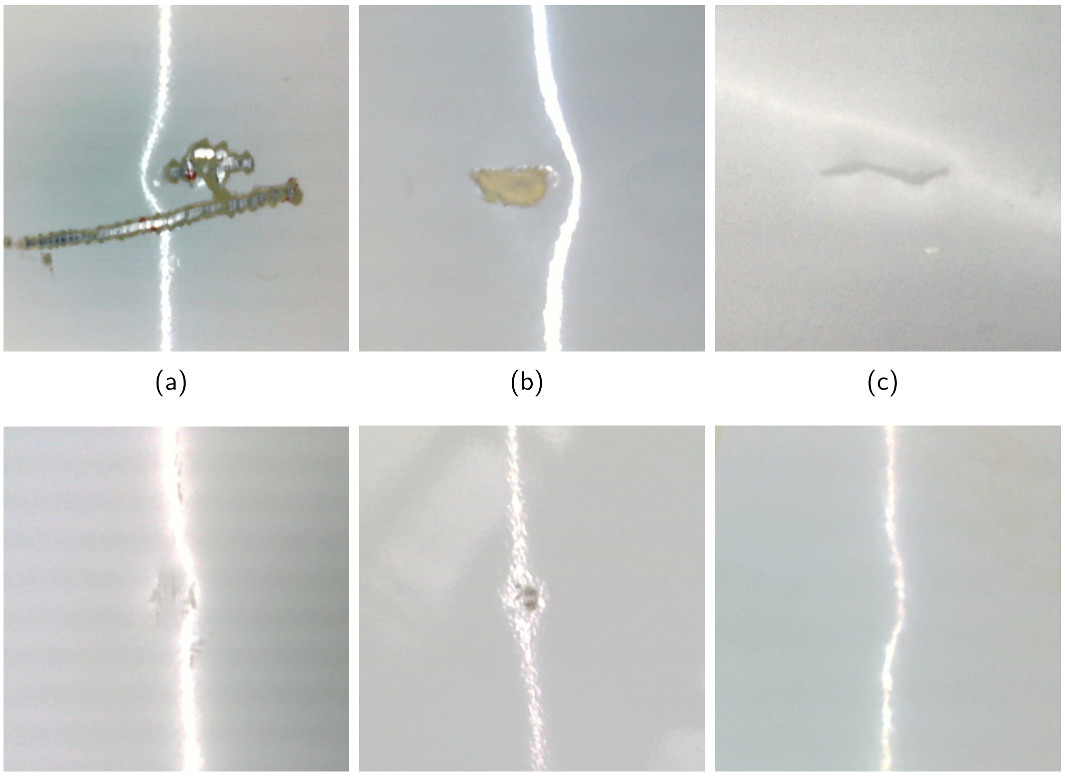

(a) (b) (c)
(d) (e) (f)

Figure 5.7: Different appearances of dents. Images are cropped to show detail.

The plane parts used for the dataset have different surface specularities. Figure 5.8 (a) shows a dent on a highly specular surface. Here, the specular reflection of the light strip is very bright and has crisp borders. Figure 5.8 (b) shows a dent on a less specular surface. The specular reflection on this surface is much more blurred and less bright and does not have well-defined borders. These images depict the range of specularity present in the dataset.

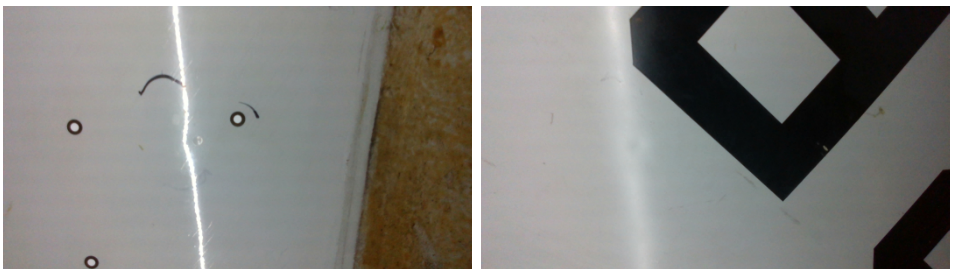

(a) (b)

Figure 5.8: Differences in surface specularity (a) high specularity (b) low specularity.

Other Features

Due to the small number of plane parts available for the dataset, and some of these previously being used in other experiments at the DLR, other features that are not common to aircraft are also represented in the dataset. For example, two of the aircraft parts have small, black, ring-shaped stickers scattered across the entire surface, as shown in Figure 5.9 (a) and (b). Additionally, two of the parts also have april tags and lines or arrows drawn with markers on them. Most parts also have tape around the outer edges. In some images, the tracking target or structures used to mount the part upright are also visible, such as in Image (d). These features are present in a large number of images.

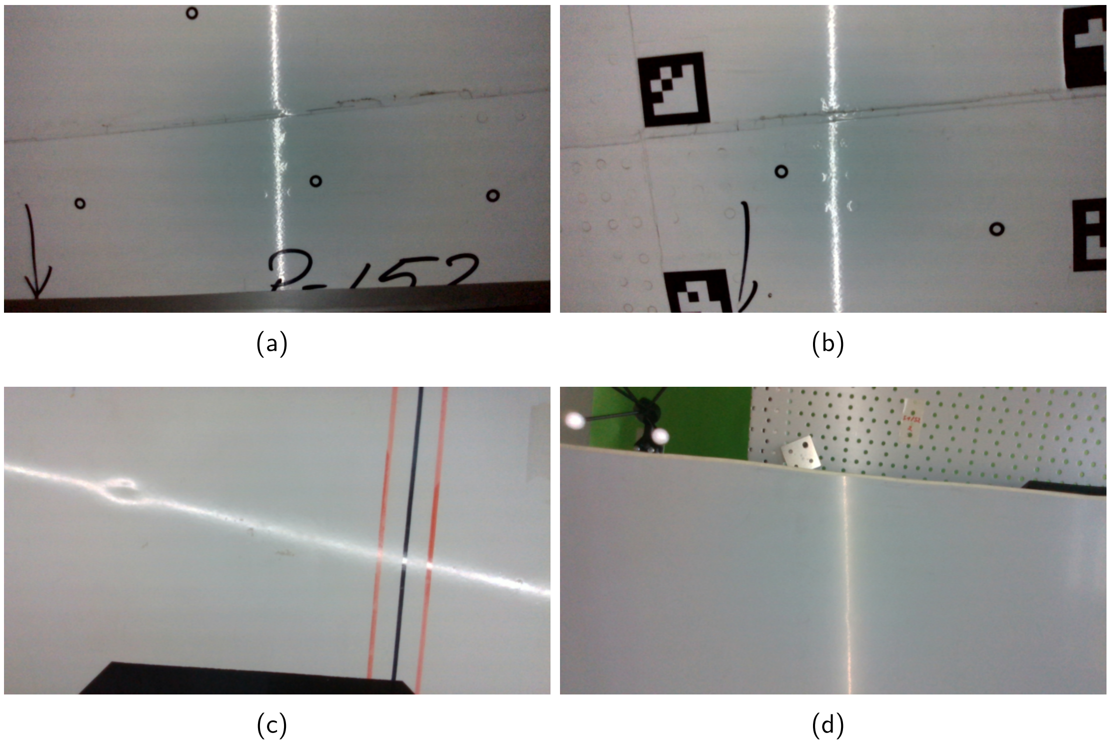


Figure 5.9: Other features common in the dataset.

