Augmentation of Camera based Non-Destructive Testing using Robotics and Motion Tracking

A thesis submitted

in partial fulfillment for the award of the degree of

Doctor of Philosophy

by

Saurabh Chatterjee

Department of Aerospace Engineering

Indian Institute of Space Science and Technology

Thiruvananthapuram, India

March 2024

Certificate

This is to certify that the thesis titled ***Augmentation of Camera based Non-Destructive Testing using Robotics and Motion Tracking*** submitted by **Saurabh Chatterjee**, to the Indian Institute of Space Science and Technology, Thiruvananthapuram, in partial fulfillment for the award of the degree of **Doctor of Philosophy** is a bona fide record of the original work carried out by him/her under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. K Kurien Issac
Senior Professor,
Department of Aerospace Engineering,
IIST Trivandrum.

Dr. Deepu M.
Professor & head,
Department of Aerospace Engineering,
IIST Trivandrum.

Place: Thiruvananthapuram
Date: March 2024

Declaration

I declare that this thesis titled ***Augmentation of Camera based Non-Destructive Testing using Robotics and Motion Tracking*** submitted in partial fulfillment for the award of the degree of **Doctor of Philosophy** is a record of the original work carried out by me under the supervision of **Dr. K Kurien Issac**, and has not formed the basis for the award of any degree, diploma, associateship, fellowship, or other titles in this or any other Institution or University of higher learning. In keeping with the ethical practice in reporting scientific information, due acknowledgments have been made wherever the findings of others have been cited.

Place: Thiruvananthapuram
Date: March 2024

Saurabh Chatterjee
(SC14D016)

This thesis is dedicated to my parents Suranjan Chatterjee and Amrita Chatterjee for supporting me throughout the long period of the research and development needed for this thesis . . .

Acknowledgements

I would like to express my gratitude to everyone who has contributed to the successful completion of my Phd thesis.

First and foremost, I would like to thank my guide, Dr K. Kurien Issac, Senior Professor at the Indian Institute of Space Science and Technology (IIST), for his invaluable support, guidance, and encouragement throughout the course of this project. His expertise, insights, and constant feedback have been instrumental in shaping my research. I am especially grateful to the fact that he had supported my decision to change the objective of this work to one which I had a greater personal interest in and which would be more relevant to the technological progress of the country.

I would also like to thank Mr. Harikrishna S. from Vikram Sarabhai Space Centre (VSSC), Composites Entity. His valuable insights into the importance of Robotic and Automated Systems for Non-Destructive Testing in Aerospace applications were the main inspiration and driving force behind this investigation. His help in procuring sample panels from VSSC for demonstration of path planning algorithms is also greatly appreciated.

I am also thankful to my co-founders and colleagues at Vashishtha Research Private Limited, especially Mr. Vignesh K.C. and Mr. Jeevan Philip N., who were instrumental to the development of hardware needed to demonstrate the Robotic Path Rlanning process. I would also like to acknowledge the Department of Aerospace Engineering faculty members at IIST, who have supported me throughout my academic journey. Their unwavering support and encouragement have helped me grow both academically and personally.

Finally, I would like to express my heartfelt gratitude to my family and friends for their unwavering support, love, and encouragement that have been the driving force behind this project's success.

Saurabh Chatterjee

Abstract

The usage of Camera based Non-Destructive Testing (NDT) methods such as Thermography and Laser Shearography in Aerospace applications such as the inspection of large composite panels for defects presents some challenges. Even though they are area based methods, they still capture only a small fraction of the surface to be inspected at a time. Moreover, the surfaces which they are supposed to inspect can be curved or complex in shape, while the image that is captured by the camera is rectangular in shape according to how the camera frustum intersects the surface. Hence it can be challenging to achieve a complete coverage of the surface from multiple viewpoints and also relate the data from images gathered to the actual location of the defects in the panel.

Computer Aided Inspection (CAI) involves the use of hardwares like robotic and motion tracking systems, and associated algorithms for path planning, motion tracking and image stitching. While the CAI solutions for Ultrasonic NDT and Radiography (X-ray) have been available for a long time in the form of C-scan machines and Computed Tomography systems respectively, the same cannot be said for lesser known NDT methods such as Thermography and Laser Shearography. While some solutions are known to exist at research centres such as NASA, the know-how related to this is not shared in the public domain. Augmentation of the camera based NDT processes is therefore much needed as these processes are gaining more acceptance due to their non-contact, non-hazardous and area based nature of inspection.

