Computer Vision based Aircraft parts Inspection

Al Arafat

Université de Bourgogne, Bourgogne, France
Universitat de Girona, Catalunya, Spain
Heriot-Watt University, Scotland, United Kingdom

A Thesis Submitted for the Degree of
MSc Erasmus Mundus in Vision and Robotics (VIBOT)

· 2016 ·

Abstract

Visual preflight walk-around inspection becomes an exhausting job for maintenance operators and co-pilots as the number of daily flight increases. This thesis focuses on developing autonomous exterior visual inspection of few aircraft parts to assist the grounds-men, by reducing their work-load. Priority is given to automatic inspection of aircraft landing tire, engine turbine blades, and detection of presence of wheel chock and pitot cover, using computer vision and image processing methods. Challenges are addressed to make the system robust to scale, illumination, surrounding environment, and inspection time constraints. Suitable cues are wisely used to define regions of interest, and further to detect anomalies. The utilized features are color, shape, ridges, geometrical primitives (lines, circles), and periodicity of intensity signals. Projection of simplified 3D model onto the image plane is used to approximate shapes to simplify the implementation which eventually reduces detection as well as inspection-time complexity. The developed system is integrated with a ground mobile robot which autonomously does walk-around and stops at predefined checkpoints. The robot uses a pan-tilt-zoom camera to capture images of check-items which the system later process for automatic inspection.

The implementation results give promising accuracy and thus, demonstrate the usability of our system for automated airplane exterior inspection.

Contents

List of Figures

Acknowledgments

First, I would like to thank my supervisor, Professor Jean-José ORTEU for allowing me to work in this interesting project. I would also like to thank my co-supervisor, Igor Jovančević, PhD student. They were supporting me by giving directives, providing feedback on my work, and moreover, by encouraging me.

This work was part of the AIR-COBOT project approved by the Aerospace Valley world competitiveness cluster. The partners of the AIR-COBOT project (AKKA TECHNOLOGIES, Airbus Group, ARMINES, 2MoRO Solutions, M3 SYSTEMS and STERELA) are also gratefully acknowledged for their support.

Thank you to my friends who were giving me ideas about solving problems and at most, to my family for their continuous support.

Chapter 1

Introduction

1.1 Airplane Inspection

Airplanes are periodically inspected, either on the airport tarmac between flights or in a hangar as for maintenance requirement. These routine inspections can be classified into five different types, A-check, B-check, C-check, D-check and Daily check. All these inspections are done by human operators. They do 70% of the total inspections, visually. Occasionally, they use inspection tools, depending on varying required tasks. Relatively lighter, Daily check, can have different names; pre-flight, post-flight, and overnight are few of them. Daily check requires visual inspection of a set of external aircraft items from specific positions, before, after, or between flight(s).

Walk-around inspection is a part of the primary step in order to complete a daily check. Maintenance operator and co-pilot work on checking specific items such as: Fuselage, Nose Section, Tire, Engine blades, Pitot, Oxygen bay, Static Port, Air Inlet Valve, Altimeter, Transponders, Emergency Locater Transmitter, and few other sensors, their conditions and whether they are covered or free to operate. They do this checking by following a conventional walk-around path and stopping at each fixed checkpoint. Figure 1.1 shows the conventional walk-around trajectory for Airbus A320.

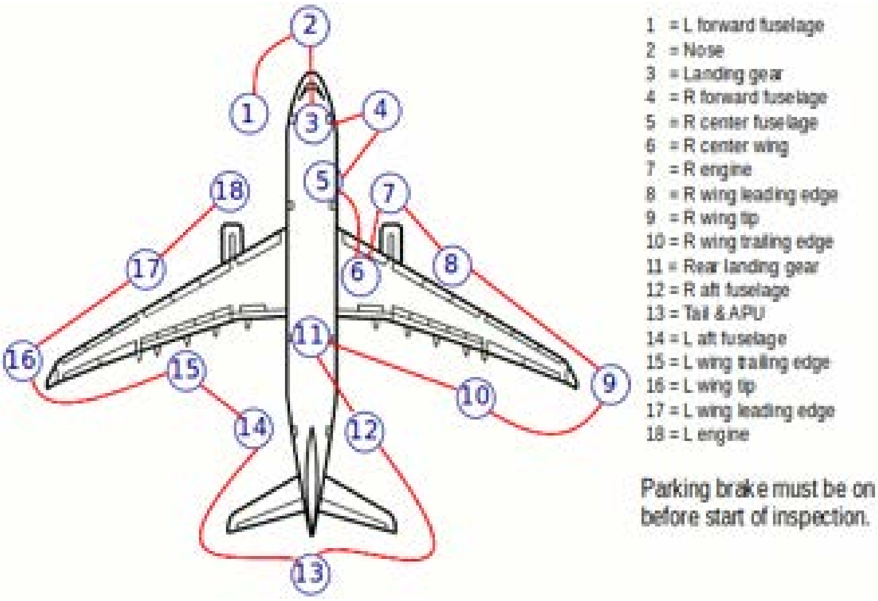


Figure 1.1: Airbus A-320 walk-around [1]

Due to the increasing number of airplanes and flights, airplane ground period is becoming shorter. This shorter inspection phase makes daily check, particularly pre-flight, more challenging. This could let the visual checks being monotonous and even stressful for co-pilots and maintenance operators. Moreover, since the number of flight increases around the world and needs immediate inspection, daily check also requires more human resources which eventually increases maintenance costs.

In order to make the inspection quicker, more exhaustive, more accurate, and more traceable, a multi-partners research project, “Air-COBOT”, is initiated. This project works on developing a mobile robot equipped with a Pan-Tilt-Zoom (PTZ) camera and a 3D scanner to perform automatic inspection. The principle aim of this multi-partner project is to reduce the man-hour needed for ground inspection and help maintenance operator and co-pilot to lessen their work-load.

1.2 Objectives

Responsibilities of this thesis work fall under the scope of the “Air-COBOT” project. The “Air-COBOT” project, as a whole, engaged in autonomous navigation of the robot and inspecting required major aircraft items for daily check. For this purpose, it investigates 2D images and 3D point clouds from about 20 checkpoints. The entire system works automatically, although limited to one specific aircraft model, Airbus A-320. Thus, all the required tasks are restrained toward solving problems considering Airbus A-320 model.

This thesis focuses on detecting and inspecting few of the required items using 2D images. The objective of this thesis can be categorized into 4 subsections.

- **Inspecting aircraft tires**: Images from the robot include landing tires surrounded by other objects. First, the system needs to detect the tires in the image and then, decide whether the tires are usable or not by examining the tire threads' condition.

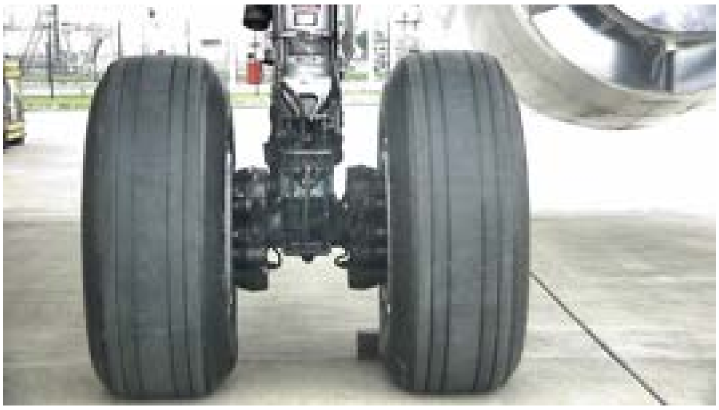


Figure 1.2: Airbus A-320 landing wheels

- **Detecting the presence of chocks in front of tires**: The aim of this requirement is to check whether the grounds-men removed the wheel chocks or left them mistakenly, before the flight.

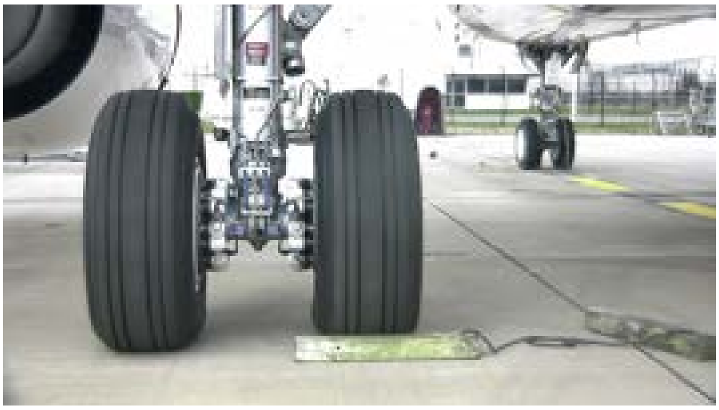


Figure 1.3: Airbus A-320 wheel chocks, one wheel has got a chock

- **Inspecting reactor or engine blades**: This task also requires the engine to be isolated from the background, at first. Then, the system looks for any foreign object stuck between the engine blades.

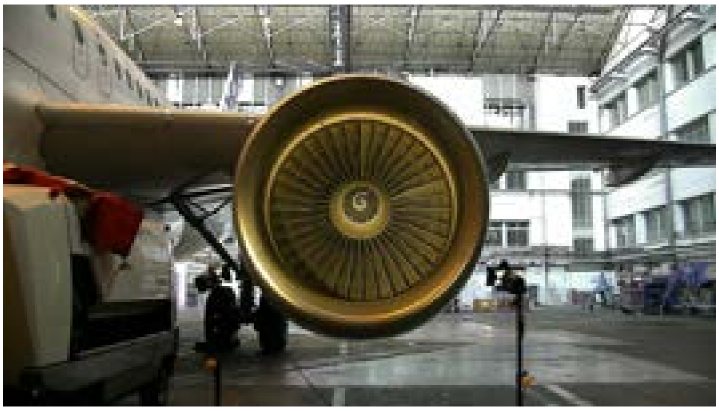


Figure 1.4: Airbus A-320 engine

- **Checking whether any pitot tube is covered by a pitot-cover**: Pitot tubes are used to measure the airspeed of an aircraft. This subsection aims to find out whether any of the Pitot tubes is left covered or all of them are uncovered to work accordingly.

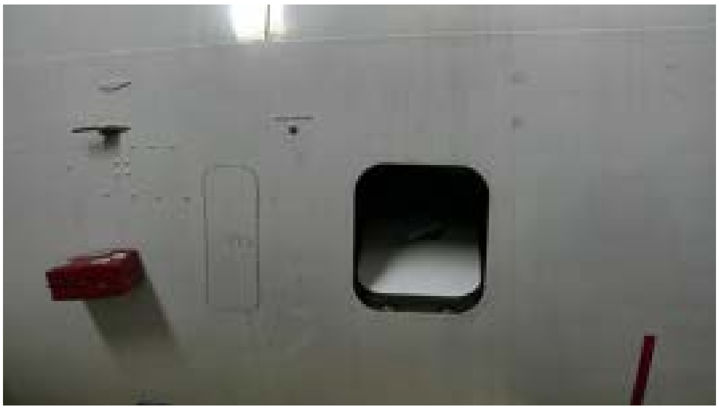


Figure 1.5: Airbus A-320 pitot tubes, one of them is covered and another one is open

1.3 Context and Dataset

"Air-COBOT" is an autonomous mobile robot on the ground, capable of communicating with human operators and sensing infrastructures around the aircraft. In a typical scenario, navigation algorithms lead the robot to a checking position which is localized with respect to the aircraft model. Further, automated image acquisition is provided by controlling PTZ camera and pointing it towards the item to be inspected. Inspection is performed by processing the images taken by the camera [3].

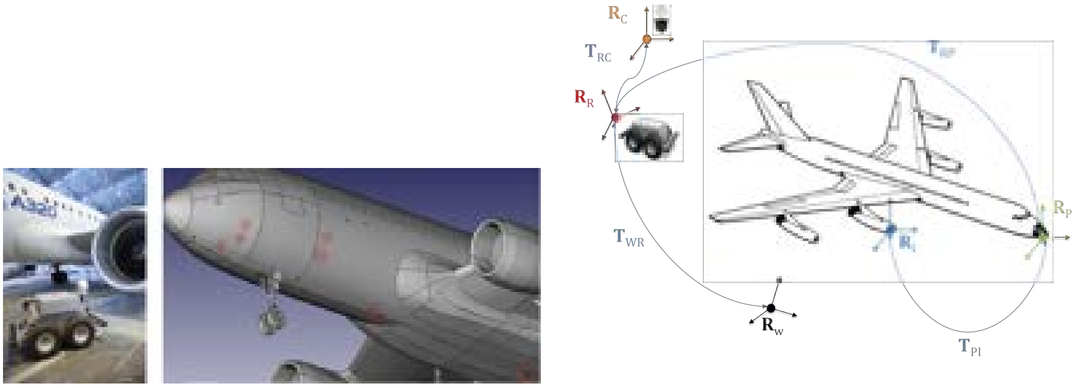
1.3.1 Projection

We have a simplified 3D model of Airbus A-320 airplane (Figure 1.6a). To assist detection and inspection of aircraft items, information from the simplified 3D model is used. Basically, this information is used to extract the model of an item during it's detection or inspection. CATIA software is used fo this purpose *i.e.* to derive the model from the global simplified numerical airplane model. Further, database of these items is created.

Later, projection is done utilizing the following information:

- Approximate pose of the robot provided by localization module,
- Relative position of the camera on the robot,
- Current pan, tilt, zoom values,
- Position of the element in the 3D model of the plane,
- Calibration parameters of the camera for certain zoom.

Figure 1.6b shows all the coordinate frames involved in projection.



(a) 3D model of Airbus A320 airplane

(b) Coordinate frames involved in projection

Figure 1.6: Sample tire images

All the transformations (Fig 1.6b) are applied to point PTZ camera toward a particular item and project the item onto the image plane. This step is also essential in robot pose estimation with respect to the airplane.

1.3.2 Dataset

The entire thesis work requires three different image datasets. They are,

1. Dataset for landing wheel, which contains 118 images. The same dataset is also used for wheel chocks detection.

2. Engine inspection dataset contains 104 images.

3. Dataset for pitot covers detection contains 42 images.

The datasets are created on real Airbus A-320 airplane. To check robustness of the implementation, the real datasets have been extended by varying brightness and/or contrast of the images up to 10%. All the dataset images are in RGB-color space and have a resolution of 1920×1080.

1.4 Related Works

1.4.1 Robotic Aircraft Inspection

In a whole, robotic airplane inspection from ground, based on computer vision, is a specific application not studied enough before. Previous effort mainly focuses on detailed airplane skin inspection with robots crawling on the airplane surface.

Mel Siegel et al. [4] designed and constructed Automated NonDestructive Inspector (ANDI) at Carnegie Mellon University to inspect aircraft. ANDI used to crawl to move and checked rivets on the fuselage. For inspection, ANDI utilized Eddy-Current sensors and 3D stereoscopic cameras to search for cracks and surface corrosion. In contrast, we are proposing visual inspection from exterior of an airplane using a ground robot. Our robot with PTZ camera and 3D scanner gives us space to efficiently inspect more airplane items within a short period of time and without having direct contact with the airplane.

In 2015, Easyjet [5] initiated a work with Bristol University to develop a flying robot to inspect aircraft. The project is planned to use high definition cameras and lasers to scan the outer surface of an aircraft. Alongside the drone, their crew would use virtual reality glasses to do technical inspection when aircrafts are far from home base. Our project initiated long before them and has already been implemented. Moreover, using a ground robot for inspection is cost-effective and more easy to maintain. Our obvious constraint is that we can not verify the state of the upper part of the airplane.

1.4.2 Visual Inspection

Visual Inspection of aircraft, as an entire project and towards solving related specific problems, has not seen in the literature except the one from Mel Siegel et al [4]. In spite of this, there are works that used visual inspection for industrial purposes to ensure quality production which are related to the thesis tasks. Visual inspection, often, also requires detection of the object beforehand of inspection. Domingo Mery et al. [6] used Automated Multiple View Inspection of the test object using X-ray imaging and calibration, as *a-priori*. José A. Calderón-Martinez et al. [7] proposed solution to defect detection and classification using Convolutional Neural Network where they used TDI camera as imaging module.

1.5 Organization of the rest of the thesis

This paper describes an insight to construct systems for four mandatory requirements of the thesis work, to inspect airplane tires and engine blades, and to detect the wheel-chock and pitot tube cover. To accomplish each of the tasks, individual pipeline of image processing, and computer vision methods are followed. In Chapter 2, we elaborate tire inspection. Engine inspection is illustrated in Chapter 4. Detection of the presence of wheel-chock and pitot cover are discussed in Chapter 3 and Chapter 5, respectively. Finally, we conclude our thesis in Chapter 6.