Metal Plates

Figure 5.10 shows some of the images of metal plates. Image (a) shows a dent on an area painted blue, (b) shows a dent near the boundary of blue and red paint. Image (c) shows the specular reflection line near two areas of missing paint. Image (d) shows how dents appear on an unpainted metal surface. The plate images contribute more variety in surface colors, as well as more dents near the boundaries of different colors. These images also show how the surface texture of the painted areas differs from real aircraft surfaces, being much bumpier and uneven due to the painting process. Many of the dents on the metal plates also appear to be deeper than the dents on plane parts, likely due to the much lower thickness of the plates.

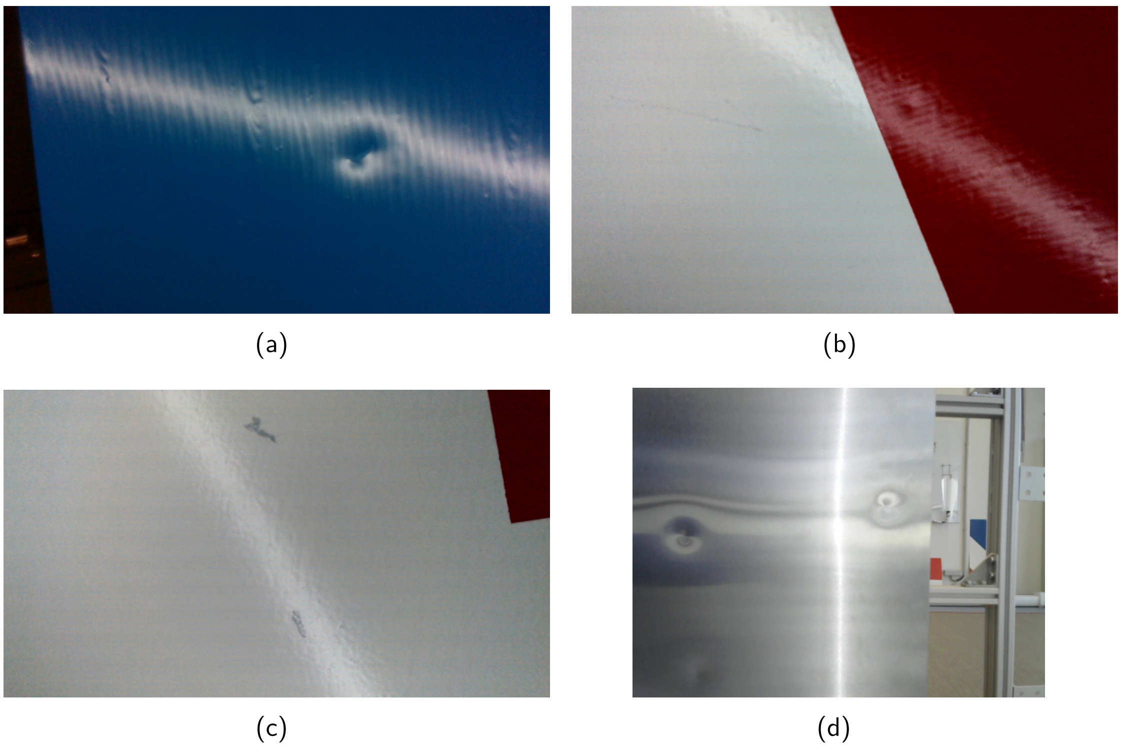


Figure 5.10: Metal plate images, featuring different colors and unpainted surfaces.

Dataset Split

The images of real aircraft parts were split into three subsets for training, validation, and testing with a ratio of 60:20:20. Table 5.1 shows the exact number of images in each subset. The images of metal plates were also split to allow additional validation on these images. No images of metal plates are meant for testing purposes, since these do not accurately represent the appearance of aircraft parts. Table 5.2 shows the exact number of images in each subset.

The images of plane parts and metal plates are saved separately to allow the creation of two different datasets: Dataset A containing only real airplane parts and Dataset B containing both plane parts and metal plates. Dataset A has only realistic images but provides less training data, only 1934 images. Dataset B has about double the training images (3981), but may not accurately represent the appearance of real dents on aircraft surfaces.

	Total	Training	Validation	Testing
Percent	100%	60%	20%	20%
Images	3224	1934	645	645

Table 5.1: Plane images split.

	Total	Training	Validation
Percent	100%	66.7%	33.3%
Images	3071	2047	1024

Table 5.2: Plates images split

5.2 Deep Learning Results

5.2.1 Model and Dataset

Quantitative Results

The overall performance of both models is very good, with both achieving at least 99% precision and recall, which is very high compared to other works on aircraft dents.

Table 5.3 shows the results for the YOLOv12 experiments. The results for this model were mixed. Training on Dataset B resulted in slightly lower precision, while recall and AP_{50} were higher. This result was found when training both with and without additional data augmentation. The highest $AP_{50\text{-}95}$ was achieved on Dataset B with additional data augmentation. Determining which of the datasets is more suitable for YOLOv12 depends on which evaluation metrics are more important for a specific use case. The use case here is detecting the location of defects on aircraft surfaces, which would later be measured and evaluated by a professional to determine the need for repairs. In this scenario a high recall rate is extremely important, since all dents need to be found and documented. A slight drop in precision would result in more false positives, which would result in more areas of the aircraft being needlessly examined. High AP values are beneficial for accurately estimating the size of a dent, but the exact position and size are less important than finding all dents.

Model	Dataset	Set	Precision	Recall	AP_{50}	$AP_{50\text{-}95}$
YOLOv12-X	A	test	0.935	0.889	0.950	0.563
YOLOv12-X	B	test	0.914	0.902	0.952	0.573
YOLOv12-X	A + aug	test	**0.997**	0.980	0.992	**0.765**
YOLOv12-X	B + aug	test	0.990	**0.997**	**0.994**	0.746

Table 5.3: YOLO test set comparisons

Table 5.4 shows the results for the R-DETR experiments. The model trained on Dataset B with additional data augmentation performed best overall. The test on Dataset A with data augmentation achieved the same recall and AP_{50}, and only slightly lower precision. For this model, training on Dataset B resulted in greater or equal AP_{50} and $AP_{50\text{-}95}$ both with and without additional data augmentation. RT-DETR benefits from the inclusion of metal plates in the training data.