This thesis presents algorithms and techniques to augment the camera based NDT Systems using Motion Tracking, Viewpoint Planning, Image Stitching and Coverage Path Planning. The working of these algorithms is demonstrated using theoretical simulations on a 'Digital Model' of the system and also on prototype robotic and motion tracking hardware. The development of these prototype hardware and their operation, as well as their limitations is also explored.

The first chapter gives an introduction to the problem being solved and the motivation behind the present approach. Several case studies of existing solutions in the field of NDT Augmentation using Robotics, Drones and Motion tracking systems are presented. The NDT systems being augmented are using Camera based techniques like Thermography and Laser shearography, though in some cases ultrasonic NDT is also given as examples.

These case studies are primarily deductions from studying videos and conference papers. The exact methodology as to how the solution is reached is not disclosed.

The second chapter gives the details about the construction and operation of the prototype 5-axis cartesian robot used for the inspection of a panel. The accuracy and repeatability of the robot are analyzed using metrological methods and its limitations are documented.

The third chapter details the development of the prototype motion tracking systems and characterizes the performance of the optical hardware to give some insights about the limitations of the different cameras and optical systems. The algorithms for motion tracking using April Tags and integrating it with the 3D model of the panel being scanned are detailed. The stereo and monocular tracking pipelines, through which the image data captured by the camera are converted into pose information for the motion tracking integration are fully described.

Viewpoint planning and Image Stitching using the virtual 3D model of the panel is the subject of the fourth chapter. A user friendly 3D viewer is developed to accept input from the user and manually plan the viewpoints on the 3D model of the panel. The G-code for positioning the robot is generated from the viewpoints chosen by the user and the robot is moved using this code to record the corresponding images. The image obtained are stitched onto the 3D model of the panel by using a 3D stitching algorithm as presented, which is based on mapping points in the pointcloud to the images taken. Blending is done at the overlapping areas covered by more than one viewpoint, using the Grassfire algorithm. The 3D image thus formed is presented to the user. For verifying the technical correctness of the method, simulated data taken from the virtual camera model is also used in the image stitching process. Evaluation of the stitching result is done by comparing it with data from the actual panel.

The Fifth chapter improves on the manual viewpoint planning approach presented in the previous chapter by addressing the problem of coverage path planning. While in the literature much study has been done on optimization based methods and geometric methods utilizing the slicing approach, the method present here relies on geodesic path generation and mapping a grid of viewpoints onto the surface. Nodes corresponding to viewpoints are generated automatically and the node generation is stopped based on certain criteria. This approach is semi automated and relies on the user to give good starting conditions for the node propagation. A large number of examples are presented to show the strengths and weaknesses of the method.

The Last chapter details the conclusions from the study. The algorithms for the aug-

mentation of camera based NDT Systems, including the development of virtual 3D models, motion tracking system integration with NDT, path planning including automated coverage path planning, 3D image stitching and visualization are fully detailed and documented. Prototype hardware for cartesian robot and motion tracking have been developed for the demonstration of these algorithms and also their limitations are noted. Simulation models for verifying coverage path planning and image stitching techniques have also been extensively explored. It is hoped that the presented work will lead to a unified common model for the augmentation of camera based NDT systems to be used in industrial applications.

Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations

CAI	Computer Aided Inspection
CNC	Computer Numerical Control
CPP	Coverage Path Planning
DOF	Degrees of Freedom
ISRO	Indian Space Research Organization
NASA	National Aeronautical and Space Agency
NDT	Non Destructive Testing
GPS	Global Positioning System
IR	Infrared
IP	Internet Protocol
LED	Light Emitting Diode
MPG	Manual Pulse Generator
MTF	Modulation Transfer Function
PTZ	Pan Tilt Zoom
QA	Quality Assurance
RGB	Red Green Blue
STL	Standard Tessellation Language
TSP	Traveling Salesman Problem
TVPP	Traveling View Planning Problem
UAV	Unmanned Aerial Vehicle
USB	Universal Serial Bus
VPP	View Planning Problem
VSSC	Vikram Sarabhai Space Centre

Nomenclature

Standard mathematical conventions are used. Italic or plain text are used to represent scalars or components of vectors, while bold font is used to represent vectors, matrices and sets. Capital letters are employed for representation of matrices and sets of complex data. The nomenclature of the three main chapters (motion tracking, viewpoint planning and image stitching and semi automated coverage pathplanning) are separate, and are noted down separately in order to avoid a conflict in representation.