Chapter 2

Tire Inspection

2.1 Introduction

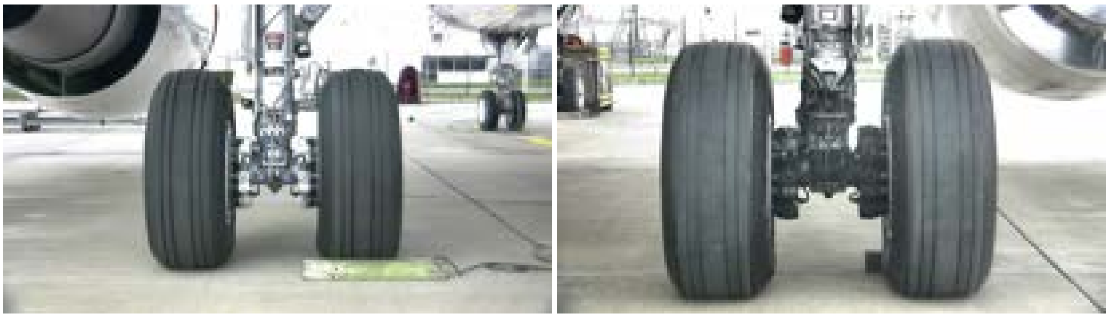
Inspection of tire is an important security check task. Before a flight or after a flight, the tire condition is checked for further use. The main aim of this check is to check the tire treads, whether they are clearly visible or faded out. Clear tire treads are essential to have, for an airplane, to avoid sliding on airport apron.

Tire tread pattern differs with airplane models. As the project is intended to inspect Airbus A-320, the implementation works particularly for A-320's tire tread pattern. Figure 2.1a shows the tire tread pattern of Airbus A-320. Each tire contains 4 parallel treads.

To capture images for tire inspection, our robot tends to get positioned approximately at the predefined walk-around point. The captured images are expected to have both the tires being well focused. However, due to noisy localization of the robot with respect to airplane, the captured images might miss one tire. Additionally, this noisy localization is the reason for scaling problem of image objects, for our case, tires. If the treads are sharply visible, then the tire is classified as usable and if a tire has faded treads, then it is classified as non-usable. Example images in Figure 2.1 show usable and non-usable tires.

Inspection of tires includes three different tasks, **(1)** detecting the tires, **(2)** segmenting them from the background, and **(3)** examining the tread condition of the segmented tires. To detect the tire from the image, template matching technique is used. Later, we introduce image processing pipeline to isolate the tires and check the treads.

This work has two main contributions. **(1)** The first contribution is to introduce ridge based intensity profiling to inspect the tire treads' condition. **(2)** The second contribution is in developing the entire methodology to inspect tires robustly. Our system shows robustness



(a) Usable good conditioned tire

(b) No-usable bad conditioned tire

Figure 2.1: Sample tire images

to noises accumulated from relative pose changes of the robot with respect to the airplane, to its surrounding environment, and to illumination changes.

We project tire onto 2D image plane from 3D simplified Airbus A-320 model, illustrated in Section 1.3.1, to approximate the size of the tires. Tire projection result is shown in Figure 2.2.

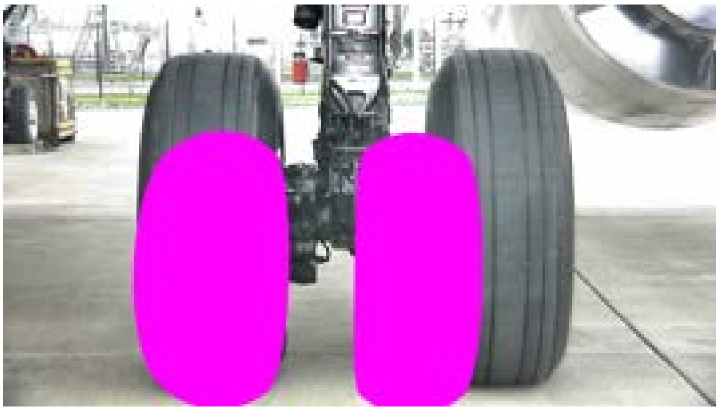


Figure 2.2: Tire projection onto image plane

We use Normalized Correlation based Template Matching (NCORR) to detect the tires. A small tire patch is used as the template for matching. First, a Bilateral filter [8] smooths both the original captured image and the template patch. To isolate tire from the background, GrabCut segmentation [9] method is used. Markers for the GrabCut are automatically generated using two different information.

1. First information is the output matching matrix. This output matrix shows matching values for the patch at each pixel's neighborhood of the image. We use it to localize our tires in the image.

2. Second resource is the approximate size of the tire in the image (Figure 2.2). We use this approximation to estimate bounding boxes around the tires. We set these bounding boxes as foreground markers.

After segmenting the tires using GrabCut segmentation, we do filtering to neat the segmentation result. To examine the treads, we detect ridges from the segmented tire regions and later, do intensity profiling of the ridge image. Our system identifies the location of the worn-out treads (or defects) on the tire surface and subsequently, measures their lengths. It comments on whether a tire is reusable or needs a change by analyzing the defect length.

We discuss about related works in Section 2.2. Tire inspection approach is shown in section 2.3 and are discussed in section 2.5 and section 2.7. Section 2.8 shows our experimental results.

2.2 Related Works

Tire inspection. Automated tire inspection is an interesting problem to solve that attracts several researchers and scientists, from a long before. Some of them were using different image modalities and some of them were using additional instruments.

Back in 1939, W. H. Capen [10] had patented his work on tire inspection where he used X-rays and special apparatuses which is the earliest known attempt in literature for inspecting tire. Use of optical devices and apparatus is yet now an attractive field of interest for researchers. In recent patented works, [11–14], researchers inspected tire shape using different imaging methods and auxiliary gears. Among such latest works, Mizukusa et al. [11] used light source and a curved mirror to detect the deformed tires. In contrast, the aim of our task is different from their's as we work on 2D images and examine the tire treads from images without using any additional hardwares.

Use of image processing techniques, without additional equipments to inspect tires offers strategical improvements. They can significantly reduce maintenance and operational cost for additional equipments. Takahashi and Sakoda [12] examined tire shape by taking 2D images and using a-priori offset image. Simialrly, patent work from Teti et al. [15] proposed in-motion vehicle tire inspection. Multiple cameras were used and when at least one tire of any car came into the field of interest of any of the cameras, they captured images. Later, they examined tread depth using image processing methods. For our task, we use one PTZ camera to capture aircraft tire image and from there, detect and inspect the treads. Thus, our method would be computationally less expensive from using multiple cameras and processing several images from several cameras.

Yair Wiseman et al. [16] detected tire surface defects by directly working on JPEG compression data. While their method did not give good results for motion blur, white defect, or smaller defects, our method finds small tread fading, white defect, and also works with varying illuminations. Moreover, due to using intensity profiling to find defects, our framework is fast enough to find faded tire treads on the tire surface.

André P. Dias et al. [17] identified tire and red dots on tire, separately, using Adaptive mean based Background Subtraction method. The tire was presumed to be on conveyor belt which results in invariant background for images. Contrarily, our method detects and inspects tires robustly whether the background is from hanger or from tarmac.

Another approach toward tire inspection is to use Machine learning with image processing techniques to classify defects from no-defects. Yuanyuan Xiang et al. [18] used dictionary representation for defect detection. They learned the dictionary from normal images by using K-SVD algorithm and classified 'defect' and 'no-defects' by using representation coefficients of the image patches. Their experiment had 93.4% detection accuracy with 1.9% error rate.

Similarly, Qiang Guo et al. [19] detected tire defect by first, using Local Total variation filter to separate out texture from background and defect. Later, vertical mean filter was used to extract defect from background. However, the implementation was limited to detecting defects on tire surface during production stage. Our problem statement requires inspection of tires that are already in use.

Tasneem Wahdan et al. [20] detected tire type by extracting thirty six near-centre coefficients of discrete cosine transform and gradient difference at 0°, 45°, 90°, and 135°. Template matching was also used to detect tire DOT code in order to classify the tire type. Their method gave 100% accurate classification of tread pattern but failed in classifying DOT codes. Inspecting tire tread using gradient difference can conclude in numerous false alarms for altering illuminations and for unusual spots on treads.

Tire inspection can be carried out by examining the tire treads [21]. Penny Chen et al. [21] detected groove defects by using automatic image thresholding, and surface defect by examining tread patterns using template matching.

Tire Detection. Previous works on tire inspection assumed that the tire has already been detected and thus, skipped tire detection. The works were involved in detecting defects on the tire surface, with calibrated visual system. In contrary, our work requires us to detect the tire before doing inspection. Although no previous work on aircraft tire detection has been seen in literature, researchers worked on similar detection for different purposes.

One of the ways of detecting tire is to use filter convolution. Ofer Achler et al. [22] detected car wheels by computing Gaussian filter-banks with principal component analysis and later, comparing the Gaussian mixture model of wheel and roadbed. The method did 78% accurate detection while our implementation gives us accuracy rate of 98.3%, although the two methods are not tested on the same dataset.

Many algorithms exist to perform object detection using machine learning [23, 24], template matching [25, 26], and saliency detection [27–30], although not particularly for tire detection. Roger M. Dufour et al. [25] identified target in a noisy image when the geometric parameters such as rotation and size are unknown by generating library of templates and later, fitting Newton's algorithm towards global minimum. However, their method was computationally expensive. Similarly, Luke Cole et al. [26] did object recognition using template matching. They proposed a two-tier hierarchical representation of images for each object to reduce the number of comparisons required for matching.

Segmentation. There are many segmentation methods based on different cues. Researchers use color [31–33], texture [34–36], shape [37, 38], and other cues as features to segment objects.

Another approach to do better segmentation is to use Interactive image segmentation [39–44]. Interactive image segmentation requires the user to specify seed points or draw strokes for foreground and background, separately. Seeded Region growing algorithm [44] is one of the basic powerful segmentation techniques which uses pixel values around the seed point to segment the image into regions. It suffers from few basic problems which led to work on further improvement.

An improvement, Watershed segmentation [39, 45, 46], does interactive segmentation by working on gradient image and using markers to restrict flooding. The main problem of watershed segmentation is that it suffers from finding a good level of hierarchy [39]. Interactive segmentation proposed by P. Miranda et al. [47] asked for user marking on the contour of the foreground object. They used arc weights to compute the path between neighboring points. The method can not be optimal and may require a large number of user interaction for highly textured image.

Similarly, Yuri Y. Boykov et al. [40] used graph cut algorithm to segment objects by modeling foreground and background pixel values according to histograms. They used max-flow min-cut algorithm to classify two virtual nodes, background and foreground. It suffers from over-segmentation problem.

C. Rother et al. [9] proposed GrabCut method that is an extension to the graph cut framework to segment color images. They modeled the foreground and background colors by Gaussian mixture models. GrabCut works nicely for separable background-foreground and can underperform when the foreground and background share similar color distribution [42]. For our work, we use GrabCut method to extract tires from the image because of it's simplicity in user interaction.

There are many improvements over GrabCut that have been offered. Shoudong Han et al. [41] proposed to integrate multi-scale nonlinear structure tensor texture (MSNST) feature within GrabCut framework to get improved segmentation performance. They extracted MSNST features in RGB color space and extracted color features of the image in $L*a*b*$ color space with conjugate norm. The segmentation results improved from Grabcut but it was computationally expensive.

In the same way, Wenxian Yang et al. [42] suggested improvement over GrabCut by introducing multiple marking over the random walks algorithm [43]. The random walk algorithm approaches toward minimizing a Dirichlet energy with boundary conditions, where different boundary conditions (different input seeds) always result in different harmonic functions [42]. L. Grady et al. [43] introduced the random walk algorithms for image segmentation where they produced better segmentation result comparing to GrabCut.

2.3 Developed Framework

Our method consists of three main steps: *detection, segmentation,* and *inspection.* The flowchart in Figure 2.3 shows the steps.

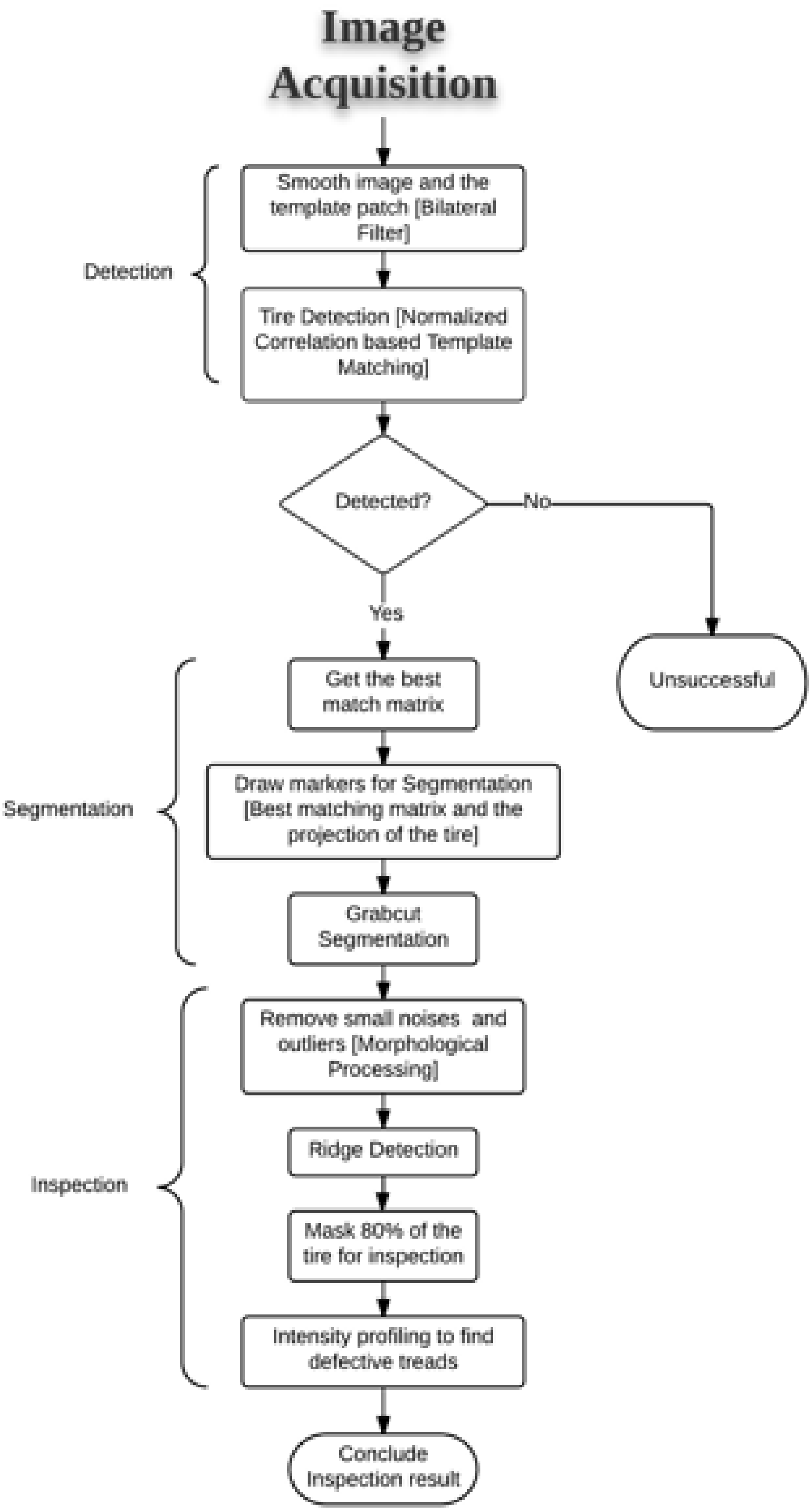


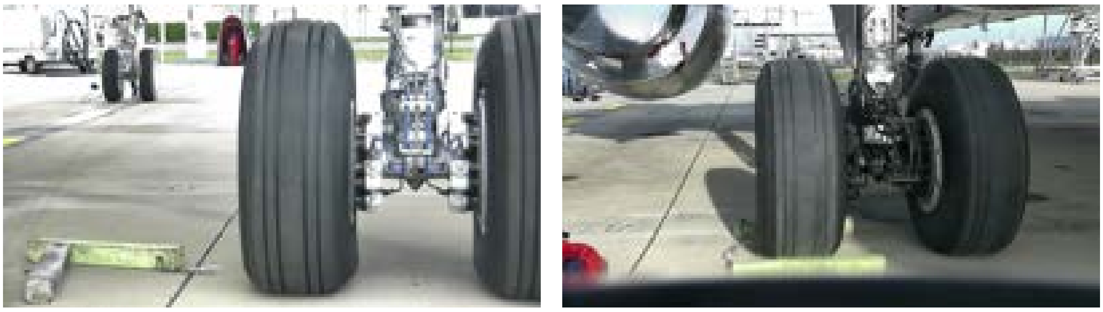
Image Acquisition
Smooth image and the template patch [Bilateral Filter]
Detection
Tire Detection [Normalized Correlation based Template Matching]
Detected?
No
Yes
Unsuccessful
Get the best match matrix
Segmentation
Draw markers for Segmentation [Best matching matrix and the projection of the tire]
Grabcut Segmentation
Remove small noises and outliers [Morphological Processing]
Inspection
Ridge Detection
Mask 80% of the tire for inspection
Intensity profiling to find defective treads
Conclude Inspection result

Figure 2.3: Work flow pipeline for tire inspection

2.4 Dataset

We generate our sample dataset based on three Airbus A-320 airplanes, *i.e.* 12 tires. The images are taken in hangar as well as outside on the tarmac with different weather conditions. We use 118 images for tire inspection, captured from the approximated walk-around position.