Table 5.5 shows the results of both models on both datasets with data augmentation. The highest precision, 100%, was achieved by RT-DETR on dataset B. YOLOv12 had the highest recall when trained on Dataset B and the highest $AP_{50\text{-}95}$ when trained

Model	Dataset	Set	Precision	Recall	AP_{50}	$AP_{50\text{-}95}$
RT-DETR	A	test	0.970	0.948	0.947	0.627
RT-DETR	B	test	0.956	0.928	0.953	0.640
RT-DETR	A + aug	test	0.997	**0.993**	**0.995**	0.735
RT-DETR	B + aug	test	**1.0**	**0.993**	**0.995**	**0.753**

Table 5.4: RT-DETR test set comparisons

on Dataset A. RT-DETR also had the highest AP_{50}, regardless of dataset. Once again, determining the best model is dependent on the use case, since no model performed best on all metrics. With recall being so important in this case, YOLO is slightly better. Both models achieved very high performance, with only two false negatives by RT-DETR and one false negative and 3 false positives by YOLOv12 on the test set. The performance difference between the models is very small, so both are good options for the task of dent detection.

Model	Dataset	Precision	Recall	AP_{50}	$AP_{50\text{-}95}$
RT-DETR	A + aug	0.997	0.993	**0.995**	0.735
RT-DETR	B + aug	**1.0**	0.993	**0.995**	0.753
YOLOv12-X	A + aug	0.997	0.980	0.992	**0.765**
YOLOv12-X	B + aug	0.990	**0.997**	0.994	0.746

Table 5.5: YOLO and RT-DETR test set comparisons

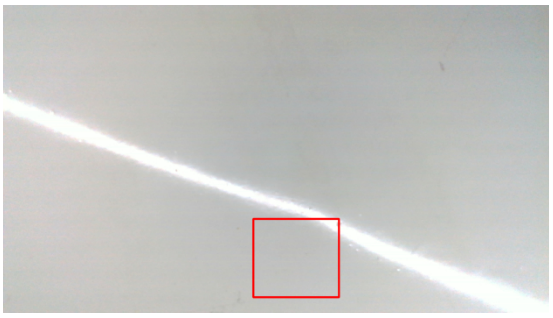
Qualitative Results

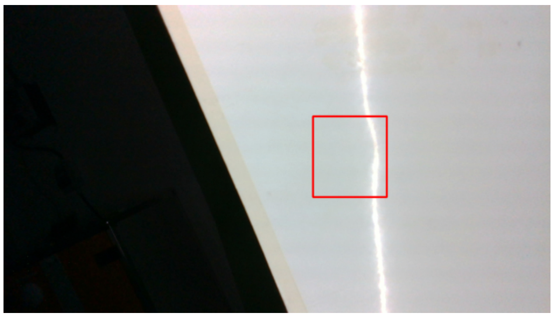
To gain a more thorough understanding of the results, the false positives and false negatives of the models trained on Dataset B were examined. Since the number of incorrect classifications on the test set was so small, the validation set was also examined. Table 5.6 shows the number of false positives and negatives for each model and dataset split.

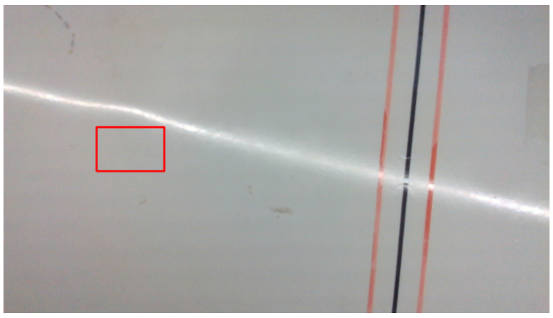
Model	Dataset Split	False Positives	False negatives
YOLOv12	Test	3	1
YOLOv12	Validation	4	3
RT-DETR	Test	0	2
RT-DETR	Validation	1	3

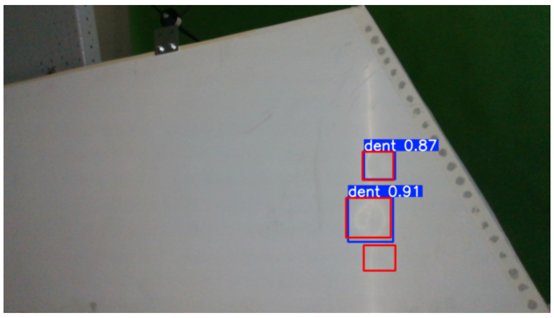
Table 5.6: Overview of detection mistakes by both models on the test and validation sets.

Figure 5.11 shows the missed detections of YOLOv12; the ground truth bounding boxes are depicted in red, and any predictions in blue. Image (a) is the only one from the test set. Images (a), (b), and (c) depict a high-contrast specular reflection with a very subtle curve near the dent. The specular reflection in Image (d) has very low contrast; the dent that was missed is nearly invisible in the image. These images show that YOLOv12 has some difficulty detecting dents when the curve of the specular reflection is very slight.


(a)


(b)


(c)



(d)

Figure 5.11: Examples of missed predictions by YOLO. Image (a) is from the test set, (b), (c), and (d) are from the validation set. The predicted bounding boxes are shown in blue, the ground truth bounding boxes in red.

Figure 5.12 shows some of the false positives produced by YOLOv12. Images (a) and (b) are from the test set, (c) and (d) from the validation set. Image (a) has a false detection in the faint specular reflection of a window. This reflection reveals a slight unevenness in the broader area. Subtle surface deviations like these are often caused by rivets. Rivets are located adjacent to the joint between surface plates, which is visible as a dark line in the upper left of the image. Rivets are a probable cause of this false positive. Image (b) shows a very small irregularity in the shape of the specular reflection line. The surface in this area likely has a small deviation, the cause of which is unknown. Image (c) shows a false positive on an elliptical area of dirt near the specular reflection. Image (d) shows a correct detection of the dent on the left side of the specular line, as well as a false positive on the lower right side of the curve in the line. This detection has a relatively low confidence, but is still above the threshold of 0.25 for detection. Increasing the confidence threshold would reduce false positives like these but would also cause more false negatives. These images show that YOLOv12 is vulnerable to false positives when subtle surface deviations or dirt are visible in images.