Motion Tracking

u, v	Image Coordinates
x, y, z	World or Object Coordinates
$\mathbf{R}$	Rotation Matrix
$\mathbf{t}$	Translation Vector
$\mathbf{K}$	Camera Calibration Matrix
$\mathbf{E}$	Essential Matrix
$\mathbf{F}$	Fundamental Matrix
B	Baseline Distance
$\mathbf{R}_{\mathbf{w}}^{\mathbf{b}}$	Rotation Matrix from world to body coordinates
$\mathbf{t}_{\mathbf{w}}^{\mathbf{b}}$	Translation vector of the body in world coordinates
$\mathbf{u}^{\mathbf{w}}$	Up vector in world coordinate system

Viewpoint Planning and Image Stitching

$\mathbf{n}, \mathbf{t}, \mathbf{b}$	Unit vectors for normal, tangent and binormal at a point
p_h, p_v	Size of the viewing frustum, projected on the surface (Horizontal and Vertical)
d_0	Distance of Camera from viewpoint
WIDTH	The width of the Camera image in pixels
HEIGHT	The height of the Camera image in pixels
θ_{VFOV}	The vertical field of view of the camera in radians
n_x, n_y, n_z	Components of the normal vector

θ, ϕ	Yaw and Pitch joint angles
j_0	Joint Offset
$\mathbf{w_c}$	Wrist Centre
$d(x, y)$	Distance to the edge of the image
p_{avg}	Weighted pixel value computed
n	number of points in the pointcloud
v	number of viewpoints

Semi Automated Coverage Path Planning

p_f	Projection Factor
$\mathbf{p_0}, \mathbf{p_1}, \mathbf{p_e}$	Position Vectors
$\mathbf{n}, \mathbf{n_0}, \mathbf{n_e}$	Normal Vectors
$\mathbf{t_0}, \mathbf{t_1}, \mathbf{t_e}$	Tangent Vectors
$\mathbf{b}$	Binormal Vector
θ_s	Search Angle
u, v	Barycentric coordinates
$\mathbf{A}, \mathbf{B}, \mathbf{C}$	Vertices of a Triangle
$\mathbf{N}$	Normal of a Triangle
i, j	Coordinates in the 2D Grid
p_h, p_v	Size of the viewing frustum, projected on the surface (Horizontal and Vertical)
d_0	Distance of Camera from viewpoint
WIDTH	The width of the Camera image in pixels
HEIGHT	The height of the Camera image in pixels
θ_{VFOV}	The vertical field of view of the camera in radians
$\mathbf{M}$	The set of triangles in a Mesh
$\mathbf{R}, \mathbf{L}, \mathbf{U}, \mathbf{D}$	The nodes to the Right, Left, Up, Down of the current node
n	number of viewpoints
M	number of triangles in a Mesh

Chapter 1

Introduction

Non Destructive Testing (NDT) is an integral part of the manufacturing process, in order to ensure that the manufactured parts are free from defects that might compromise their usability. Together with metrology (Dimensional Inspection), it forms a part of the inspection process for Quality Assurance (QA) that is needed to certify that a manufactured item is fit for usage. Furthermore, NDT is also required over the lifetime of a product, as structural components suffer from fatigue, impact damage and corrosion over time, which weakens their mechanical strength and may cause a failure.

The detection of defects in structures using methods of Non-Destructive Testing is gaining more importance for ensuring a total quality control over manufactured products and monitoring and maintenance of these assets over their intended lifetime. This is particularly important for Aerospace applications where a serious defect in even a single part can cause catastrophic failures and even the loss of life.

This thesis proposes certain approaches for augmenting image based NDT systems with robotics and motion tracking, and demonstrates their application using prototype systems. This chapter explores the motivation behind this study and the literature survey describing the current state of the work in this field.