The robot is randomly misplaced within a circle with radius of r=50cm, and orientation with an error in range $[-15\,\text{deg}, 15\,\text{deg}]$. This can produces preferable images with both tires (Figure 1.2), but also ill-posed images with one tire or partially visible second tire (Figure 2.4a).



(a) Image missing second tire

(b) The second tire is ill-posed

Figure 2.4: Ill-posed tire images due to the noisy localization of the robot

2.5 Detection

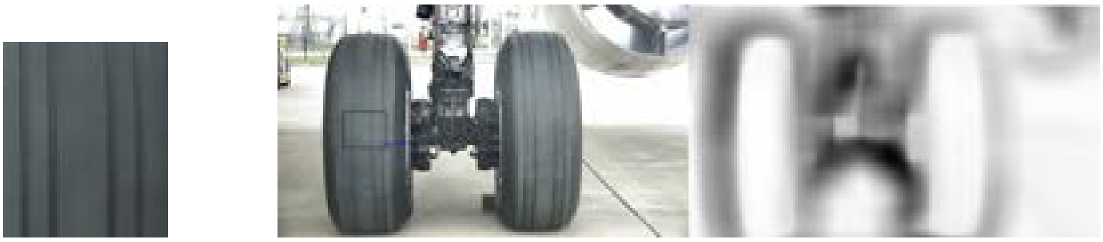
In the captured images (Figure 1.2), although the tires seem to be two separate dark regions, the detection becomes challenging due to the non-homogeneous intensities. Tire color varies significantly depending on the tire age as well as on the amount of dust present on the tire. Moreover, shadows, varying illumination caused by weather conditions, and altering surrounding environment (tarmac or hangar) add more complications. In detection and segmentation phases, high precision is required because the inspection requires exact segmented tire regions.

We use normalized cross correlation (2.1) based template matching to detect tire.

$$R(x,y) = \frac{\sum_{x',y'}(T(x',y')\cdot I(x+x',y+y'))}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x',y+y')^2}} \tag{2.1}$$

Luke Cole et al. [26] proposed a two-tier hierarchical representation of images for each object to reduce the number of comparisons required for template matching. They used a lego dataset of 100,000 images, 1000 per class and got best recognition rate of 86%. In comparison, our method has tire detection rate of 98.3%.

We use Figure 2.5a as our template patch for detecting tires. Instead of applying direct template matching using the patch, we apply Bilateral filter [8] on the original and template image. As Bilateral filter [8] smooths an image while preserving edges, we use it to remove the non-uniformity and noises of the images. It can be seen in Figure 2.5b that the higher matches are found around the tires region.



(a) Template patch used for matching

(b) Results from template matching: left image shows the best match found for the patch in the image and the right one shows the matching value matrix for the patch (white pixels mean matches)

Figure 2.5: Template matching to detect tire

The pattern of tire treads does not vary for a specific airplane model. It also does not have sharp details on the tire surface. Moreover, the probability of capturing rotated tire images is close to zero. Thus, we use Template matching based exhaustive search to detect the tire patch. We use normalized cross-correlation similarity metric because it is invariant to global brightness changes. Normalized cross-correlation based template matching detects tires with 98.3% accuracy, finds tires in 116 images out of 118 images.

We also evaluated alternative approach relying on Saliency detection. For our images, the middle part of the landing gear (between the two tires) is the most salient region. We detected the most salient point of the tire image using Covariance based Saliency detection [30]. Region Covariance based saliency detection gave us 82.20% accuracy in detecting the most salient region. Figure 2.6 shows an example of Covariance based saliency detection.

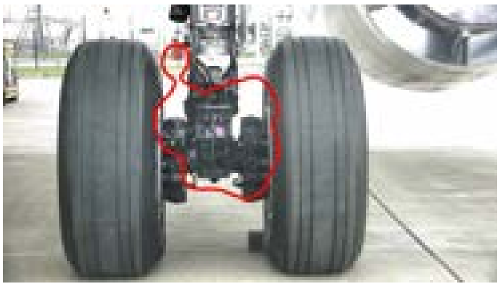


Figure 2.6: Most salient region detection using region covariance based saliency detection

Normalized cross-correlation based template matching has over-performed saliency based approach in detecting respectively, tires and the landing gear.

2.6 Segmentation

This section describes the segmentation of the two tire regions from the background. We continue from tire detection toward segmentation. Inspection of the tires requires accurately segmented tires regions. The tire-alike shadow regions and the presence of other dark objects around the tires, complicate the task of precisely segmenting the tires.

Color based Region growing segmentation method [44] suffers from over-segmentation since it is limited to local homogeneity criteria without having the global perspective (Figure 2.7).

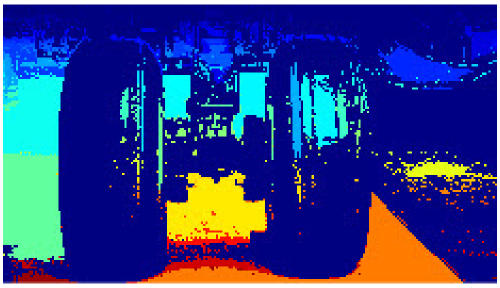


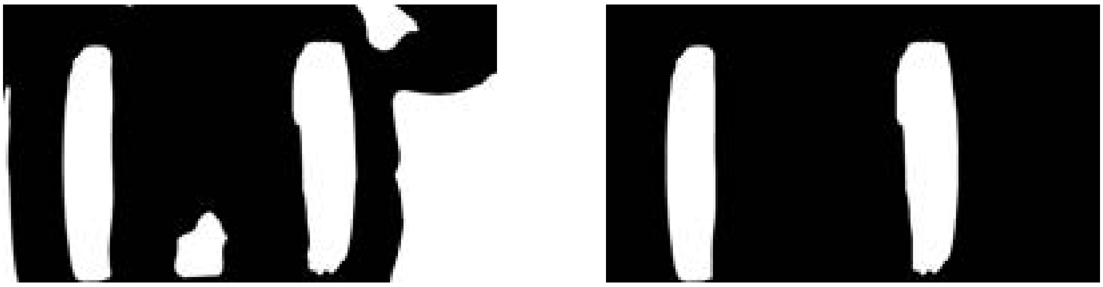
Figure 2.7: Region growing based tire image segmentation

To obtain better segmentation, we consider using our detection result. Assuming that, tires-alike regions have been successfully identified (Figure 2.5b), we use user interaction based image segmentation. Interaction based segmentation requires user input such as markers, seed points, or strokes for differentiating foreground and background. Thus, we start with creating markers for the segmentation algorithm.

We generate markers for interaction based segmentation using the following conditions:

- First, we binarize the similarity score image shown in Figure 2.5b. We obtain regions like Figure 2.8a).
- We keep dominantly vertical regions by calculating the fitting ellipses [48] to the regions and later, using their orientations.
- We keep two big regions that have similar orientation with respect to image axes.
- We also check another condition to select two tire regions, having similar shape. We use the property of a tire as the height of a tire region is more than twice to its width.

After isolating these two regions, we obtain the markers around both tires using information from projection (Figure 2.2). Figure 2.8b shows the isolated tire regions from the binary image.

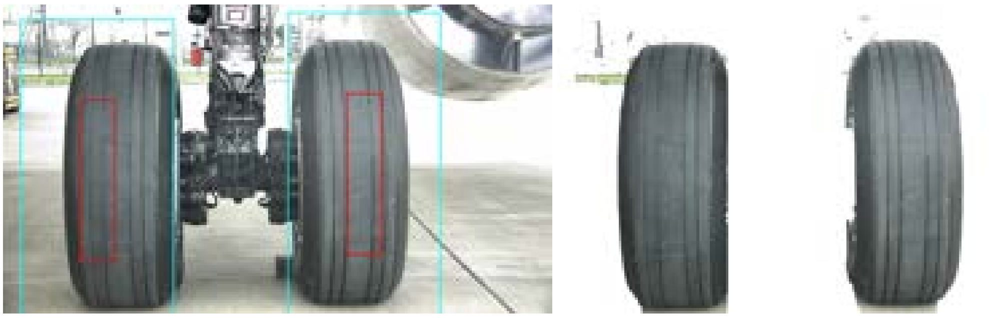


(a) Binarization of similarity score image shown in Figure 2.5b

(b) Locating the tires in the image to put markers for interaction based segmentation

Figure 2.8: Generating markers for interactive segmentation methods

To suite our requirement, we use GrabCut [9] segmentation method to segment the tires from the background. GrabCut segmentation method combines both the texture and the contrast information. It estimates the color distribution of the foreground and background using Gaussian mixture model [9]. We use three markers for GrabCut, **(1)** rectangle around the tires as probable foreground, **(2)** small rectangles on the tires to indicate certain foreground, and **(3)** rest of the image as background. Figure 2.9a shows the marker image for GrabCut segmentation. Using these markers, we get our output as in Figure 2.9b.



(a) Markers created for GrabCut segmentation

(b) Grabcut segmentation result

Figure 2.9: Segmenting tires from the image using GrabCut

After extracting the tires from the background, we remove the small disjoint outliers using morphological operations. First, we erode to disconnect tire and the outliers. Later, we do opening to remove small disjointed outliers. Finally, we dilate to retain to previous tire shape. Final image after removing noises and outliers is shown in Figure 2.10.

Additionally, we use Watershed [39,45,46] algorithm with markers generated in Figure 2.9a. Due to the presence of ambiguous similar color around the tire and the background, marker

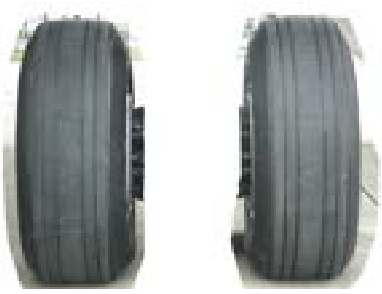


Figure 2.10: Final isolated tire portions from the image

based Watershed produce under-segmented results Figure 2.11.

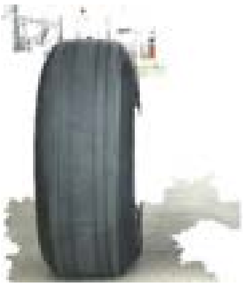


Figure 2.11: Tire image segmentation using Watershed algorithm

2.7 Inspection

This section illustrates the decision making step. The outcome should show the condition of the tires, whether they are worn out or not. The decision is carried out by examining the health (depth) of the treads of each tire. If the treads are faded, then the method informs the operator that the tire should be changed. We use ridge based intensity profiling of the tires to inspect the treads health.

In this section, we start working on Figure 2.10. We measure the gradient magnitude of the image to understand the homogeneity characteristics of intensity within the segmented regions. Figure 2.12 shows the gradient image.

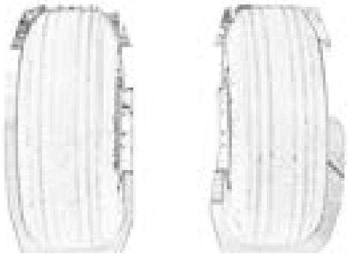
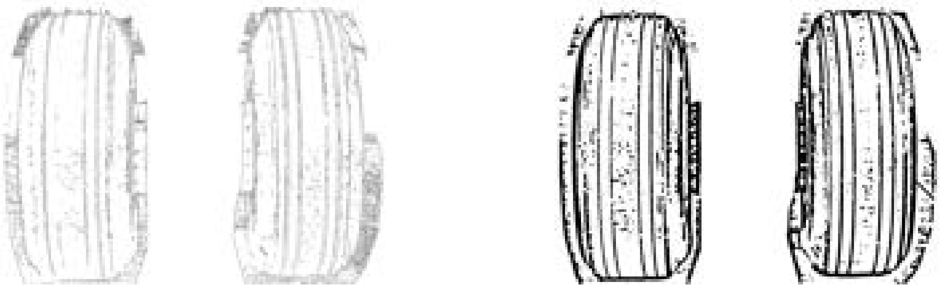


Figure 2.12: Gradient image to understand intensity changes along tire treads

Gradient image shows that there exist gradient changes along the treads. This observation leads us to selecting either edge or ridge detection for tread inspection. We know that edge detection enhances the significant intensity changes, the steepness of slope at each point. Thus, for our instances, edge detection results in having two fringes along each tread because we get two intensity changes (from high to low and again from low to high). Figure 2.13a shows why edge detection does not suit our problem definition. Ridges (intensity valleys), on the contrary, are useful to distinct out the treads from the tire (Figure 2.13b). We detected ridges using Frangi Vesselness filter [49]. He proposed to use the eigenvectors of the Hessian to compute the likeliness of an image region to contain ridge [49].



(a) Edge detection over the segmented image

(b) Ridge detection over the segmented image

Figure 2.13: Evaluating edge and ridge detection for our particular purpose

To avoid false alarms, we inspect only mid 80% of the tire region. We discard the upper and lower 10% of the tire region. Final ridges to inspect are shown in Figure 2.14.

To check the condition of the treads, we read intensity value of the ridges along the line. We use line iterator method of OpenCV. Using the profile line (red line in Figure 2.14), we scan the ridge image from top to bottom for each tire region. The lines to iterate are generated by finding the *left and right*-most extreme points for each of the tire regions. Later, their iteration is conditioned by the *top and bottom*-most extreme points of the regions. For each iteration, we receive one-dimensional signal with peaks at the presence of treads.

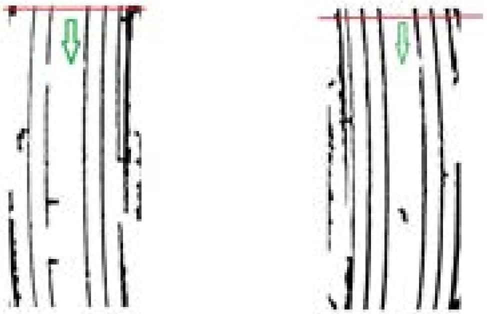


Figure 2.14: Final ridges and line iterating over the ridges to find the faded treads

If the known pattern of treads (for Airbus A-320, there should be four treads) is not found in any 1D array, the system reports a discontinuity of treads for that location. The line iterates till the end of the regions and count the size of discontinuities. The final decision about changing the tire depends on the size of the discontinuity. Figure 2.15 shows the final tire inspection result for our system.

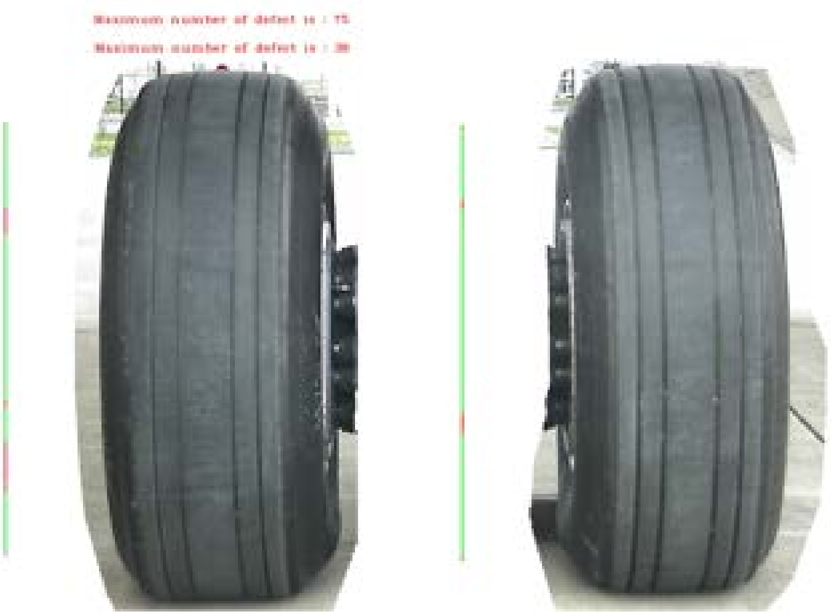

Maximum number of defect is : 75
Maximum number of defect is :

Figure 2.15: Final tire inspection result

2.8 Result

Figure 2.15 shows our inspection result where the red stars are indicating discontinuities in treads. The accuracy of our detection is 98.3% on our dataset of 118 images. Inspection approach has shown false negative rate (FNR) of 0% with acceptable false positive rate (FPR) of 14%. Inspection dataset contains 99 worn out tires and 132 tires in good condition.

Challenges that cause numerous false alarms are due to the presence of strong backlight, notably when the robot is placed under the fuselage and capturing images to the sun.

2.9 Conclusion

We present an original approach to detect and inspect airplane tires based on existing image processing techniques. Our experiment shows the tire region detection accuracy of 98.3% and inspection accuracy with 7% FPR and 0% FNR. Our proposed pipeline combines template matching and GrabCut segmentation which in most of the cases, is robust in partitioning out the tires only. Our inspection model, however, can only inspect the tire treads along the frontal tire surface and is not yet robust to inspect the treads on the very top and very bottom part of the tire. This is to avoid false-alarms since sometime our segmentation results contain outliers.

Our method shows a sequence of operations to detect and inspect tires which has significant influence in airplane industry. We are currently working on improving our result by increasing the dataset and evaluating binary classification techniques.

Chapter 3

Wheel Chock Detection

3.1 Introduction

Another important part of walk-around is to check the presence of chock before tire inspection. The object of this task is to detect chock in front of a wheel while doing the tire inspection on the captured images. Thus, this system works in parallel with the tire inspection process. It uses the tire segmentation (Sec. 2.6) result, to detect the wheel chock.