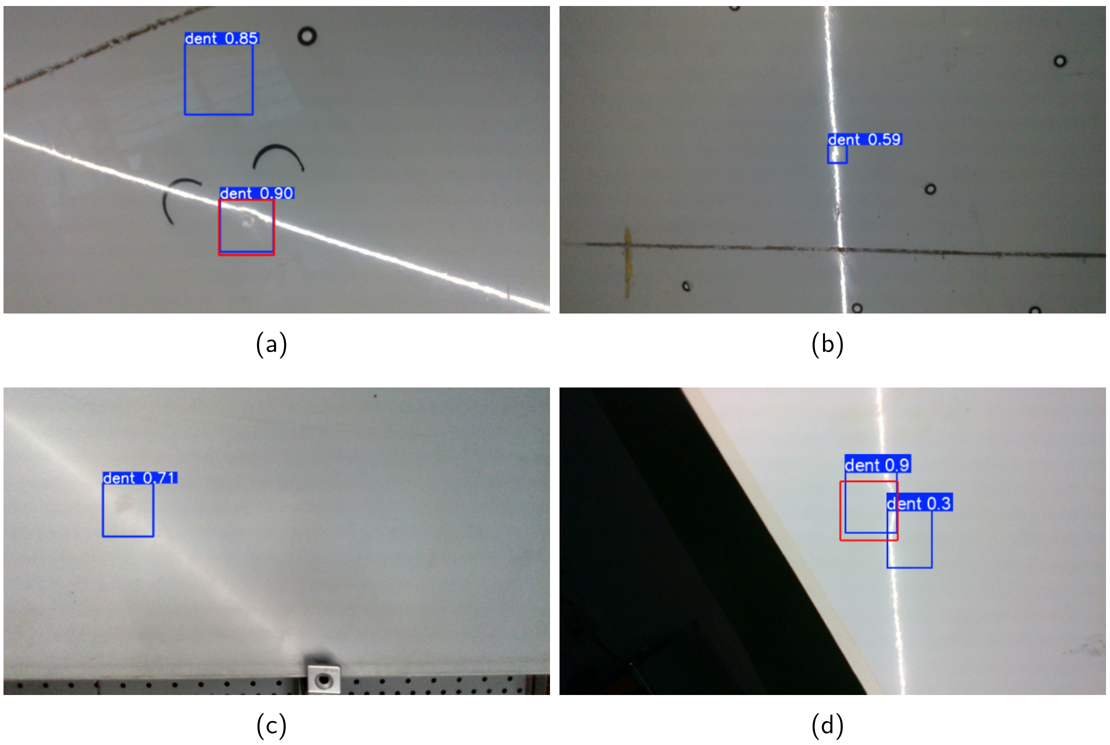


Figure 5.12: Examples of false positives by YOLO. Images (a) and (b) are from the test set, (c) and (d) from the validation set. The predicted bounding boxes are shown in blue, the ground truth bounding boxes in red.

Image 5.13 shows all detection mistakes made by RT-DETR. Images (a) and (b) are from the test set, (c) through (f) from the validation set. Image (a) shows the missed detection of a dent made visible by the specular reflection from a window, not the light strip. Images (b) and (c) show missed detections of the same dent made by the impact gun. Only a subtle thickening of the specular reflection line is visible, as well as a reflection on the far side of the impact site. Image (d) shows a false negative where only a small, slight deviation in the shape of the specular reflection is visible. Image (e) shows the same false negative made by YOLOv12, suggesting the information in this image is too subtle for this dent to be detected. Image (f) depicts the only false positive by RT-DETR. This image actually has two predictions of the same dent with almost identical coordinates, both above the confidence threshold of 0.25, so the duplicate detection is a false positive. Additional post-processing could be done to remove duplicate detections with high IoU like this. These images show that RT-DETR is prone to missing very small dents that cause only subtle deviations in specular reflections, such as the dents made by the impact gun. RT-DETR is also unlikely to produce false positives.

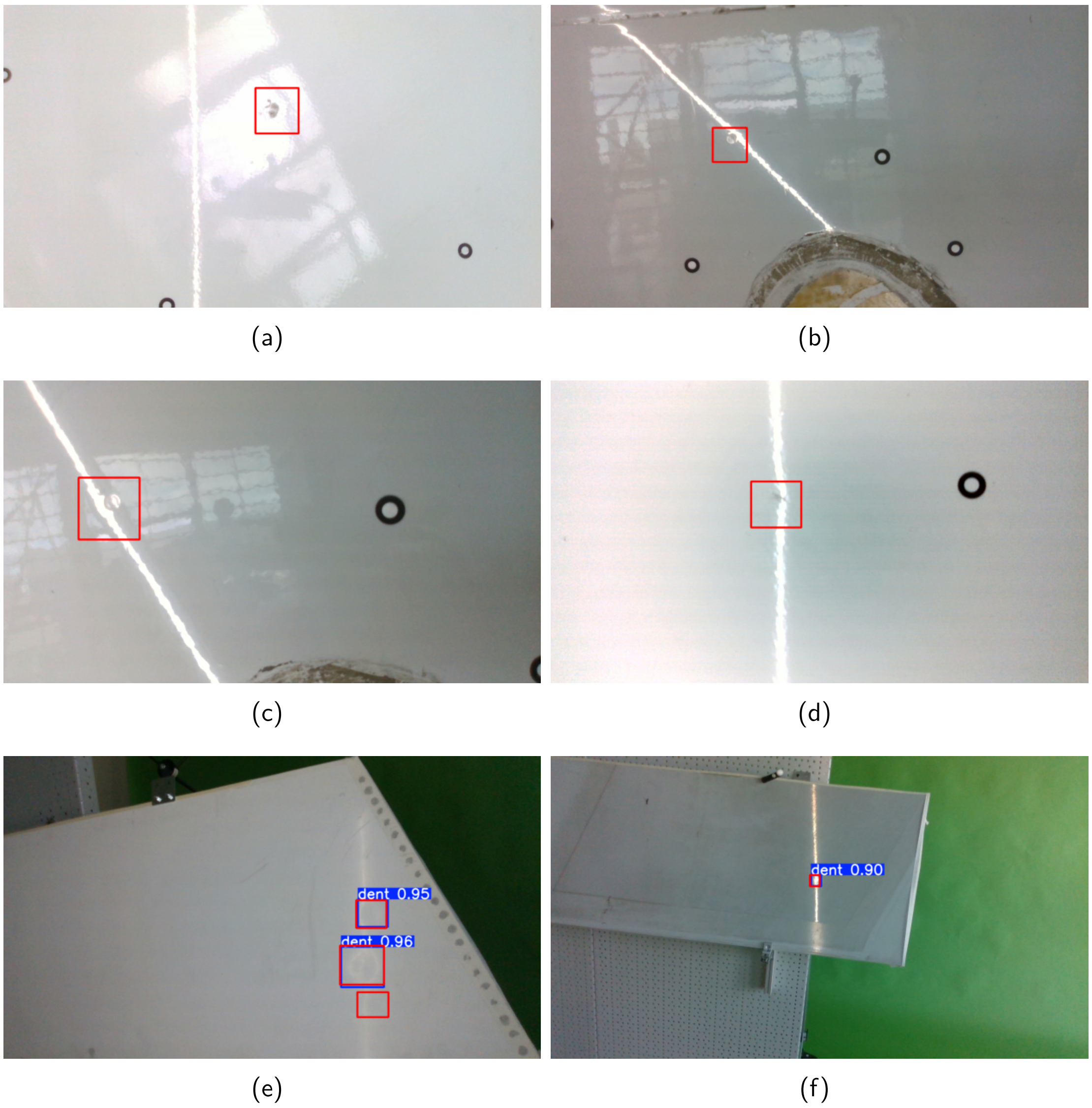


Figure 5.13: All incorrect detections by RT-DETR. Images (a) and (b) are from the test set, (c) through (f) are from the validation set. The predicted bounding boxes are shown in blue, the ground truth bounding boxes in red.