1.1 Motivation behind the study

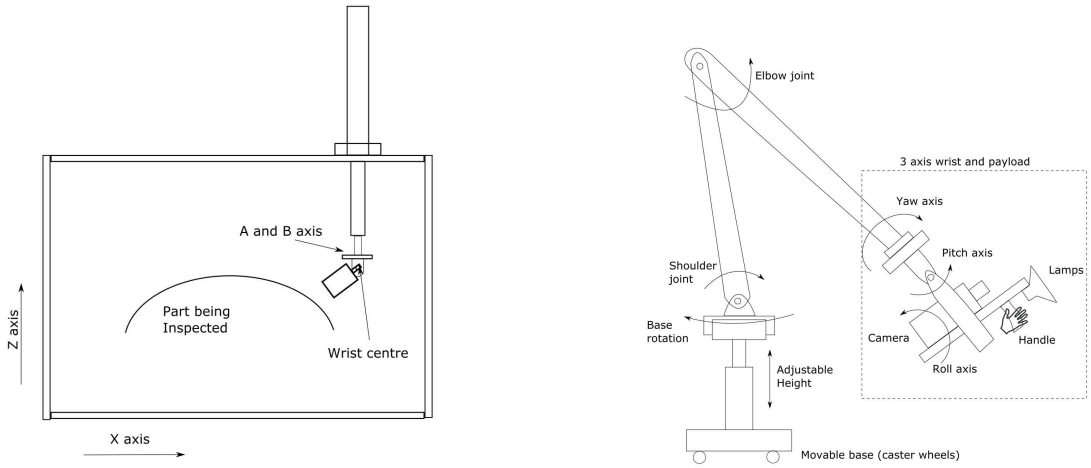
An equipment for Non-Destructive Testing, or inspection in general, can be used by itself as a stand alone device. For example, a bore dial gauge can be used to manually measure the diameter of a hole and the value can be noted down manually. If a large number of such readings need to be taken in order to measure the circularity of the hole, we should go for an electronic gauge that saves a large number of data points in a computerized fashion. In this case, the data recording is done electronically, though the inspection device is still

being moved by hand. If a part has thousands of holes, or thousands of parts with the same holes need to be inspected, we need a robotic or automated solution in order to save time and manual errors. In order to move the robot to the appropriate position in space to inspect the hole, we would need a robotic path planning process or algorithm. Finally, when all the data has been gathered, we may need to combine the data to present a useful report for the user in an automated manner. We can see now in this example, how a simple operation of measuring a hole diameter of a part has become very sophisticated. However the improvement in productivity over the basic manual method would be huge. This approach is known as Computer Aided Inspection or the augmentation of Non-Destructive Testing methods. With the progress of technology, the digitization of recording and displaying data of all kinds through automated means is becoming more and more important. Thus, the approach of automating the positioning of the recording device, the recording of the data, combining and displaying the data will become more and more prevalent in the industry in future.

In this particular study, we are interested in the inspection of panel surfaces that make up the aero-structures of aircrafts and space vehicles. Many methods such as liquid penetrant, eddy current, ultrasonic, radiography, etc. can be used for Non-Destructive Testing of such panels. Optical methods based on camera systems such as Thermography [12] and Laser Shearography [13] have been noted for their area based, non-contact and non-hazardous nature. Optical methods are usually used for detecting defects close to the surface of the structure. Non-contact techniques are very useful in cases where the surface is fragile and can be damaged by contact probes rubbing on it (such as space composites and historical artworks) or in cases where the surface is hot or inaccessible in some other way.

In such methods, the defect data is gathered by analyzing image data that is taken by a camera and this data is gathered from the panel according to how the camera's viewing frustum is positioned and oriented with reference to the panel. However, the images or data gathered by the camera does not always easily give an indication of where this data has been recorded on a panel. Also, although the camera covers an area of the panel, it is still only a small part of the surface and hence an automated system that can scan over the panel from multiple viewpoints is desired. Hence, methods of NDT augmentation are a useful addition over the base NDT system that simply captures the images and analyzes them for defects.

Various methods therefore have been proposed for augmentation of a data gathering system for an outdoor or on-site inspection. Figure 1.1 provides a conceptual diagram for four different methods that could be used for the NDT augmentation of a camera based



(a) Robotic System for NDT

(b) Passive Arm for NDT

(c) Drone Inspection of Aircraft

(d) Motion tracking system

Figure 1.1: NDT augmentation methods

NDT system.

A robotic arm or manipulator system is a common method of inspection of large aircraft structures or panels. Figure 1.1a shows a robotic system which is a cartesian robotic cell with 5 degrees of freedom scanning over a panel. The robotic system can be programmed to follow a certain path to go to the relevant viewpoints in order to inspect the surface.