Figure 3.2a is an example where one wheel has a chock in front of it and another one is free. Generally, wheel chocks are of rectangular shape. Observation suggests that if there is any chock then we can find it by looking at the lower portion of a tire. Our system begins with narrowing down the search space to the lower portion of the segmented regions. Otsu [50] and Adaptive thresholding [51] have been utilized to disregard any other objects from region of interest. It thus, becomes convenient for the system to search around the lower wheel area.

Later, the system detects lines using Hough transform (HT) (Sec. A.1) and Probabilistic Hough Line transform (PHT) (Sec. A.1) in the narrower window. Our system uses the results from HT to confirm the presence of a wheel chocks.

In Section 3.2, we illustrate on related works. The framework of our developed system is shown in Section 3.3. Detection model is discussed in Section 3.5. Finally, we conclude (Section 3.7) after showing our experimental results in Section 3.6.

3.2 Related Works

Visual detection of airplane wheel-chock, is a very particular problem. No prior work has been done before on this very specific topic. But, researchers worked on similar problems where they detect rectangular shapes. Thuy Tuong Nguyen et al. [52] used standard hough transform on edge features to detect rectangular objects. Yangxing Liu et al. [53] detected line segments and then, used Markov Random Field to find rectangular shaped objects. Both the works used extensive searches although they defined limited search window. In contrast, we look for only PHT lines rather than rectangular shape after creating a limited search window. This speeds up the implementation for real-time application.

3.3 Developed Framework

Our developed system detects the wheel chocks using image processing techniques. We develop a pipeline of image processing techniques to accomplish the task. The flowchart in Figure 3.1 shows the steps.

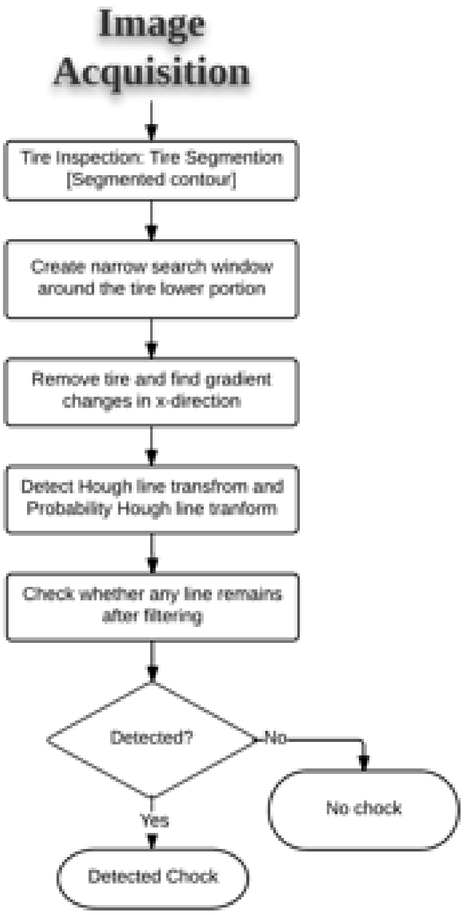
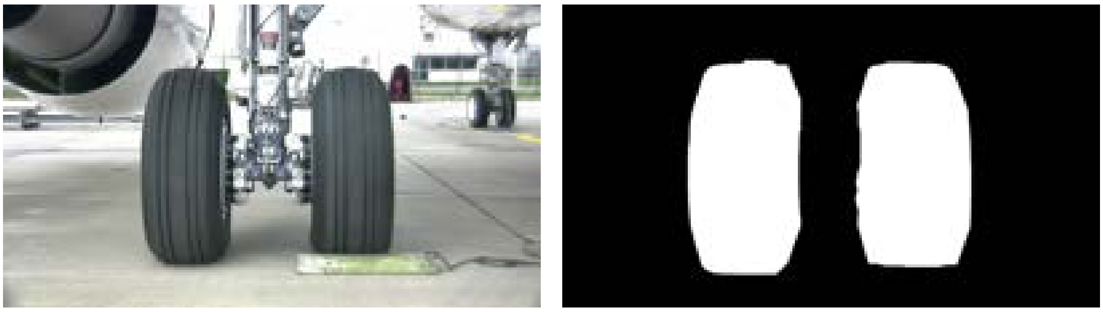


Image Acquisition
Tire Inspection: Tire Segmention [Segmented contour]
Create narrow search window around the tire lower portion
Remove tire and find gradient changes in x-direction
Detect Hough line transfrom and Probability Hough line tranform
Check whether any line remains after filtering
Detected?
No
Yes
No chock
Detected Chock

Figure 3.1: Work flow pipeline for wheel chock detection

3.4 Dataset

We test our method on the reduced tire inspection dataset. The dataset contains 109 images with 218 tires. Our system also uses the corresponding segmented contour images from tire inspection module (Sec. 2.6). In our dataset, we have 18 tires with chocks. Figure 3.2b shows an example of segmented contour image.



(a) Main image for chock detection

(b) Segmented tires

Figure 3.2: Tires segmentation

We use Figure 3.2a in the later sections to demonstrate our implementation.

3.5 Detection

First, we create a small search window to look for chocks. To create the window, we use the contour image from Figure 3.2b and find the center point of the spatial contour using the first contour moment, discussed in Sec. A.1. Later, we create the window around the lower half of the contour with few additional lower pixels so that we get the area where the chock can be in the image. The result image is shown in Figure 3.3.

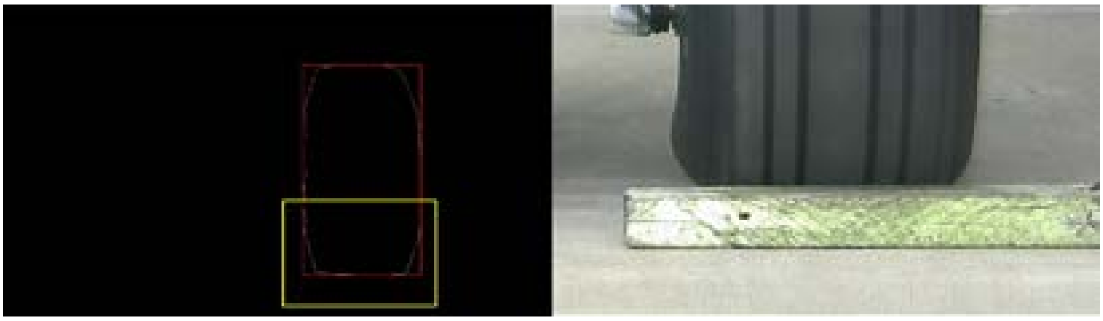


Figure 3.3: Narrow window to search for chocks

The narrowed image is used for further inspection. However, we do few preprocessing steps before searching for chock lines. In preprocessing steps, our system tends to remove the tire from the image and to keep only the chock to ease the detection. We do two-step thresholding, one to detect the contact edge between tire and chock (or ground), and another one to remove the tire portion. **(1)** We use Adaptive Gaussian threshold to binarize the image. In Adaptive Gaussian thresholding, the threshold value is the weighted sum of neighborhood values where weights are set by Gaussian window. This is to find the contact edge between the tire and the chock (or ground). Figure 3.4 shows thresholded image.

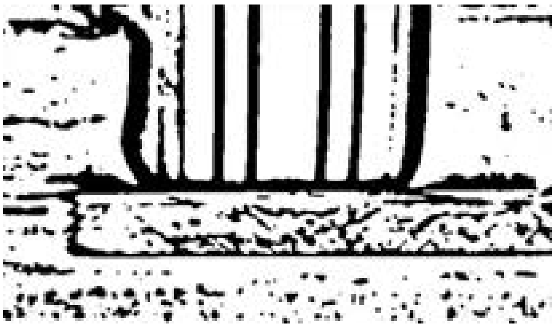


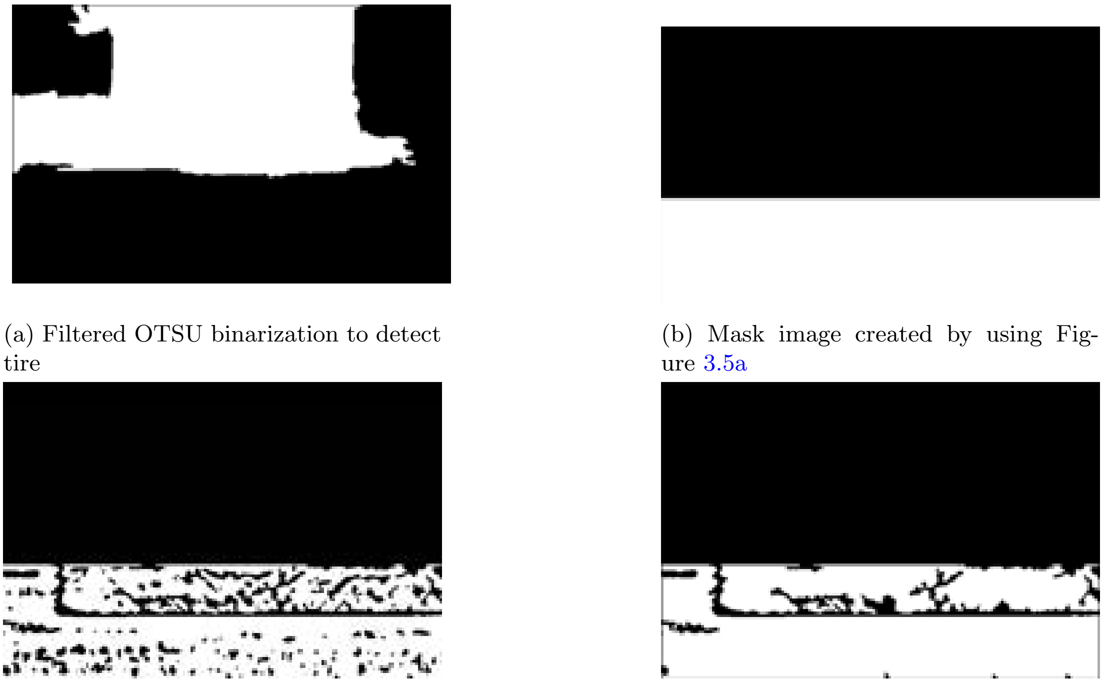
Figure 3.4: Binarized image using Adaptive Gaussian threshold

(2) OTSU [50] thresholding is used on the Figure 3.3 to detect the tire. After binarization, we remove small noises from the thresholded image using Median filter. Finally, we create a mask image from the detected tire portion size and use it to remove the tire from Figure 3.4. Figure 3.5 shows the steps.

After removing tire, we search for any lines in the rest of the image as we know that chock has horizontal rectangular box shape. To simplify and restrain HT to detect only the chock corresponding lines, we first use Sobel operator [54,55]. We keep only exact horizontal edges by subtracting the y-direction gradient image from x-direction gradient image. Figure 3.6 shows the result.

The gradient difference image sometimes gives us noises along the bottom border of the image. These noises are discarded since we know that the chock cannot be at the border of the image. After these preprocessing steps, we detect line segments by employing PHT [56] on the same masked gradient difference image. Figure 3.7 shows the results of PHT.

However, PHT detects lines at the edge of tire and chock (or ground, when there is no chock). Thus PHT lines at the contact edge of tire are not considered. Finally, we check in the result image whether there is any remaining line. If yes, then we know that the line is from chock and we conclude that there is chock in front of the tire. The result image, Figure 3.8 shows the decision.



(a) Filtered OTSU binarization to detect tire

(b) Mask image created by using Figure 3.5a

(c) Keeping only the lower portion of Figure 3.4

(d) Filtering the image from Figure 3.5c: median filter, removing small regions

Figure 3.5: Steps we follow to remove the tire and keep only the lower portion

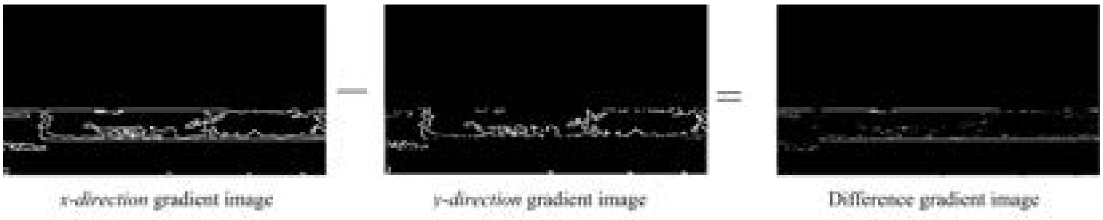

x-direction gradient image
y-direction gradient image
Difference gradient image

Figure 3.6: Gradient difference image to detect chock line

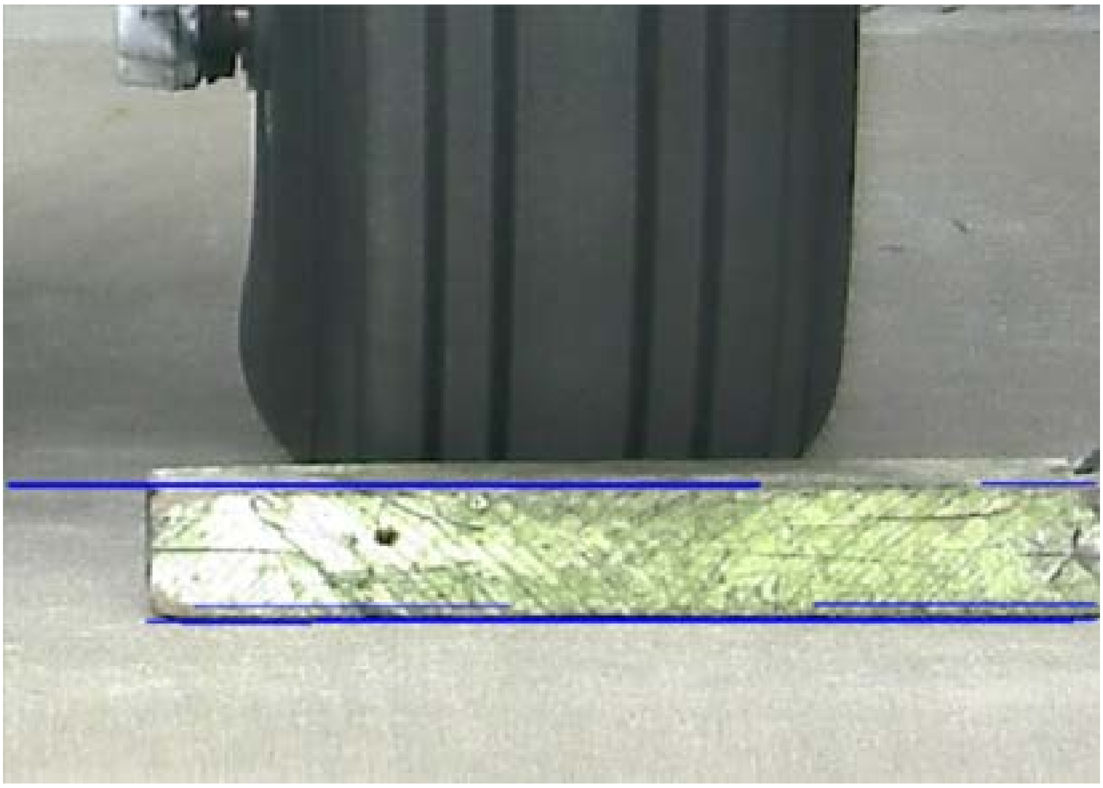


Figure 3.7: Probabilistic hough line detecting wheel chocks

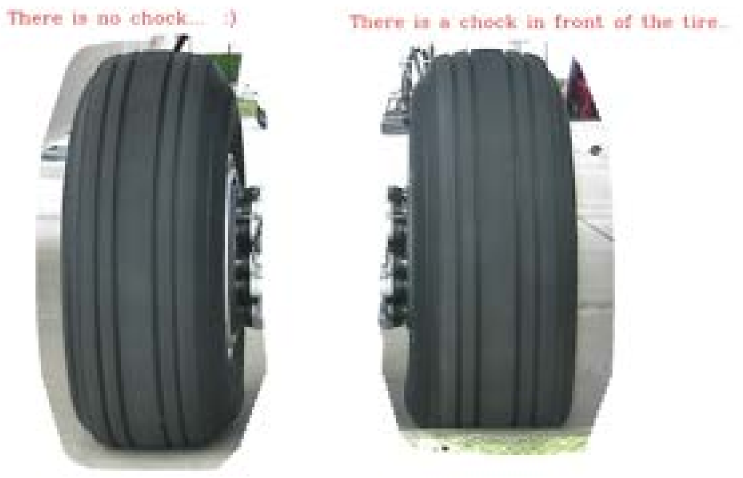


Figure 3.8: Final decision result

3.6 Results

Figure 3.8 shows our inspection result where we detected the presence or absence of the wheel chock. Chock detection has shown false negative rate (FNR) of 0% with false positive rate (FPR) of 5%.

Challenges we experience in detecting wheel chocks are noises. Rarely, we get straight marking lines on the tarmac ground for which we get false alarms. Another challenge that we face is the chock color. Occasionally, black wheel chocks can be used which are of exactly the same color as tire. This creates problem both in tire segmentation and subsequently, in finding the border edge between tire and chock.