5.2.2 Speed

The processing speed of both models was tested on an 86 second long video captured during the robotic inspection testing. The frames of the video were sequentially processed, and the time needed for prediction on each frame saved. These times were later averaged to determine the average speed per frame, and the number of frames per second calculated. The test was carried out twice and the results averaged. RT-DETR achieved 47.232 fps and YOLOv12 47.097 fps. At a resolution of 640x640, there is no significant difference in the speed of these models when processing images sequentially. Both models achieve more than 30 fps at this image resolution, making them suitable for processing videos captured at 30 fps. Table 5.7 shows the full results.

Model	RT-DETR	YOLOv12
Measurement 1 (fps)	47.22935	47.09106
Measurement 2 (fps)	47.23517	47.10293
average	47.23226	47.09699

Table 5.7: Results of video speed testing.

5.2.3 Robotic Inspection

The lighting setup and trained models were tested on a robotic inspection setup. A UR10e was used to inspect plane part B for dents. The part was mounted vertically and positioned in front of the robot and the light strip and RealSense camera attached to the end of the robot, as shown in Figure 5.14.

The robot was set to follow a predetermined path: first it approaches the lower right corner, then it moves to the lower left corner. From here, it moves up until the camera's view overlaps slightly with the view from the previous height, and then moves to the right edge of the part. It continues to scan the part by alternating between moving left and right and shifting up in between, until the entire part has been viewed by the camera. Figure 5.15 shows the approximate path of the end effector. During these movements, the robot keeps the camera directed in the same orientation in Cartesian space, since the inspected part is mostly flat, with minimal curvature. The path was also designed to maintain a distance of approximately 30 cm between the camera and the inspection surface.

The video from the camera was streamed to a computer at a resolution of 1280x720 at 30 fps and passed to the detection model for inference at a resolution of 640x640; both the original video and and the video with annotations were saved. During the inspection, an additional camera was positioned behind the robot to capture the process from an external viewpoint. The inspection was carried out with the robot at two different movement speeds for both detection models.

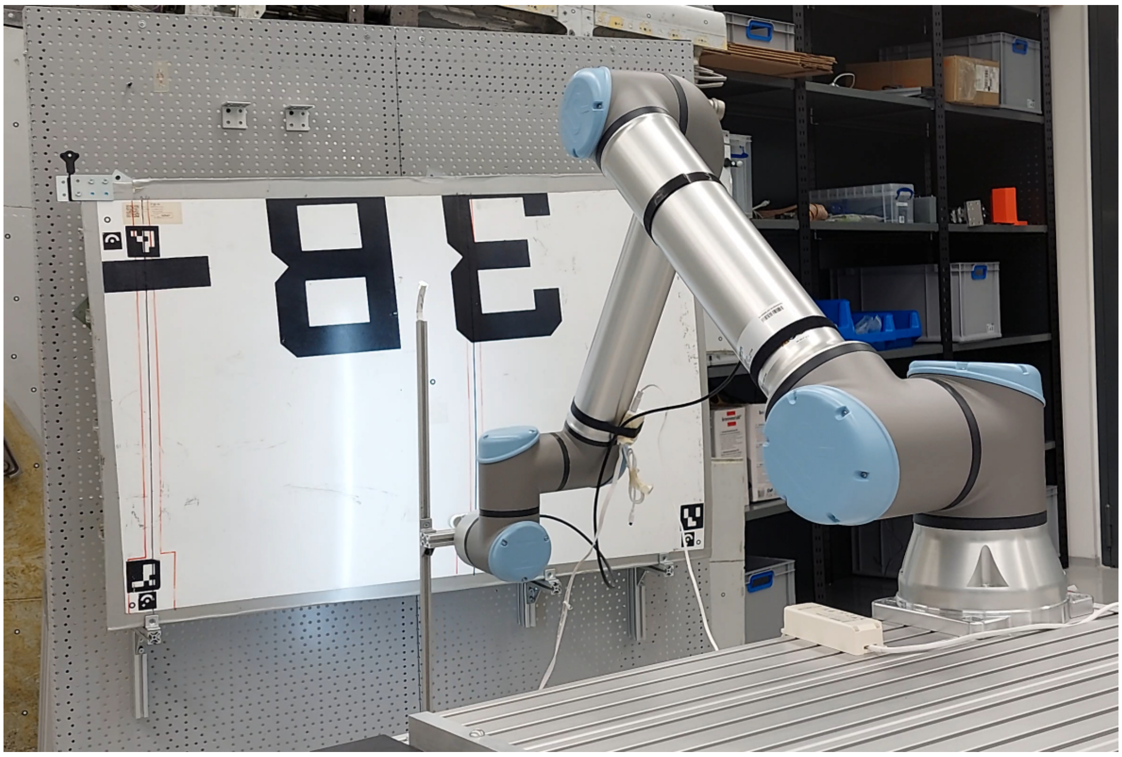


Figure 5.14: Robotic inspection setup.