A mobile passive arm with encoders fitted at the joints is shown in Figure 1.1b, where springs and counterweights are used to offset the weight of the payload while the technician guides it by hand from point to point. The encoders will give the position and orientation of the end effector (NDT system shown in 1.1b is a cooled IR camera with heating lamps for active Thermography) by using the equations of inverse kinematics. It is mounted on a movable base and encoders can be fitted to the wheels as well to estimate the motion on a planar floor as it may need to be moved around to access the entire aircraft or structure.

A drone is shown inspecting an aircraft in Figure 1.1c. It can use differential GPS and IMU for estimating the location and orientation of the drone. The drone can fly over and

swivel its payload to inspect different parts of the aircraft.

A motion tracking system is shown in Figure 1.1d. In this method, the NDT device will be moved by hand, but the position and orientation information can be recorded by means of a motion tracking system. The motion tracking system consists of cameras and motion tracking markers. It can also be implemented using laser trackers, though that method is very expensive.

Implementations of all the above methods in the industry and research labs are given in the next section. It is to be noted that the official literature or publications do not cover all the NDT Augmentation methods in much detail. However a great deal of information can be found online in the form of video sharing content on platforms like YouTube. This means that while such methods have been implemented at organizations like NASA, the 'know how' of how the augmentation system works in terms of algorithms is sorely lacking. The YouTube videos provide next to no information on how these processes are actually achieved and even their corresponding published literature provides very little information. This may be due to the proprietary knowledge of the subject at hand.

Hence, a large part of the motivation behind this thesis is to develop similar approaches and bring the 'know-how' of these algorithms for NDT augmentation into the public domain and make it available to the NDT community in general.

1.2 Literature Survey

In this section, we will cover the existing work in the field of NDT augmentation in the form of case studies. The 'Literature' being studied in this section is mostly based on a few conference publications and Youtube Videos as the journal publications in this specific area is very lacking. Publications related to the methodologies and algorithms of motion tracking, path planning, coverage and image stitching are covered in their respective chapter introduction sections. This section focuses mostly on the outcomes achieved by studying some related papers and the Youtube videos associated with them, and not so much on 'how' the outcomes were achieved. In most of the cases, the know-how is not given as public knowledge.

1.2.1 Automated Robotic Manipulators

In this section we explore various examples of Robots being used to scan a surface using Non-Destructive Testing instruments.

1.2.1.1 Case Study 1: NASA Collaborative Robots for Thermographic Line-Scanning

Figure 1.2: Dual Robot for Thermographic inspection - 3D scanning of structure [1]

Cramer et al [1] describes a Dual Robot based system for Non-Destructive Inspection of an Aircraft Fuselage using Thermography, implemented at NASA. Their system is shown in [14]. This case study is very important because it demonstrates all the use-cases we will explore later in the thesis. Zalameda et al [15] and Cramer [2] also describe this use case as well as other applications of Thermographic NDT in NASA for the inspection of aero structures.

In the first operation (Figure 1.2), a 3D scanner (Creaform Metrascan) is used to generate the 3D model of the panel (fuselage structure). This panel is generated as a set of points (pointcloud) captured by the 3D scanner.

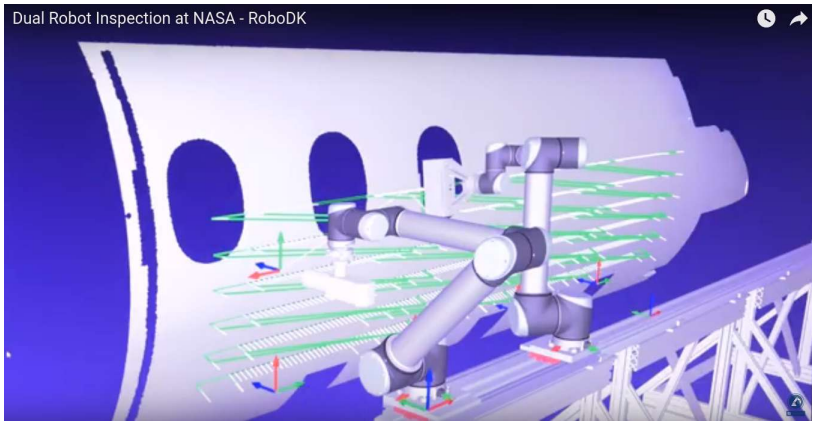


Figure 1.3: Dual Robot for Thermographic inspection - Path Planning [1]

After the 3D model of the panel has been generated by scanning, the path planning for generating the path of the NDT system over the structure is done. A virtual model

environment, presumably in RoboDK is used for this purpose (Figure 1.3). It is to be noted that in this example, the heating system and the infrared camera for Thermographic inspection are carried on two independent robots.