3.7 Conclusion

Our implementation is suitable for industrial use. It is fast enough for a robot to detect a wheel chock while it is performing tire inspection, from the same ground checkpoint. Our approach has high detection accuracy although with acceptable false-positive rate.

We are working on making our solution more robust, robust to noise and color. In future, we would like to use machine learning to detect wheel chocks but, beforehand, we would work on creating bigger dataset with lot more variations.

Chapter 4

Engine Inspection

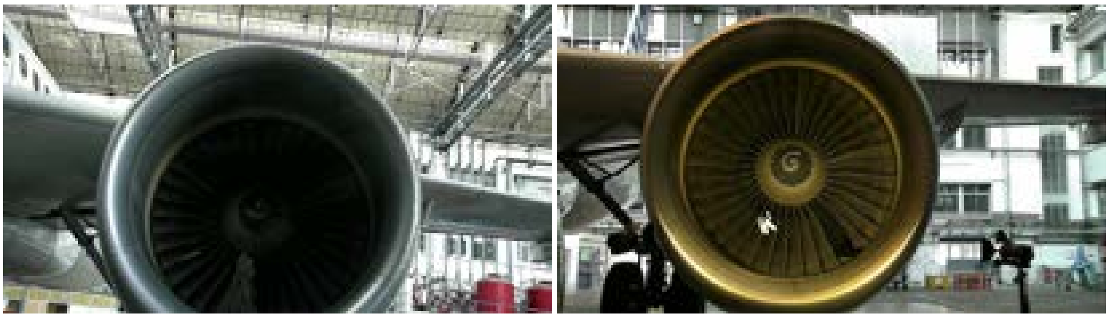
4.1 Introduction

One of the most important inspection tasks in the pre-flight checklist is to inspect the engine. Engines are inspected carefully before or after each flight to make sure that they are fine for next use. Basically, visual engine inspection looks for problems in the engine turbine blades. It aims for finding any of the two main problems, **(1)** is there any unwanted visible object stuck between blades, and **(2)** is there any blade missing. The directive toward solving these problems comes from the structure of the engine. The structure of the engine is elaborated below and shown in Figure 4.1b.

Structure of engine.

- The engine has round shape. This is a very important property that can be used to detect an engine.
- Airbus A-320 engine has 36 continuous blades. Each blade makes 10° angle difference with it's next and previous blade. This very distinct shape has major importance in inspection.
- The painted swirls on the engine spinner is also an important mark to find the engine center.
- There exist color difference between the engine cowling and the blades. This can have use in detecting the contour or border line between the cowling and blades.

To initiate engine inspection, our robot captures engine images using the PTZ camera from particular (conventional for engine inspection) walk-around point. Although, the scale of the object (engine) is variable to the noisy localization of the robot with respect to airplane. Since the engine blades are inside cowling, surrounding lighting has vital impact on engine inspection. Figure 4.1 shows the impact of lighting on visibility. Figure 1.4 shows an example where the blades are well illuminated and free of object.



(a) Badly illuminated engine blades with objects (b) Well illuminated engine blades with objects

Figure 4.1: Sample engine images

Inspection of engine, thus, comprises solving two different tasks: **(1)** detecting engines from the image and **(2)** later, inspecting the blades. To detect the engine, feature detection technique has been used and to inspect the blade condition, signal processing techniques has been used.

This work has one main contribution, inspecting engine blades using Fourier Transform. The developed system works robustly and is invariant to illumination and scale. The system begins detecting engine cowling using Hough Circle transform. The size of the engine is approximated from projection beforehand to help detection. The projection (Section 1.3.1) of the engine cowling has been done by utilizing the rough estimation of the robot pose and the known position of the camera on the robot. Figure 4.2 shows the engine cowling projection onto image plane.

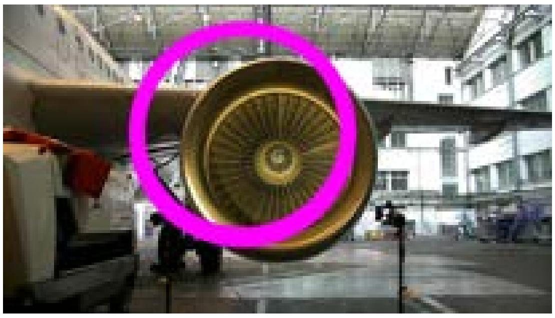


Figure 4.2: Projection of engine cowling

Before inspection step, the detected engine images are filtered to remove noises and compensate lighting changes. This step substantially counterbalance the illumination problem. To inspect the engine blades, the system reads the intensity values across the blade region, between cowling and spinner, by generating multi-circle paths. Figure 4.3 shows an example of multi-circle paths to read pixel values.

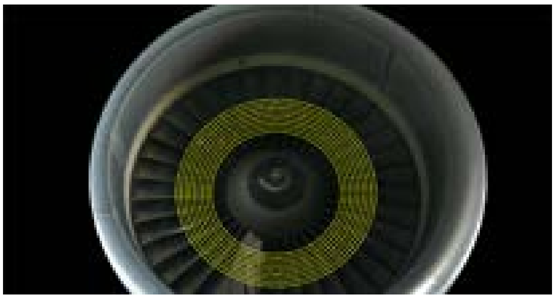


Figure 4.3: Multi-circle paths to read pixel values for inspection

Our system reads the pixel values along each of the circles individually. Each of the intensity readings, *i.e.* 1D signal is divided into n equal-sized sections. Then, we compute the Fourier Transform for each of n sections of each 1D signal. Finally, principle component frequency of each of the n sections is calculated. This step goes for all the circular iterator readings. Because of the number of the windows for each circular iterator, the principle component value for each of the sections is constant for non-problematic engine blades and different for problematic engines. In favorable case, the number of peaks is $\frac{36}{n}$ which is directly related to frequency. The system shows the location of the problems which can be missing or broken blades or any object stuck within the blades.

We discuss about related works in Section 4.2. The framework of our developed system is shown in Section 4.3. Later, detection model is discussed in Section 4.5. In Section 4.6, we describe the inspection model. Finally, we conclude after showing our experimental result in section 4.7.

4.2 Related Works

Computer vision based inspection of aircraft engine blades is getting more attention in current time. Richard C. Thomson et al. [57] invented a manual in-flight inspection system using CCD cameras mounted near the airplane door to inspect engine blade rotation speed. This work was interesting but it can not find any defect automatically and needs co-pilot interaction. While offering automatic defect detection is one of the main tasks of our implementation.

Other works on turbine blades inspection were using separate instruments in industrial setting to inspect blades [58–60]. The purpose of these works was to inspect blades before they get installed onto a plane. Dirk Padfield et al. [58]] developed “Aid to Visual” system which could inspect High pressure turbine disk using contour template matching, although they did registration of the captured images. [59, 60] used respectively, infra-red imaging and X-ray CT system to inspect defect in turbine blades. They inspected turbine blades in production line setting before installation on the airplane. In contrast, our implementation works faster, and inspects engine blades during service time.

Inspecting wind turbine blades visually [61, 62] has similar purposes. Martin Stokkeland et al. [61] in his thesis offered visual wind turbine inspection using Hough transform. Huiyi Zhang et al. [62] used edge detection and Sobel gradient operator to detect cracks on wind turbine blades. Our method works robustly even though our working environment throws more challenges due to complex surrounding.

4.3 Developed Framework

Our method consists of two main steps: *detection* and *inspection*. The flowchart in Figure 4.4 shows the steps.

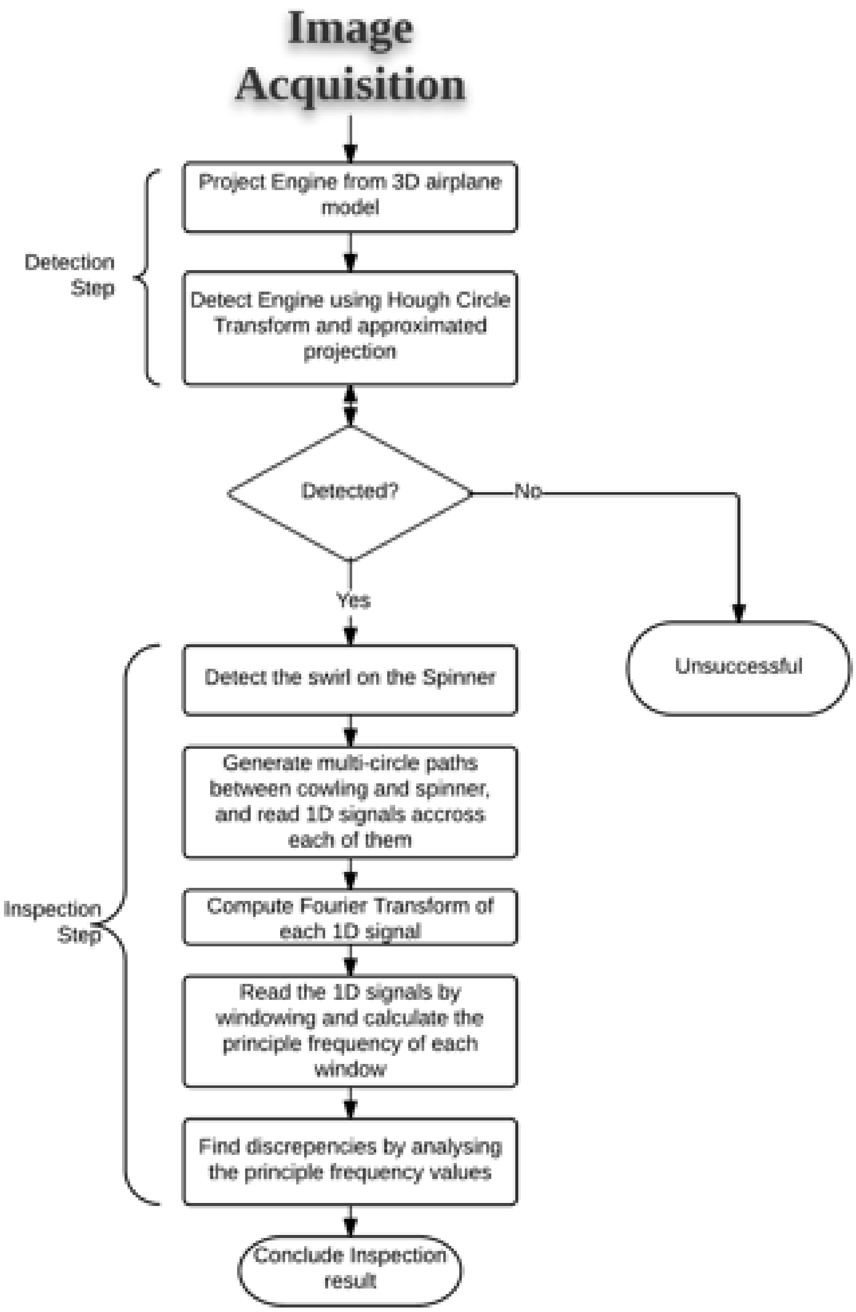


Image Acquisition
Project Engine from 3D airplane model
Detect Engine using Hough Circle Transform and approximated projection
Detection Step
Detected?
No
Yes
Unsuccessful
Detect the swirl on the Spinner
Generate multi-circle paths between cowling and spinner, and read 1D signals accross each of them
Compute Fourier Transform of each 1D signal
Read the 1D signals by windowing and calculate the principle frequency of each window
Find discrepencies by analysing the principle frequency values
Inspection Step
Conclude Inspection result

Figure 4.4: Work flow pipeline for engine inspection

4.4 Dataset

Our sample dataset for engine inspection is based on a single Airbus A-320 airplane. We imaged only one real engine. The dataset contains 104 images for inspection. We put different objects within the engine blades region to simulate challenges and use an external light source to vary illumination. All the images are captured from the approximated walk-around point of the robot and thus, with maximum possible perspective variations. Figure 4.1 and Figure 1.4 show images with different light, scale, and damage conditions. We use the following image, Figure 4.5, throughout this chapter to illustrate our implementation.

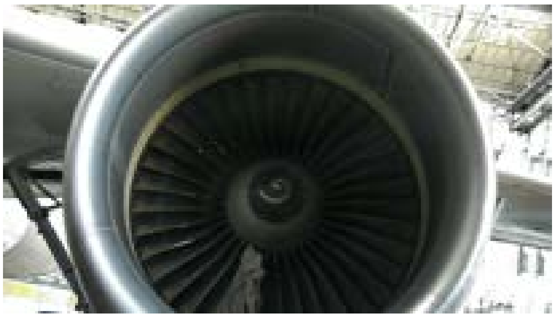


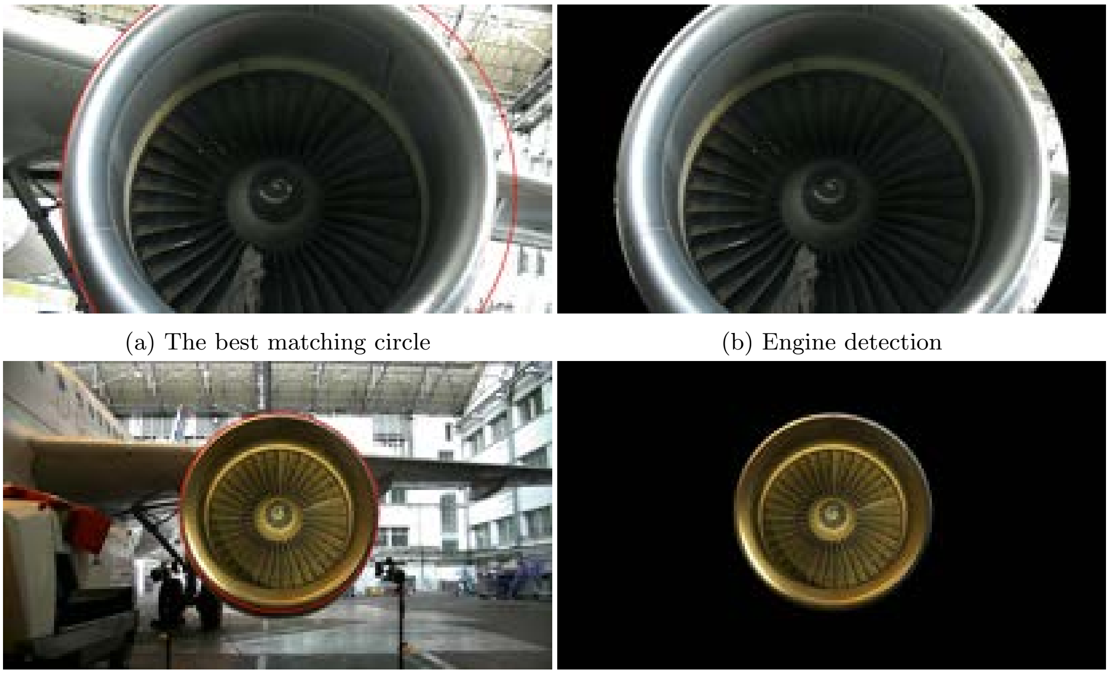
Figure 4.5: Image we use to demonstrate our developed system

4.5 Detection

In the images, the engine cowling is a big detectable circle. This suggests us to begin with hough circle transform [63] to detect the engine.

We use the Hough gradient method [64] (OpenCV implementation) to improve the efficiency of hough circle transform. Before hough circle extraction, we applied Gaussian filter [51] to remove noises from the images.

Additionally, we restrict hough transform to detect only the circles that fit best with the engine. To achieve this, we project 3D model (Figure 4.2) of the aircraft to approximate the engine size. The system uses this approximate size and detects only the best matching circles with the highest number of votes. Finally, we mask out the engine portion for inspection purpose. Figure 4.6 shows the detection result for our reference Figure 4.5.



(a) The best matching circle

(b) Engine detection

(c) The best matching circle

(d) Engine detection

Figure 4.6: Sample tire images

4.6 Inspection

Inspection step comprises four important tasks. All of them are discussed below.

Contrast Enhancement to improve illumination. In our images, blades are usually not well illuminated being placed deep inside the engine inlet. Thus, before passing through inspection step, we compensate the illumination by doing histogram equalization. Histogram equalization is a common technique to do contrast enhancement. It works in the following steps,

- Calculates the probability of having a pixel intensity value in the image by using,

$$P(i) = \frac{n_i}{n}$$

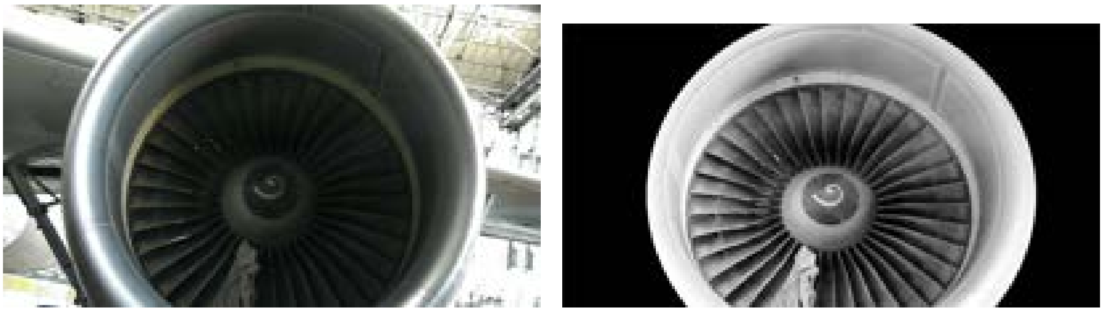
  here, n is the total number of pixels in the image and n_i is the number of occurrence for an intensity value.