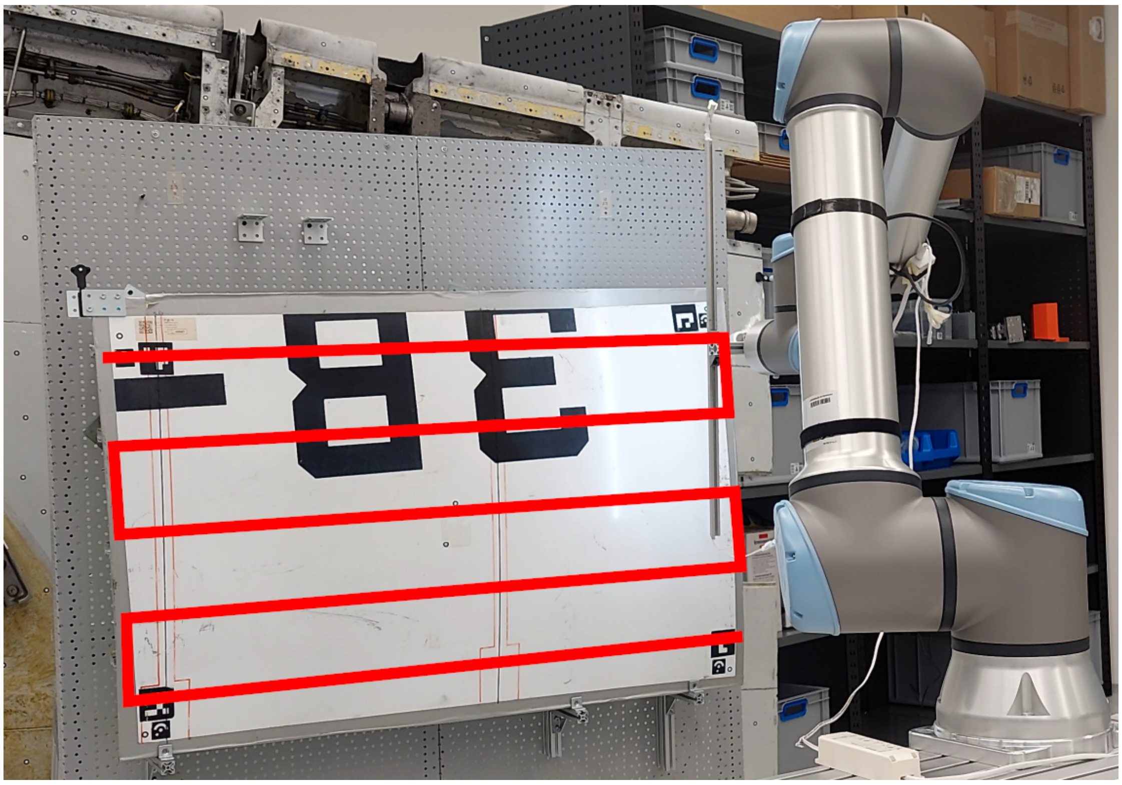


Figure 5.15: Robotic inspection path of plane part B.

Figure 5.16 shows frames from the video of RT-DETR detections with the robot set to the slower speed. The dent is not detected until it comes close to the specular reflection and causes a significant curvature of the line shape. The dent continues to be detected in every video frame until the dent has passed to the other side of the specular reflection, and the line shape no longer has a significant curve. In this example, the dent was visible for 11 consecutive video frames. The size of the bounding box varies slightly in each frame, but matches the position of the dent well.

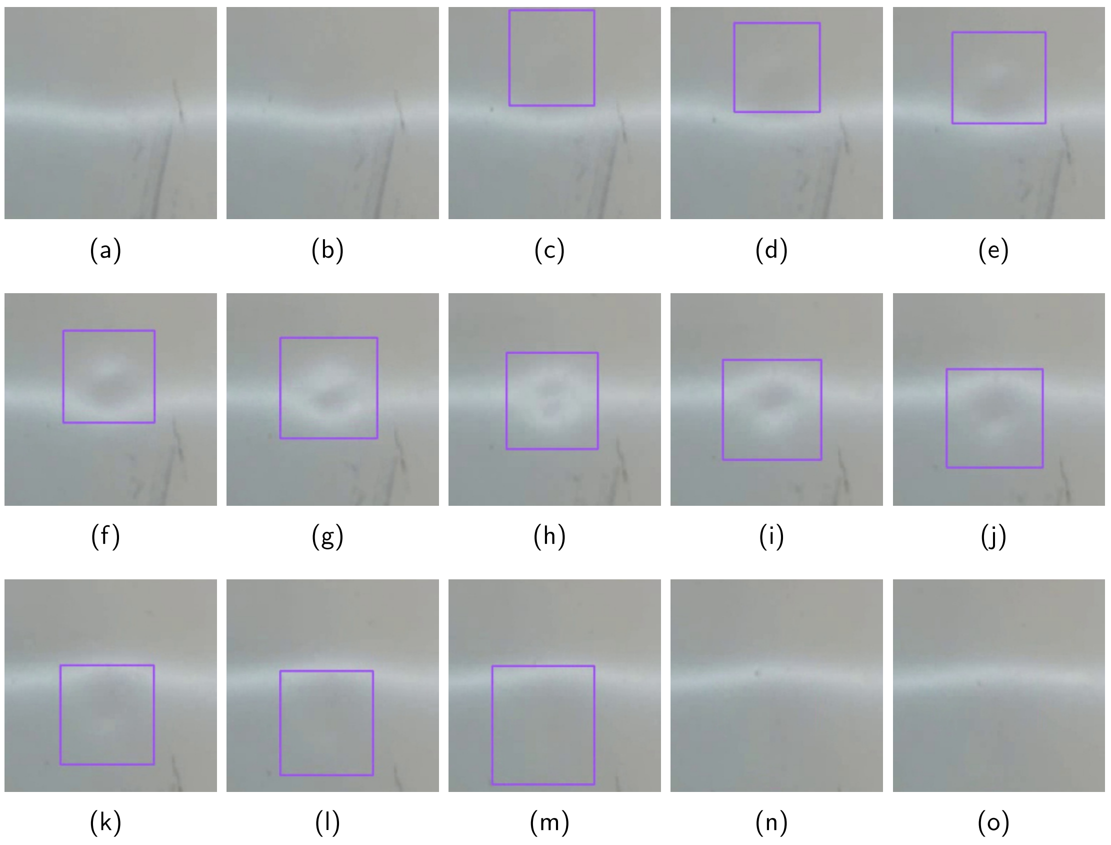


Figure 5.16: RT-DETR detection on video created during robotic inspection. The video frames are cropped and rotated for increased visibility.

Both models were able to detect all three dents in at least 4 consecutive video frames at both robot speeds.YOLOv12 produced a false positive in two video frames, shown in Figure 5.17. The area of the false positive appears slightly darker in the video. During later examination of this area of the aircraft part, no dirt or other responsible features could be found, only a single cat hair. The slow speed setting takes about 65 seconds to complete the path from the lower right corner to the upper left corner, the fast setting takes about 32 seconds. The entire surface of the part is inspected, which has an area of 0.773 m^2. Videos of the full inspection with RT-DETR can be viewed at https://youtu.be/JI4b641FUNQ (slow) and https://youtu.be/x6PyziefCkc (fast).

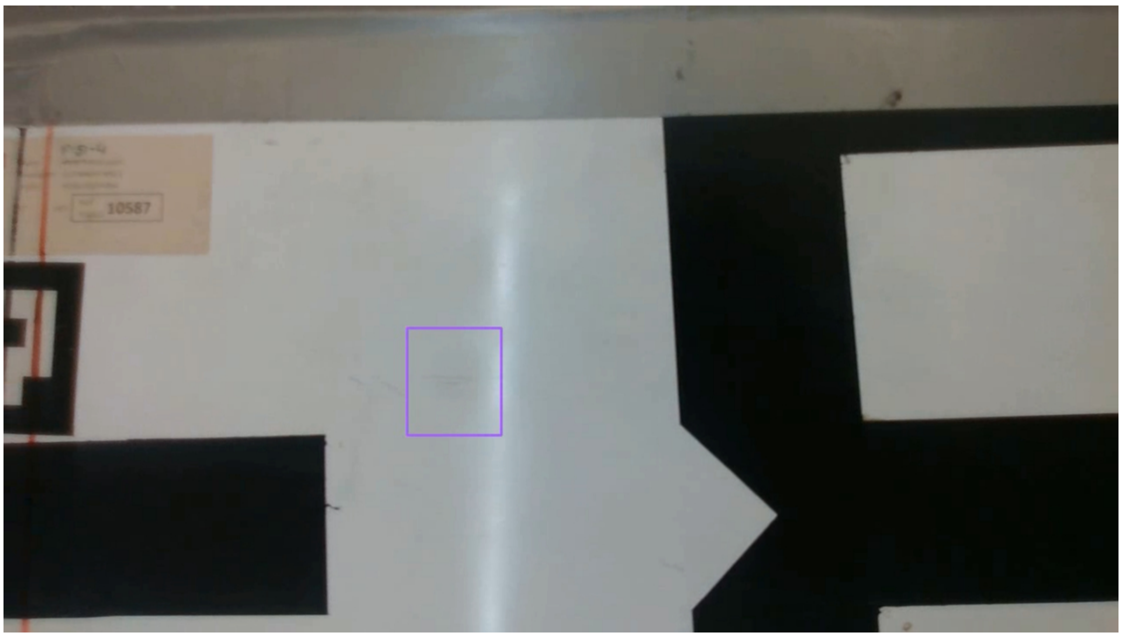


Figure 5.17: False positive produced by YOLOv12 on a video.

Chapter 6

Discussion

6.1 Dataset

6.1.1 Dataset Images

The creation of a new dataset of dent images was successful. Over 3000 images of aircraft parts were captured and annotated, as well as an additional set of over 3000 images of metal plates. The two datasets are sufficiently large for deep learning, and feature a decent amount of variety in dents, features, and surface specularities. The images contain line-shaped specular reflections from the light strip as well as some other reflections from windows. The dataset contains both bounding box annotations as well as segmentation masks. The dataset could still be improved by adding more images to increase diversity and be more representative of real inspection scenarios. A wider variety of aircraft parts, especially parts with higher convexity, is needed, as the available parts were mostly flat or only gently curved; the reflected line would appear very different on more curved surface geometry. A wider variety of surface colors and other defects is also desirable, since the dataset contains mostly dents and missing paint on white surfaces. Since most of the featured dents are roughly circular, more variety in dent shapes may be helpful. Also, more variety in backgrounds would make models more robust against false negatives on background scenery. Lastly, more variety of light sources and reflections would help to accurately represent real inspection conditions.

A dataset created by one person using only a few plane parts in one building has some biases. Some airplane features will be overrepresented, some will be missing completely. Backgrounds and lighting present in the images will have less variation. The annotation method and style of the individual annotator will introduce bias, especially concerning where to draw the boundary of a dent that does not have sharp boundaries; some annotators would favor smaller or larger boundaries. It is still important for researchers to publish small datasets because these could be combined into a larger dataset with more diversity.

6.1.2 Annotation Method

The big advantage of the tracking-based annotation method is that each dent only needs to be measured once for annotation, regardless of how many images it appears in. This method is especially helpful, and potentially time-saving, for use cases where many images of a small number of features need to be annotated. The method is also helpful for creating segmentation masks, since these take more time to manually annotate than bounding boxes but take the same amount of time with the new method.

The big downside of this annotation method is the large upfront time investment needed to set up and calibrate the tracking system components and determine the transformation between the tracking system's coordinate system and the camera's coordinate system. All of these steps are vulnerable to inaccuracies, and determining the causes of errors can take considerable time.

The localization accuracy of dents in some of the captured annotations is lower than desired, resulting in some captures being discarded. The decision was made to keep images with only slight location inaccuracies in order to maintain a large dataset. This vulnerability to inaccurate localization is another drawback of using this annotation method under time constraints, when not all causes of inaccuracies could be identified in time.

The decision to keep mid-tier annotations may have affected the ability to learn precise bounding box locations, but excluding them could have resulted in an overall performance drop due to the dataset being significantly smaller. The quality issues were more severe on plane parts, perhaps due to the larger size of the parts and inaccurate tracking of their orientations. The decision to retain these images allows for later manual reannotation if higher accuracy is desired.

For future work with this annotation method, frequent recalibration of the tracking system and targets is recommended, as the tracking cameras can slightly shift, and tracking targets can slightly change when touched, resulting in image annotations drifting farther away from the correct position over time. Another recommendation is using only individual markers to track large objects, as these can be positioned far away from each other, increasing the accuracy of the tracked orientation, thus improving annotation localization of large parts.

Due to less than ideal calibration of the measurement tool, the outlines of some segmentation masks in the dataset are somewhat irregular and could benefit from post-processing to smooth the borders.

6.2 Models and Inspection

Both RT-DETR and YOLOv12 achieved very high recall and precision on the test dataset. The performance difference was very small, making both models viable options for the task of detecting dents on aircraft surfaces. Analysis of the few incorrect detections made reveals that both models are prone to missed detections when only small deviations are visible in the specular reflections near dents. Specular reflections from other light sources like windows are also responsible for some false positives and negatives. The percentage of images with these characteristics is relatively low, which may account for these difficulties. Future work should strive to add more of these difficult images to the dataset.

The comparison of training on Dataset A and Dataset B showed that increasing the amount of training data with images of dents on other types of surfaces is a viable option for improving the performance of some models. RT-DETR had better or equal performance in all 4 metrics on the combined dataset. YOLOv12 saw a slight drop in some metrics on this dataset, but the most important metric, recall, improved. Combining datasets of aircraft defects with defects on other surfaces should be studied further as a method for increasing the amount of training data to improve the performance of deep learning models. Other approaches of increasing the amount of available training data in the field of aviation should also be studied, for example, by using generative models like GANs or creating synthetic data through 3D modeling.

The speed testing shows both models are fast enough for processing video at above 30 fps at a resolution of 640x640 pixels, making them suitable for detecting defects on a live video stream.

This inspection setup demonstrates how quickly the entire area of a plane part can be inspected with a robot while producing detection results in real time. Both models are capable of high performance even when the camera is moved relatively quickly. All dents on the part were successfully detected by both models, and the false positive of YOLOv12 during this test is consistent with its lower precision on the testing dataset. The performance of a real-world inspection setup would be measured on its ability to detect every dent on an aircraft, not its ability to detect the dent in every single image. This performance is expected to be slightly higher than the performance on the test dataset, since each dent is present in multiple frames of the video feed, giving the model several chances to detect the dent on slightly different images.

The full automation of the dent inspection process requires the deep learning model to be integrated into a larger software system. This system would need to save the results of positive detections and associate them with information about the global position of the detected area on the aircraft for confirmation and repair. An additional localization system or sensors would be needed to determine the position of the camera relative to the aircraft. The robot would also need path planning and a way to keep track of which surfaces have not yet been inspected to ensure all required surfaces are inspected. The detection system could also be extended to provide an estimate of the size of the detected dent.