- Computes the Cumulative distribution function as,

$$cdf(i) = \sum_{j=0}^{i} P(j)$$

- Later, remaps the new image by multiplying the cumulative distribution values with the desired intensity level range.

Histogram equalization works well when an image has similar darkness or brightness in both foreground and background. For our images, we have leptokartic distribution [65] of dark intensity values and hence, histogram equalization works nicely in equalizing this distribution. Although histogram equalization has indiscriminating effect, *i.e.* it increases contrast of both the image and the background noises, we reduce this effect by removing noises before histogram equalization. We use Gaussian low pass filter [51] to remove noises before applying histogram equalization. Our final contrast enhanced image is shown in Figure 4.7b.



(a) Main image

(b) Engine image after histogram equalization

Figure 4.7: Sample tire images

Detection of the engine center. We work with contrast enhanced image to detect the center point of the turbine blades. For this purpose, we consider the second property from 4.1, 10° angle difference between two neighboring blades. The center point is important to generate the multi-circle paths to extract pixel values across the blade regions.

To find the center of the blades, first, we detect edges of Figure 4.7b using canny edge detector [66]. Further, we detect the first two best fitting (with higher votes) lines with 10° angle difference between them using Hough line transform. Hough line transform is discussed in Section A.1. The intersection of these two lines defines the center of the engine blades. We find the intersection of these two lines in the following way.

- Let, we have two line segments. Line 1 is from (x_1, y_1) to (x_2, y_2) and line 2 is from (x_3, y_3) to (x_4, y_4).
- Now, we compute the slope of the lines. For line 1, slope is, $m_1 = \frac{dy}{dx} = \frac{y_2-y_1}{x_2-x_1}$ and for line 2, the slope is, $m_2 = \frac{dy}{dx} = \frac{y_4-y_3}{x_4-x_3}$.

- Constants for the two lines are then computed. For line 1, $c_1 = y_1 - \frac{y_2-y_1}{x_2-x_1} \cdot x_1 = y_1 - m_1 \cdot x_1$ and for line 2, $c_2 = y_3 - \frac{y_4-y_3}{x_4-x_3} \cdot x_3 = y_3 - m_2 \cdot x_3$
- Subsequently, the x-intersection point is, $x_t = \frac{c_2-c_1}{m_1-m_2}$ and the y-intersection point is, $y = \frac{y_2-y_1}{x_2-x_1} \cdot x_t + c_1 = m_1 \cdot x_t + c_1$.

The result from finding blade center is shown in Figure 4.8.

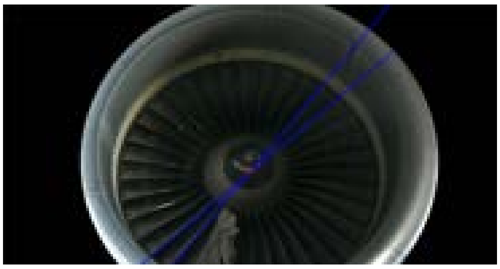


Figure 4.8: Detecting the spinner center point

Inspection. We use the coordinate of turbine blades center and generate ten circle paths with continuous increasing radius as shown in Figure 4.3. The circles start from the border of the spinner and radius increases toward the outer border of cowling. Further, we extract pixel values along each circle-path from the blades region and for each circle-path reading, we form a 1D signal that includes intensities on the path. The size of the 1D signal depends on the number of pixels in the circle-path.

Each 1-D signal is then smoothed with a Gaussian filter [51] in order to eliminate small details. We use Discrete Fourier Transform to find any problem in the engine blades. From Fourier transform [51, 67], we know that it can be used to check periodicity of any signal.

When there is no defect, a signal gets close to periodic and in the case of defect, we usually get a breach in the periodicity. We perform windowing to analyze the each 1D Fourier transform. We define the number of times to do windowing, n and then, compute window size for each 1D signal, $\frac{numOfPixel}{n}$. Since, each engine has 36 blades at 360° span, in each window we inspect $\frac{36}{n}$ blades at a time.

By Fourier analysis, we compute the frequency of the highest magnitude component within each sliding window moved along the 1-D signal. In the favorable case, the computed frequency remains constant as the sliding window moves and can not find any defect. If there is a defect, than frequency changes in particular sliding window in the region of the defect. [3]. Figure 4.10 shows the effect of defects on 1D signal and the Figure 4.9 shows the 1D signal when there is no defect. Figure 4.11 shows the final result from frequency analysis. Red zones show foreign objects or defects and green zones show no-defect.

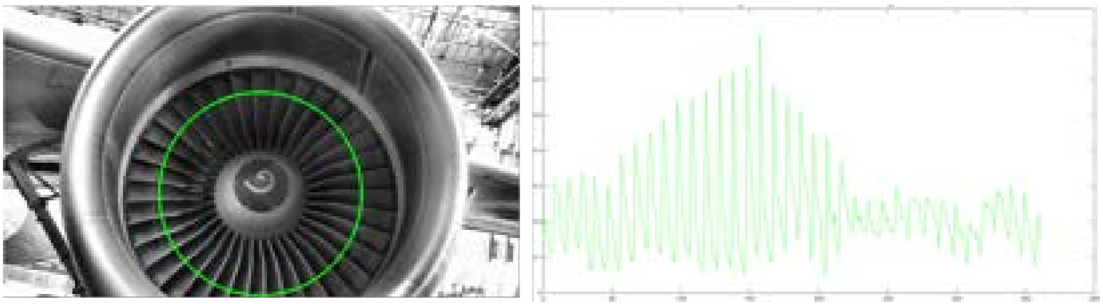


Figure 4.9: Example of 1-D signal read from the input image with no defect

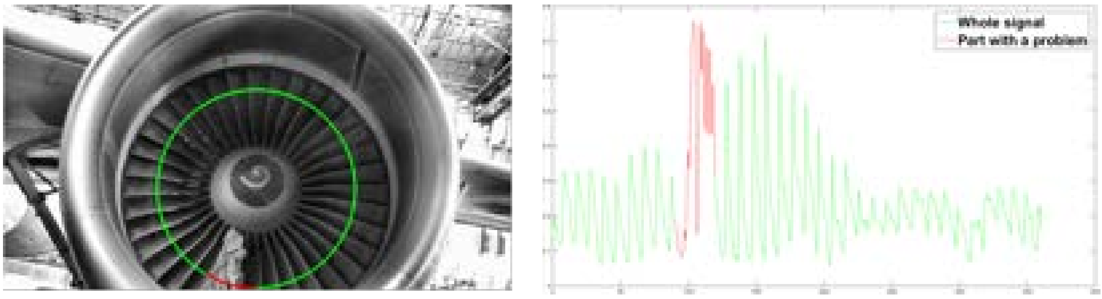


Figure 4.10: Example of 1-D signal read from the input image with defects

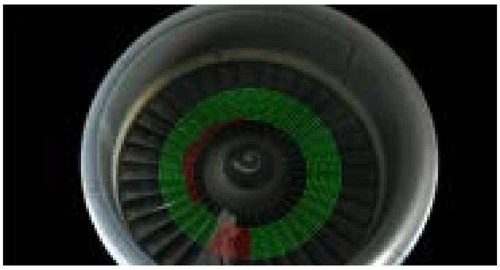


Figure 4.11: Final inspection result

Fourier transform is also susceptible to noises, ex. if we get extra lights or color that may look like an object. Thus, we also get some false alarms in the case of non-illuminated regions and in the regions where details behind the blades are visible.

Alternatively, we used texture based SIFT feature [68–70] extraction with SVM [71] classifier to classify defective and non-defective engines. But, it do not give satisfying result because the turbine blades do not have much texture on their edges. And mostly, the blade edges were not detected as SIFT features. Figure 4.12 shows SIFT key-points for our image.

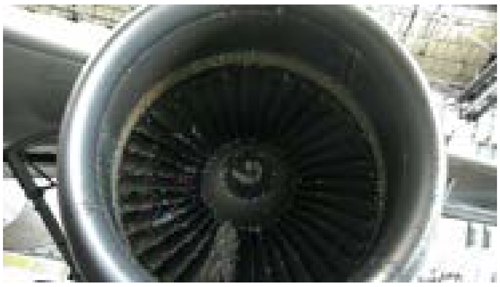


Figure 4.12: SIFT feature extraction

4.7 Result

Figure 4.13 shows our inspection result where the red portions show the defects or foreign objects. The accuracy for our detection is 100% on our dataset of 104 images.

Challenges we are experiencing in inspecting engine blades are due to the presence of strong back-light. Sometimes, these back-lights show-up in a way where they look like a defect.

We are currently working on reducing false alarm rate by discarding defects where we would detect blade edges (Figure 4.13). For detecting blade edges, we use the probabilistic hough line transform [56], A.1.

4.8 Conclusion

Our approach toward inspecting aircraft engine is original and fast enough to be implemented for real-time application. Our experiment gives high detection accuracy and low False-positive rate. We introduce Fourier analysis to find harmony in geometrical structure, *i.e.* turbine blades.

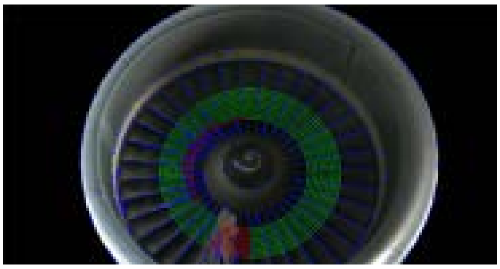


Figure 4.13: Extracted lines superimposed on inspection result from Figure 4.11

Our problem is still susceptible to noises, where we are still working. In future, we would like to improve the inspection efficiency by reducing FPR.

Chapter 5

Pitot Cover Detection

5.1 Introduction

One of the most important item-check during walk-around is to confirm that all the pitot sensors are uncovered to operate. Pitot sensors are used to measure the air-speed of an airplane during the flight. Subject to the requirement, the main objective of this implementation is to detect pitot cover, if present, otherwise validate that all the pitot covers are already removed. Images of pitot sensors are taken by using PTZ camera while the robot is approximately positioned at the specific walk-around point. Figure 5.2 shows an example image where one pitot sensor is covered and another one is open.

This specific problem is an object detection problem. To detect the pitot cover, we observe their appearance in the images. The pitot covers are always red colored and have a rectangular shape. Also the shape and the color cues are clearly identifiable. Hence, we use the basic intuition towards solving this particular problem, color based detection. Later, from the detected object(s), we search for specific one(s) that has approximate shape to the pitot cover.

Related works are briefed in Section 5.2. Later, the framework of our developed system is shown in Section 5.3. Detection model is discussed in Section 5.5. Finally, we conclude in Section 5.7 after showing our experimental result in section 5.6.

5.2 Related Works

In the literature, to our knowledge, there is no work that concerns the aircraft pitot cover detection. Basically, this is very specific task with a particular goal. There are works related to object detection using color, shape, texture, and other cues. Theo Gevers et al. [72] proposed new color model to detect multi-color objects. Feng Wanli et al. [73] proposed HSV-color space based vehicle license plate detection. They also used gradient measurements to locate the license plate. There are survey papers [74] comparing different color based segmentation. For our work, we also use HSV-color space to detect pitot cover which generally is red.

Use of shape to detect an object is also a very popular and instinctual approach. DK Banerjee et al. [75] used shape matching to detect a 3D object. They considered translation and rotation for shape matching. Use of both shape and color for better object detection is also popular in research community. Anil K. Jain et al. [76] proposed a general color and shape based object detection for better performance. For our specific task, we also used both color and shape to detect pitot cover from the captured images.

5.3 Developed Framework

We develop a pipeline of image processing techniques to accomplish the task. The flowchart in Figure 5.1 shows the steps.

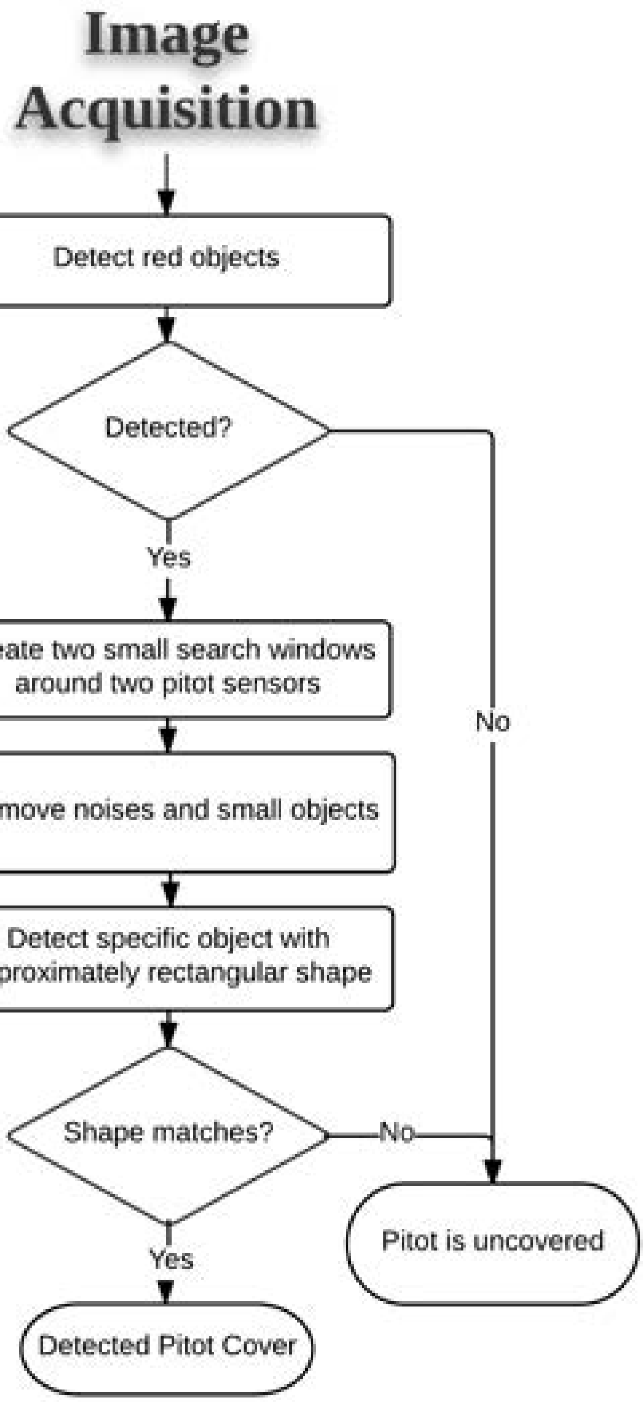


Image Acquisition
Detect red objects
Detected?
Yes
No
Create two small search windows around two pitot sensors
Remove noises and small objects
Detect specific object with approximately rectangular shape
Shape matches?
No
Yes
Pitot is uncovered
Detected Pitot Cover

Figure 5.1: Work flow pipeline for pitot cover detection

5.4 Dataset

Our dataset is created on a single airplane. We have 42 RGB images in our dataset. All the images are captured from the approximated walk-around point of the robot. We use Figure 5.2 to demonstrate our implementation.

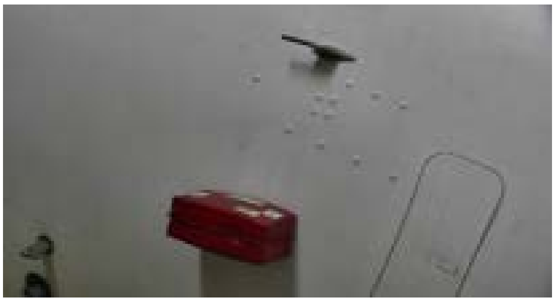


Figure 5.2: Reference image to show the implementation

5.5 Detection

First, the system uses color based image segmentation in *hsv-space* to segment out red object(s) from the image. HSV-color space has cylindrical geometry instead of RGB cube. It has three channels: *hue, saturation,* and *value.* The main advantages of using HSV-color space is that *hue* defines more perceptual colors. Figure 5.3 shows HSV-color space.