Chapter 7

Conclusion

A large new dataset of dents on aircraft surfaces was created in this work. It contains annotated images of 6 different airplane parts with dents and other defects under varying lighting conditions, as well as additional images of dents on metal plates. This dataset will be publicly available, expanding the amount of images available to researchers. A new automated annotation method was developed and successfully used to create the dataset. The annotation method utilizes a tracking system to measure the locations of dents and to automatically annotate images as they are captured, completely eliminating the tedious work of manually annotating images. The new annotation method may be useful to others seeking to create large annotated datasets. YOLOv12 and RT-DETR were successfully trained to detect dents on this dataset, achieving at least 99% accuracy and precision. This proves the viability of using a single line-shaped specular reflection to detect small dents on aircraft surfaces. These models were also tested on a robotic inspection setup, showing the viability of an automated real-time dent detection system. The system is accurate even at a high speed, so there is real potential for reducing the time needed for inspections.

A lot more work can still be done on the subject of automating dent detection. Additional datasets of aircraft defects can be created to increase the diversity of data available. Particularly images of very slight, barely visible defects are most needed, since these are most difficult for deep learning models to learn to detect. The creation of artificial dent images should also be studied further, either by using GANs or 3D modeling techniques with accurate simulation of light and reflections. Further testing of the detection models on videos is also vital to more precisely determining the performance under these real-world conditions. The detection system could be extended to provide an estimate of the size of the detected dent using LIDAR. Adding a system that determines the position of dents relative to the aircraft is still needed so the detected areas can be examined and repaired by maintenance personnel.

Bibliography

[1] Nicolas P. Avdelidis, Antonios Tsourdos, Pasquale Lafiosca, Richard Plaster, Anna Plaster, and Mark Droznika. Defects recognition algorithm development from visual uav inspections. *Sensors*, 22(13), 2022.

[2] Soufiane Bouarfa, Anil Doğru, Ridwan Arizar, Reyhan Aydoğan, and Joselito Serafico. *Towards Automated Aircraft Maintenance Inspection. A use case of detecting aircraft dents using Mask R-CNN*. 2020.

[3] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html.

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *CoRR*, abs/2005.12872, 2020.

[5] David A. Forsyth and Jean Ponce. *Computer Vision - A Modern Approach*. Pearson, second edition, 2012.

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[7] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27, page 335. Curran Associates, Inc., 2014.

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask r-cnn. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[10] Bin Huang, Yan Ding, Guoliang Liu, Guohui Tian, and Shanmei Wang. Asd-yolo: An aircraft surface defects detection method using deformable convolution and attention mechanism. *Measurement*, 238:115300, 07 2024.

[11] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[12] Glenn Jocher. Ultralytics yolov5, 2020. https://github.com/ultralytics/yolov5.

[13] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics yolov8, 2023. https://github.com/ultralytics/ultralytics.

[14] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[15] Reinhard Klette. Springer, 2014.

[16] Ann-Kathrin Koschlik, Michael J. Scott, Thore Keser, Fiete Rauscher, Hendrik Meyer, Wim Verhagen, Pier Marzocca, Florian Raddatz, and Gerko Wende. Mixed reality for aircraft structural dent and buckle non-destructive evaluation. In *34th Congress of the International Council of the Aeronautical Sciences, ICAS 2024*, September 2024.

[17] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015.

[18] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. In *Proceedings of the Eighth International Conference on Learning Representations (ICLR 2020)*, April 2020.

[19] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2017.

[20] Dulani Meedeniya. *Deep Learning - A Beginners Guide*. Chapman and Hall/CRC, 2023.

[21] Julien Miranda, Jannic Veith, Stanislas Larnier, Ariane Herbulot, and Michel Devy. Machine learning approaches for defect classification on aircraft fuselage images aquired by an uav. page 10, 07 2019.

[22] Sung Hyun Park, Amir Tjolleng, Joonho Chang, Myeongsup Cha, Jongcheol Park, and Kihyo Jung. Detecting and localizing dents on vehicle bodies using region-based convolutional neural network. *Applied Sciences*, 10(4), 2020.

[23] Angelos Plastropoulos, Kostas Bardis, George Yazigi, Nicolas P. Avdelidis, and Mark Droznika. Aircraft skin machine learning-based defect detection and size estimation in visual inspections. *Technologies*, 12(9), 2024.

[24] Simon J.D. Prince. *Understanding Deep Learning*. The MIT Press, 2023. http://udlbook.com.

[25] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016.

[26] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2016.

[27] Pengfei Shi. Study on detection of dent defects of polariser based on deep convolutional generative adversarial network. *International Journal of Materials and Product Technology*, 68(1-2):18–28, 2024.

[28] Wang Sikun, Yang Ying, and Lu Cunwei. An image measurement system for detecting dents and scratches on the surface of used car body parts. In *2023 8th International Conference on Business and Industrial Research (ICBIR)*, pages 543–548, 2023.

[29] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, 2014.

[30] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer, second edition, 2022. https://szeliski.org/Book/.

[31] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer, second edition, 2022. p.70 Figure 2.17.

[32] Mingxing Tan, Ruoming Pang, and Quoc V. Le. Efficientdet: Scalable and efficient object detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10778–10787, 2020.

[33] Jiamin Tao, Yongjian Zhu, Frank Jiang, Hao Liu, and Hongzhan Liu. Rolling surface defect inspection for drum-shaped rollers based on deep learning. *IEEE Sensors Journal*, 22(9):8693–8700, 2022.

[34] Yunjie Tian, Qixiang Ye, and David Doermann. Yolov12: Attention-centric real-time object detectors. *arXiv preprint arXiv:2502.12524*, 2025.

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[36] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[37] Lei Wang, Lilan Luo, Peng Zheng, Tianyu Zheng, and Shan He. A fast dent detection method for curved glass using deep convolutional neural network. In *2019 IEEE 13th International Conference on Anti-counterfeiting, Security, and Identification (ASID)*, pages 117–121, 2019.

[38] Wen wei Lai, Xiao xing Zeng, Jian He, and Yuan long Deng. Aesthetic defect characterization of a polymeric polarizer via structured light illumination. *Polymer Testing*, 53:51–57, 2016.

[39] Yuri Yasuda, Fabio Cappabianco, Luiz Martins, and Jorge Gripp. Automated visual inspection of aircraft exterior using deep learning. 10 2021.

[40] Hsu-Nan Yen and Min-Jyun Syu. Inspection of polarizer tiny bump defects using computer vision. In *2015 IEEE International Conference on Consumer Electronics (ICCE)*, pages 525–527, 2015.

[41] Yu-Jin Zhang. *3D Computer Vision - Foundations and Advanced Methodologies*. Springer, 2024.

[42] Yian Zhao, Wenyu Lv, Shangliang Xu, Jinman Wei, Guanzhong Wang, Qingqing Dang, Yi Liu, and Jie Chen. Detrs beat yolos on real-time object detection. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16965–16974, 2024.