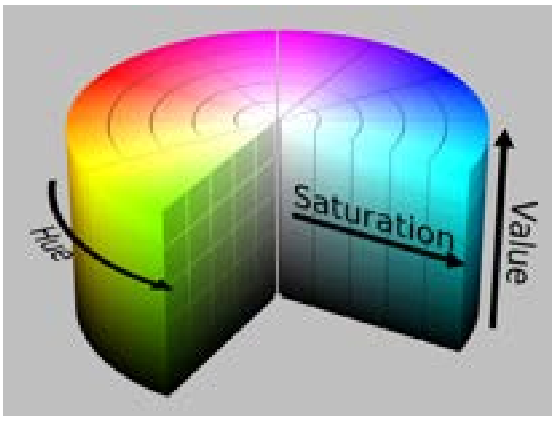


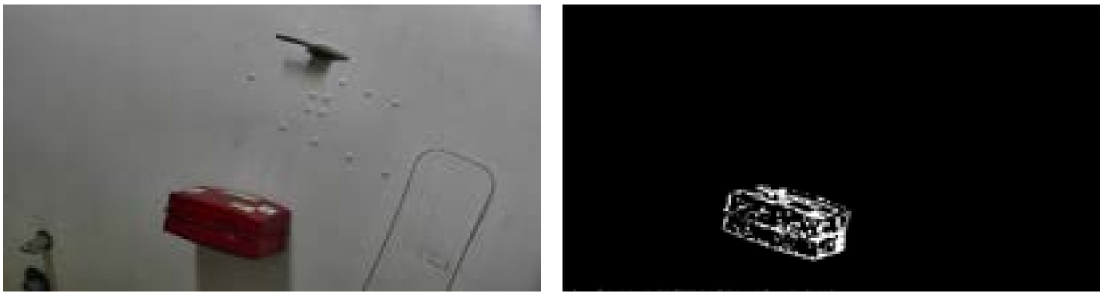
Figure 5.3: HSV-space cylindrical representation [2]

From Figure 5.3 we can find that *saturation* level determines color intensity as lowest saturation give gray level image. *Hue* represents color in 360°, starts with red at 0° and rewind back to red after 360°. And *value* defines the brightness of the color.

Beforehand, we use Bilateral filter [8] to remove noises from the image. Since we implemented our system in OpenCV, we need to use the *HSV* scales for OpenCV. In OpenCV, for *HSV*, *hue* range is [0, 179], *saturation* range is [0, 255], and *value* range is [0, 255] [77]. For red color, OpenCV has hue value approximately in range between [0, 10] and [160, 180]. We utilize only *hue* value to detect red color objects. We set our lower and higher ranges for red color as following,

Low range (h, s, v): [(0, 20, 30), (10, 255, 255)]
High range (h, s, v): [(160, 70, 100), (180, 255, 255)]

The thresholded image is shown in Figure 5.4b.



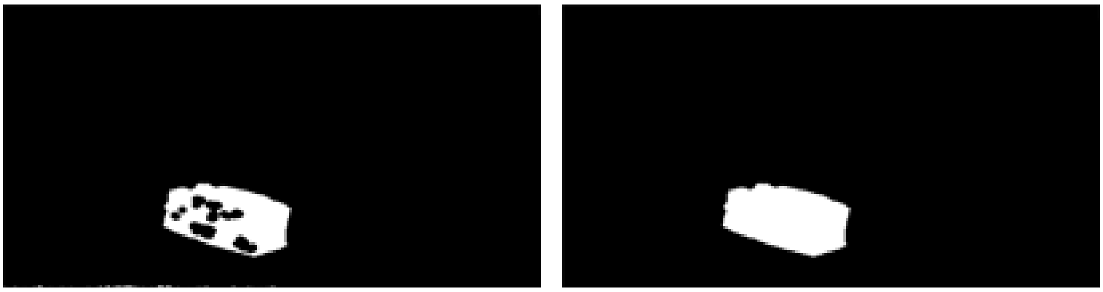
(a) Main reference image

(b) Detected red objects

Figure 5.4: Detecting red objects from the image

Thresholding, based on red color gives us noisy detection which contains disconnected components (Figure 5.4b). But, for same object, these disconnected components stay close in the same neighborhood area. In the next step, we connect these neighboring disconnected components using morphological operations. We do closing followed by opening and use same structuring element for both operations. We use (7×7) elliptical structuring matrix. Figure 5.5a shows the result after successive morphological operations. Later, we remove small area components. Since, the objects still may have holes which is not convenient for shape matching, we use flood fill, (Section A.2), to fill the whole component area. This results in a full big connected component for each pitot cover (if any). Figure 5.5b shows the final detected pitot cover.

After detecting only red objects, we match the detected objects with an approximated known mask shape. Before that, we narrow down our search area to look for objects. It reduces computational time which is required for real-time implementation. To create narrow search windows, we use the approximate location of the pitot from the projection. We have simplified 3D model of Airbus A-320 model aircraft and we do approximate projection from there,



(a) Connecting the disconnected detected red components which belong to the same object

(b) Final contours of the detected red objects after flood fill

Figure 5.5: Detection of red objects from Figure 5.4a

explained in 1.3.1. Figure 5.6 shows the search windows.

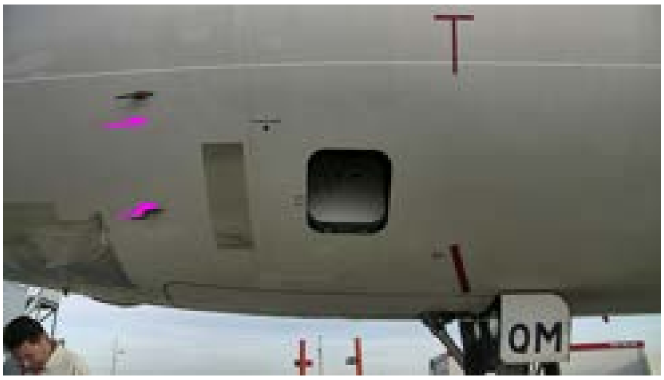


Figure 5.6: Pitot tube projection

We, then, filter out the contour images using the mask windows from Figure 5.6. The result images are shown in Figure 5.7.

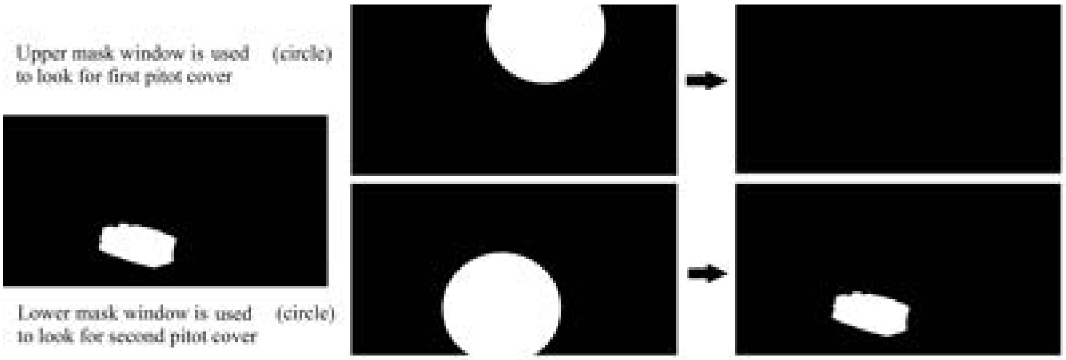

Upper mask window is used to look for first pitot cover
(circle)
Lower mask window is used to look for second pitot cover
(circle)

Figure 5.7: Results from using windows to search for pitot cover

Finally, to validate any detected red object as pitot cover, we search for rectangular shape in each of the two windows (if they contain any object). We match the shape of existing objects with an approximately simulated rectangle of size 350 × 200. We do shape matching using Hu moment [78] explained in Section A.2. Our system validates a shape as pitot cover when the matching value is higher and do vice-versa when the matching value is lower. Figure 5.8 shows the rectangular reference shape and Figure 5.9 shows the final decision result.

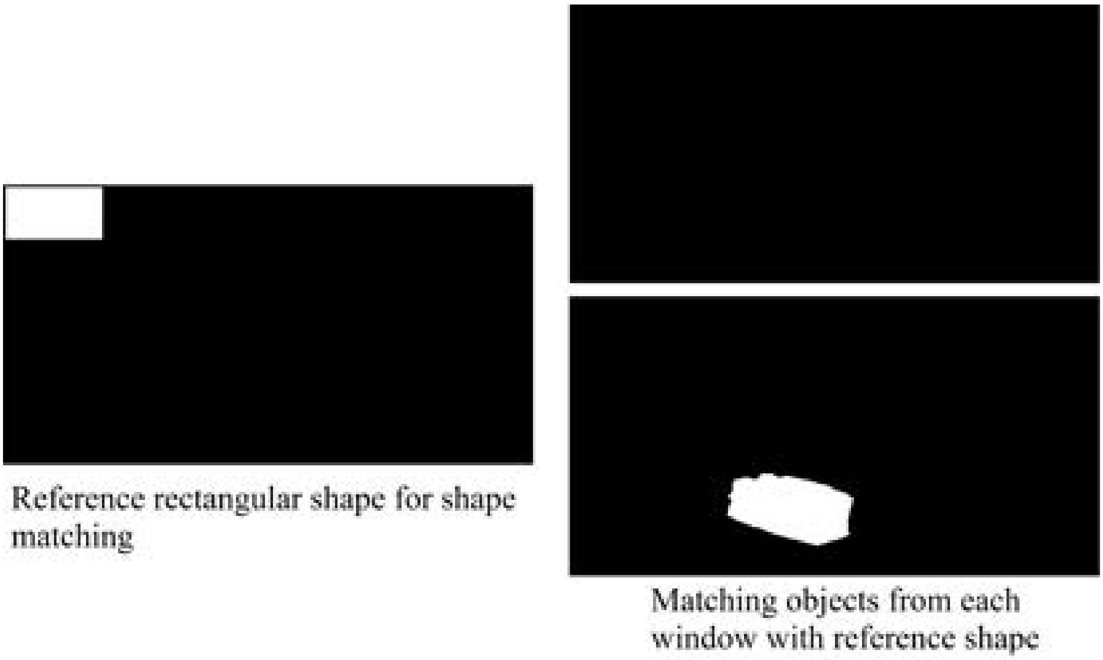
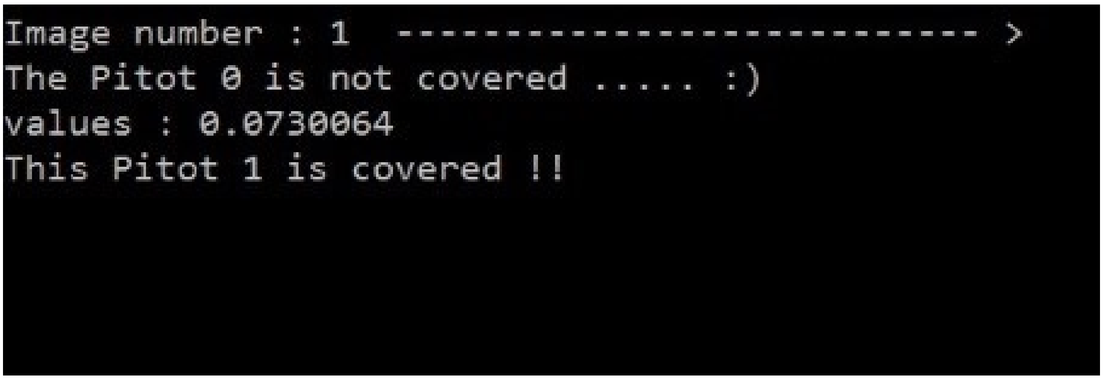

Reference rectangular shape for shape matching
Matching objects from each window with reference shape

Figure 5.8: Reference rectangular shape for shape matching



```
Image number : 1  -------------------------- >
The Pitot 0 is not covered ..... :)
values : 0.0730064
This Pitot 1 is covered !!
```

Figure 5.9: Results for pitot cover detection with Hu moment score

5.6 Result

Figure 5.9 shows our pitot cover detection result. The accuracy for our detection is 100% on our dataset of 42 images.

The main challenges we experience are due to shape. The shape of pitot cover can vary, they can be rectangular as our image shows or they cab have ribbon shape. Our system implementation is limited to detecting rectangular shape.

5.7 Conclusion

Our implementation is fast enough for real-time application. We apply our system on a robot to detect pitot covers from specific ground-check point. It gives accurate detection for pitot cover, but, limited for only rectangular shaped covers. Although our dataset images do not have any example of pitot cover of different shapes, we would work on making our system robust to different shapes.

Chapter 6

Conclusion

In this thesis, different image processing approaches are presented in regard to solving specific aircraft exterior inspection problems. Demands for real-time industrial usability are taken into consideration while developing the systems. To verify the suitability, each dataset is created with possible challenges and variants. Furthermore, contrast and brightness are simulated to add more complications. Reckoning the characteristics of each inspection and detection item, appropriate features are extracted to reduce computational expenses. Shape, color, edge, ridge, contour moment, and periodicity of intensity signals are the principle features that are studied for the implementation.

The work for solving tire detection and inspection problem is presented in Chapter 2. The approach focuses on a new way of inspecting airplane tires with 0% FNR. Chapter 4is motivated by the shape and periodicity of intensity features to inspect engine turbine blades. The most intuitive feature, color is used in Chapter 5 to check whether a pitot tube is covered or open. On the given dataset, this presented approach (Chapter 5) gives exact detection rate. Simplicity of Hough line transform works well in finding wheel chock which is described in Chapter 3 with 5% FPR.

Projection of aircraft check-items is done from simplified 3D model of Airbus A-320 to facilitate the detection and inspection tasks. The projected models work as heuristic parameters to approximate the size, shape, and location of items.

The presented methods can be found suitable for detecting and inspecting other objects where the features, we use, are also principle. They can also be applied in wide range of industrial applications, as well.

6.0.1 Future Directions

Our dataset is limited to one plane in a hangar and thus, do not offer much variability. We are working on creating a large dataset to validate our approaches. For tire detection, we have scope of improving accuracy in case of the challenges presented in Section 2.8, back-light detection problem. To overcome this, we would like to use machine learning. Later, we want to address the most important challenge toward having better tire inspection result, that is, to obtain sharp segmentation result. We want to introduce geometry processing for shape analysis to improve segmentation result. Computational requirements with tire segmentation is also demanding where we want to work further.

In engine inspection, we get number of false-alarms (Section 4.7). We need to explore feasible solutions using other image processing techniques for this specific problem. We would also want to refine our approach for detecting the presence of wheel chock and make it robust to noises illustrated in Section 3.6. Our pitot cover detection approach is limited to a specific shaped pitot cover (Section 5.6). In future, we need to increase our dataset with different type pitot covers and generalize our approach for detecting any shaped pitot cover.

Appendix A

Background

A.1 Hough Transform

Use of Hough transform (HT) in detection and inspection is straight-forward and popular [52, 59, 79, 80]. We use Hough circle transform for engine cowling detection (Section 4.5) and Hough line transform for turbine blade center detection (Section 4.6), blade inspection (Section 4.6), and wheel chock detection (Section 3.5).

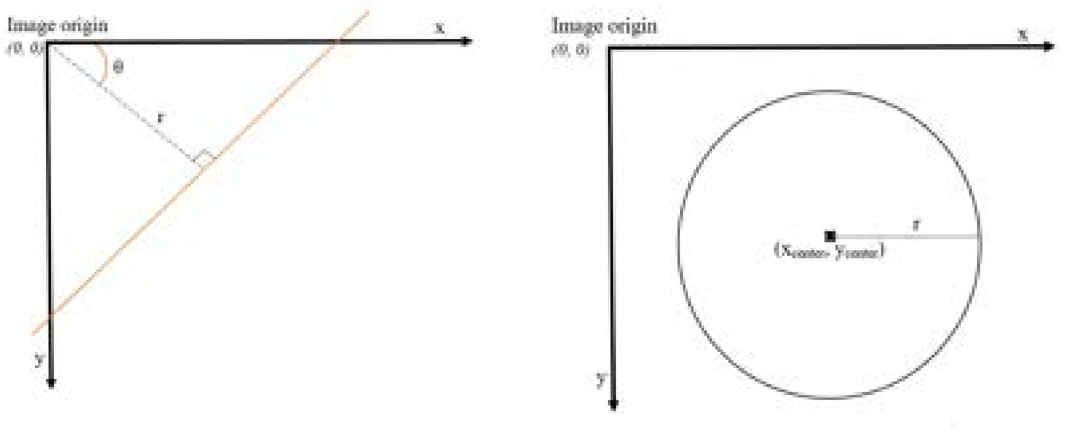
Hough transform [63] identifies geometrical features (ex. lines or circles) by accumulating a specific amount of votes from points in the image which contribute to each feature. To count the votes, hough transform generates an accumulator whose size depends on the number of parameters. The feature equation determines the number of parameters. The feature equations in Cartesian coordinate system are shown below.

$$\textit{Equation to detect line feature, } r = x \cdot cos\theta + y \cdot sin\theta$$

$$\textit{Equation to detect circle feature, } (x - x_{center})^2 + (y - y_{center})^2 = r^2$$

Line feature has two parameters , (r, θ) where r defines the distance of a line from the center and θ denotes the angle between the line normal and the x-axis. Figure A.1a shows how the hough line parameters are computed for images. Similarly, circle equation has three parameters, $(x_{center}, y_{center}, r)$, where (x_{center}, y_{center}) is the center of a circle and r is the radius of the circle. Figure A.1b shows the circle parameters.

Hence, the accumulator, for line detection, becomes two-dimensional and for circle detection, becomes three-dimensional in size. Number of votes for a line feature to be considered is the number of points on the line. Kiryati et al. [64] introduced probabilistic hough line to reduce computational cost of hough line transform where the main idea was to take small subset, n



(a) Line parameters

(b) Circle parameters

Figure A.1: Feature parameters for Hough transform

points into consideration for voting rather than taking all the points.

A.2 Contour Matching

Jamie Shotton et al. [81] used local contour to create dictionaries and eventually, to detect objects. In our work, we use contour processing in all the detection and inspection tasks.

Contour detection [82] is used to extract the boundaries of object in a binary image. It returns a set of points for the detected object. These points are used for further contour processing, as for computing contour area, to fill holes inside the contour area, or to match contours by their shapes.

Moment is a mathematic measure for the shape of points. Moments are computed using Green's theorem [83, 84], which in words is, "Area and centroid of an object in a plane figure can be determined by integrating over the perimeter". For image, we have discrete values and thus, moment of a contour of an object is following.

$$m_{j,i} = \sum_{x,y} f(x,y) \cdot x^j \cdot y^i \tag{A.1}$$

where *f(x,y)* is a binary image containing white contour on black background. The area of a contour is the zeroth moment, $m_{0,0}$ of a contour. The mass center of a contour is computed in the following way:

$$x_{center} = \frac{m_{1,0}}{m_{0,0}} = \frac{\sum_{x,y} f(x,y)\cdot x^1 \cdot y^0}{\sum_{x,y} f(x,y)\cdot x^0 \cdot y^0} = \frac{\sum_{x,y} f(x,y)\cdot x}{\sum_{x,y} f(x,y)}$$
$$y_{center} = \frac{m_{0,1}}{m_{0,0}} = \frac{\sum_{x,y} f(x,y)\cdot x^0 \cdot y^1}{\sum_{x,y} f(x,y)\cdot x^0 \cdot y^0} = \frac{\sum_{x,y} f(x,y)\cdot y}{\sum_{x,y} f(x,y)}$$

To match two contour shapes, seven Hu invariants [78] of both contours are compared. Hu et al. [78] proposed seven moment invariants that are invariant to contour scale, rotation, and reflection except the seventh one. In OpenCV, there are three existing comparison measurements. We use the following measurement for our work.

$$I(A,B) = \sum_{i=1}^{7} \left| \frac{1}{m_i^A} - \frac{1}{m_i^B} \right|$$
$$\textit{here, A and B are two object contours and } m_i^A = sign(h_i^A) \cdot logh_i^A \;\&$$
$$m_i^B = sign(h_i^B) \cdot logh_i^B,\ h_i^A, h_i^B \textit{ are Hu moments of A and B, respectively}$$

Flood-fill fills a closed contour by starting from a seed point and fill four or eight connected area around the seed point.

A.3 Sensitivity and Specificity

Inspection approaches are evaluated by calculating the false positive rate (FPR) and false negative rate (FNR). The input image in terms of the tested approach is classified as FP, in the case when the defect is not present and the approach reported a defect. It is classified as false negative (FN), in the case when there is a defect and the method reports no defect. FPR is the ratio between number of FP images and the total number of images with no defect. Following is the measure of false alarms,

$$FPR = \frac{FP}{FP+TN} \tag{A.2}$$

FNR is the ratio between the number of FN images and the total number of images with defects. This is the measure of missed defects and is considered critical in our application.

$$FNR = \frac{FN}{FN+TP} \tag{A.3}$$

Here, TN is the number of images with no defect on which the method reported no defect, and TP is the number of images with defect on which the method reported a defect. FNR increases when some defects are missed.

Bibliography

[1] Eric Bradley and Marc Romero. The a320 walk-around. http://www.equicom.net/mcdu/blog20150714.php, July 2015.

[2] SharkD. The hsv color model mapped to a cylinder. pov-ray source is available from the pov-ray object collection. https://upload.wikimedia.org/wikipedia/commons/0/0d/HSV_color_solid_cylinder_alpha_lowgamma.png, March 2010.

[3] Igor Jovančević, Larnier Stanislas, Jean-José Orteu, and Thierry Sentenac. Automated exterior inspection of an aircraft with a pan-tilt-zoom camera mounted on a mobile robot. *Journal of Electronic Imaging*, 24(6):06111–0–06111–15, Nov/Dec 2015.

[4] M. Siegel, P. Gunatilake, and G. Podnar. Robotic assistants for aircraft inspectors. *IEEE Instrumentation Measurement Magazine*, 1(1):16–30, Mar 1998.

[5] Bristol universities team up with easyjet for robot plane project. http://www.bristolpost.co.uk/Easyjet-start-using-technology-s-easy-understand/story-21071915-detail/story.html, May 2014.

[6] Domingo Mery and Miguel Carrasco. *Advances in Image and Video Technology: First Pacific Rim Symposium, PSIVT 2006, Hsinchu, Taiwan, December 10-13, 2006. Proceedings*, chapter Advances on Automated Multiple View Inspection, pages 513–522. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[7] José A. Calderón-Martinez and Pascual Campoy-Cervera. *Artificial Neural Nets Problem Solving Methods: 7th International Work-Conference on Artificial and Natural Neural Networks, IWANN2003 Maó, Menorca, Spain, June 3–6, 2003 Proceedings, Part II*, chapter Defects Detection in Continuous Manufacturing by means of Convolutional Neural Networks, pages 528–535. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[8] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Computer Vision, 1998. Sixth International Conference on*, pages 839–846, Jan 1998.

[9] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. Grabcut -interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (SIGGRAPH)*, 23:309–314, August 2004.

[10] C.W. H. Tire inspection apparatus, November 10 1942. US Patent 2,301,251.

[11] H. Mizukusa, H. Nakamoto, and W. Otani. Illumination device and inspection device of tire, May 31 2012. US Patent App. 13/156,680.

[12] E. Takahashi and N. Sakoda. Tire shape inspection method and tire shape inspection device, March 24 2011. US Patent App. 12/737,009.

[13] T. Fujii and A. Mizutani. Tire contour measurement data correction method and tire visual inspection device, August 18 2015. US Patent 9,113,046.

[14] R. Zoughi and M.T. Ghasr. Tire inspection using microwave imaging methodologies, February 5 2015. US Patent App. 14/444,851.

[15] F. Teti, S. Urban, and C. Austin. Computer vision aided automated tire inspection system for in-motion inspection of vehicle tires, September 24 2013. US Patent 8,542,881.

[16] Yair Wiseman. Camera that takes pictures of aircraft and ground vehicle tires can save lives. *Journal of Electronic Imaging*, 22(4):041104–041104, 2013.

[17] Andr P. Dias, Manuel F Silva, Nuno Lima, and Ricardo Guedes. Identification of marks on tires using artificial vision for quality control. *International Journal for Quality Research*, 9(1):27–36, 2015.

[18] Y. Xiang, C. Zhang, and Q. Guo. A dictionary-based method for tire defect detection. In *Information and Automation (ICIA), 2014 IEEE International Conference on*, pages 519–523, July 2014.

[19] Qiang Guo and Zhenwen Wei. Tire defect detection using image component decomposition). *Research Journal of Applied Sciences, Engineering and Technology*, 4(01):41–44, 2012.

[20] T. Wahdan, G. Abandah, A. Seyam, A. Awwad, and I. Shdaifat. Tire type recognition through treads pattern recognition and dot code ocr. *Ubiquitous Computing and Communication Journal (UBICC)*, 9(1):1481–1487, Feb 2015.

[21] Penny Chen, Gary D. Shubinsky, Kwan-Hwa Jan, Chien-An Chen, Oliver Sidla, and Wolfgang Poelzleitner. Inspection of tire tread defects using image processing and pattern recognition techniques. *Proc. SPIE*, 2063:14–21, 1993.

[22] O. Achler and M. M. Trivedi. Vehicle wheel detector using 2d filter banks. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 25–30, June 2004.

[23] S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1475–1490, Nov 2004.

[24] K. K. Kim, K. I. Kim, J. B. Kim, and H. J. Kim. Learning-based approach for license plate recognition. In *Neural Networks for Signal Processing X, 2000. Proceedings of the 2000 IEEE Signal Processing Society Workshop*, volume 2, pages 614–623 vol.2, 2000.

[25] R. M. Dufour, E. L. Miller, and N. P. Galatsanos. Template matching based object recognition with unknown geometric parameters. *IEEE Transactions on Image Processing*, 11(12):1385–1396, Dec 2002.

[26] Luke Cole, David Austin, and Lance Cole. Visual object recognition using template matching. In *Proceedings of Australian Conference on Robotics and Automation*, 2004.

[27] Jianming Zhang and Stan Sclaroff. saliency detection: a Boolean map approach. In *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, 2013.

[28] Jianming Zhang and Stan Sclaroff. Exploiting surroundedness for saliency detection: a Boolean map approach. *IEEE Trans. Pattern Anlaysis and Machine Intellegence (TPAMI)*, 2015.

[29] Sebastian Montabone and Alvaro Soto. Human detection using a mobile platform and novel features derived from a visual saliency mechanism. *Image Vision Comput.*, 28(3):391–402, March 2010.

[30] Erkut Erdem and Aykut Erdem. Visual saliency estimation by nonlinearly integrating features using region covariances. *Journal of Vision*, 13(4):11, 2013.

[31] Kuo-Yu Chiu and Sheng-Fuu Lin. Lane detection using color-based segmentation. In *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.*, pages 706–711, June 2005.

[32] Yinghua He, Hong Wang, and Bo Zhang. Color-based road detection in urban traffic scenes. *IEEE Transactions on Intelligent Transportation Systems*, 5(4):309–318, Dec 2004.

[33] M. N. Wu, C. C. Lin, and C. C. Chang. Brain tumor detection using color-based k-means clustering segmentation. In *Intelligent Information Hiding and Multimedia Signal Processing, 2007. IIHMSP 2007. Third International Conference on*, volume 2, pages 245–250, Nov 2007.

[34] Reyer Zwiggelaar and Erika R. E. Denton. *Digital Mammography: 8th International Workshop, IWDM 2006, Manchester, UK, June 18-21, 2006. Proceedings*, chapter Texture Based Segmentation, pages 433–440. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[35] Jinyou Zhang and H. H. Nagel. Texture-based segmentation of road images. In *Intelligent Vehicles '94 Symposium, Proceedings of the*, pages 260–265, Oct 1994.

[36] Arko Lucieer, Alfred Stein, and Peter Fisher. Multivariate texturebased segmentation of remotely sensed imagery for extraction of objects and their uncertainty. *International Journal of Remote Sensing*, 26(14):2917–2936, 2005.

[37] G. Loy and N. Barnes. Fast shape-based road sign detection for a driver assistance system. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 1, pages 70–75 vol.1, Sept 2004.

[38] H. E. Abd El Munim and A. A. Farag. A shape-based segmentation approach: an improved technique using level sets. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 930–935 Vol. 2, Oct 2005.

[39] S. Beucher and F. Meyer. The morphological approach to segmentation: the watershed transformation. Mathematical morphology in image processing. *Optical Engineering*, 34:433–481, 1993.

[40] Y. Y. Boykov and M. P. Jolly. Interactive graph cuts for optimal boundary amp; region segmentation of objects in n-d images. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 105–112 vol.1, 2001.

[41] S. Han, W. Tao, D. Wang, X. C. Tai, and X. Wu. Image segmentation based on grabcut framework integrating multiscale nonlinear structure tensor. *IEEE Transactions on Image Processing*, 18(10):2289–2302, Oct 2009.

[42] W. Yang, J. Cai, J. Zheng, and J. Luo. User-friendly interactive image segmentation through unified combinatorial user inputs. *IEEE Transactions on Image Processing*, 19(9):2470–2479, Sept 2010.

[43] L. Grady. Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1768–1783, Nov 2006.

[44] R. Adams and L. Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):641–647, Jun 1994.

[45] Serge Beucher. The watershed transformation. `http://cmm.ensmp.fr/~beucher/wtshed.html`, May 2010.

[46] Jos B.T.M. Roerdink and Arnold Meijster. The watershed transform: Definitions, algorithms and parallelization strategies. *Fundam. Inf.*, 41(1,2):187–228, April 2000.

[47] P. A. V. de Miranda, A. X. Falcão, and J. K. Udupa. Synergistic arc-weight estimation for interactive image segmentation using graphs. *Comput. Vis. Image Underst.*, 114(1):85–99, January 2010.

[48] Nicky van Foreest. Fitting an ellipse to a set of data points. `http://nicky.vanforeest.com/misc/fitEllipse/fitEllipse.html`, Jan 2016.

[49] Alejandro F. Frangi, Wiro J. Niessen, Koen L. Vincken, and Max A. Viergever. *Medical Image Computing and Computer-Assisted Intervention — MICCAI'98: First International Conference Cambridge, MA, USA, October 11–13, 1998 Proceedings*, chapter Multiscale

vessel enhancement filtering, pages 130–137. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[50] NOBUYUKI OTSU. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, Jan 1979.

[51] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, 2001.

[52] T. T. Nguyen, Xuan Dai Pham, and J. W. Jeon. Rectangular object tracking based on standard hough transform. In *Robotics and Biomimetics, 2008. ROBIO 2008. IEEE International Conference on*, pages 2098–2103, Feb 2009.

[53] Y. Liu, T. Ikenaga, and S. Goto. A novel approach of rectangular shape object detection in color images based on an mrf model. In *2006 5th IEEE International Conference on Cognitive Informatics*, volume 1, pages 386–393, July 2006.

[54] Per-Erik Danielsson and Olle Seger. Generalized and separable sobel operators. In Herbert Freeman, editor, *Machine Vision for Three-Dimensional Scenes*, pages 347 – 379. Academic Press, 1990.

[55] I. Sobel and G. Feldman. A 3x3 isotropic gradient operator for image processing, 1968. Presented at Stanford Artificial Intelligence Project (SAIL).

[56] J. Matas, C. Galambos, and J. Kittler. Robust detection of lines using the progressive probabilistic hough transform. *Comput. Vis. Image Underst.*, 78(1):119–137, April 2000.

[57] R.C. Thomson. Inspection device for aircraft engine, March 2 1993. US Patent 5,191,412.

[58] Dirk Padfield, Glen Brooksby, and Robert Kaucic. Automatic deformation detection for aircraft engine disk inspection. *Journal of Multimedia*, 2(4), 2007.

[59] G. Li, Y. Genc, E. Baleine, and D.H. Lemieux. Hough transform approach to gap measurement in blade inspection, April 11 2013. US Patent App. 13/617,416.

[60] General Electric Company. Turbine blade inspection. http://www.nikonmetrology.com/en_US/Applications/Aerospace/Turbine-Blade-Inspection.

[61] Martin Stokkeland. A computer vision approach for autonomous wind turbine inspection using a multicopter, 2014.

[62] Huiyi Zhang and John K. Jackman. Feasibility of automatic detection of surface cracks in wind turbine blades. *Wind Engineering*, 38:575–586, 2014.

[63] H.P.V. C. Method and means for recognizing complex patterns, December 18 1962. US Patent 3,069,654.

[64] N. Kiryati, Y. Eldar, and A. M. Bruckstein. A probabilistic hough transform. *Pattern Recogn.*, 24(4):303–316, February 1991.

[65] Peter H. Westfall. Kurtosis as peakedness, 19052014. r.i.p. *The American Statistician*, 68(3):191–195, 2014.

[66] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, Nov 1986.

[67] Alan V. Oppenheim, Ronald W. Schafer, and John R. Buck. *Discrete-time Signal Processing (2Nd Ed.)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1999.

[68] D. G. Lowe. Object recognition from local scale-invariant features. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1150–1157 vol.2, 1999.

[69] D.G. Lowe. Method and apparatus for identifying scale invariant features in an image and use of same for locating an object in an image, March 23 2004. US Patent 6,711,293.

[70] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[71] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[72] Theo Gevers and Arnold W.M. Smeulders. Color-based object recognition. *Pattern Recognition*, 32(3):453 – 464, 1999.

[73] FengWanli and GaoShangbing. A vehicle license plate recognition algorithm in night based on hsv. In *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE)*, volume 4, pages V4–53–V4–56, Aug 2010.

[74] H.D. Cheng, X.H. Jiang, Y. Sun, and Jingli Wang. Color image segmentation: advances and prospects. *Pattern Recognition*, 34(12):2259 – 2281, 2001.

[75] DK Banerjee, SK Parui, and D. Dutta Majumder. *Knowledge Based Computer Systems: International Conference KBCS '89 Bombay, India, December 11–13, 1989 Proceedings*, chapter Shape based object recognition, pages 363–374. Springer Berlin Heidelberg, Berlin, Heidelberg, 1990.

[76] Anil K. Jain and Aditya Vailaya. Image retrieval using color and shape. *Pattern Recognition*, 29(8):1233 – 1244, 1996.

[77] Changing colorspaces. http://docs.opencv.org/3.1.0/df/d9d/tutorial_py_colorspaces.html#gsc.tab=0.

[78] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, 8(2):179–187, February 1962.

[79] M. Godec, P. M. Roth, and H. Bischof. Hough-based tracking of non-rigid objects. In *2011 International Conference on Computer Vision*, pages 81–88, Nov 2011.

[80] O. Barinova, V. Lempitsky, and P. Kohli. On detection of multiple object instances using hough transforms. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2233–2240, June 2010.

[81] J. Shotton, A. Blake, and R. Cipolla. Contour-based learning for object detection. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 1, pages 503–510 Vol. 1, Oct 2005.

[82] Satoshi Suzuki and KeiichiA be. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32 – 46, 1985.

[83] Wilfred Kaplan. *Advanced Calculus (4th Edition)*. Addison-Wesley, Reading, Massachusetts, 1991.

[84] Eric W. Weisstein. "green's theorem." from mathworld–a wolfram web resource. http://mathworld.wolfram.com/GreensTheorem.html.