After the path is planned, the robots are instructed to move along the path as planned so as to cover the structure (Figure 1.4). The method being used here is Line-scan Thermography (see [1]), unlike conventional Thermography which stops at viewpoints to capture images.

Figure 1.4: Dual Robot for Thermographic inspection - Line Scan [1]

Finally, after completion of the thermographic line-scanning process, the NDT information (Thermography data) can be superimposed on the 3D model of the structure being inspected (Figure 1.5). The area for which the scanning was done is shown with the NDT information, while for the remaining area, the default model of the aeroplane is shown.

Figure 1.5: Dual Robot for Thermographic inspection - Image stitching [1]

1.2.1.2 Case Study 2: NASA Robot for Pulse Thermography inspection

Cramer [2] describes a Robotic system for inspection using Pulse thermography. This is a conventional process of Thermographic inspection, where the robot is moved to specific view-points and held there for the data gathering process (heating the panel using flash lamps and observing the thermal cool-down). This is illustrated well in the YouTube video [6].

Figure 1.6: Robot for Thermographic inspection - NASA [2]

Figure 1.6 shows a Pulse Thermography system mounted on a Robotic Arm by Universal Robots, which is used to scan the aircraft fuselage structure for defects.

Figure 1.7 shows the path planning process used for this application. It appears not to use any form of automated path planning. The viewpoints are all being planned manually.

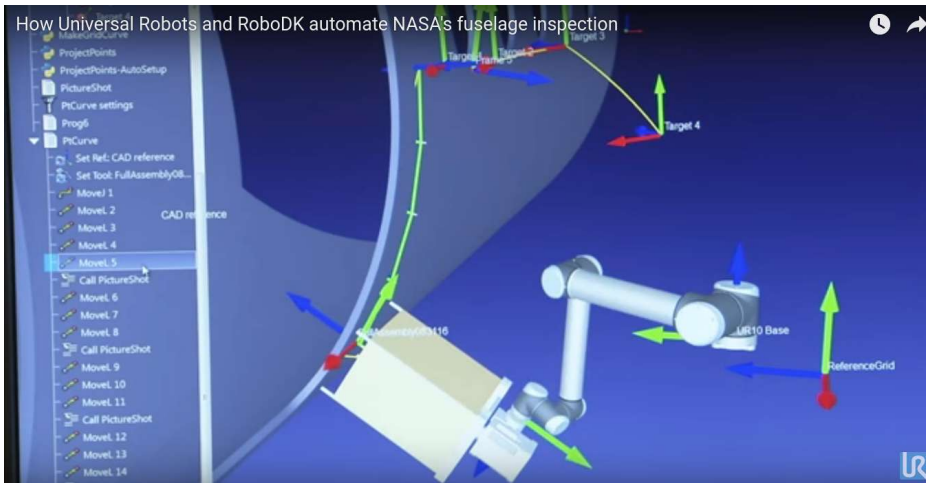


Figure 1.7: Robot for Thermographic inspection - Path Planning [2]

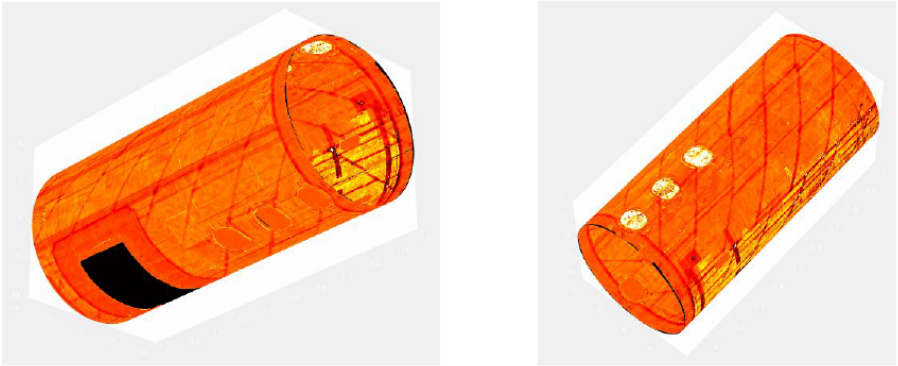


Figure 1.8: Robot for Thermographic inspection - Image Stitching [2]

Figure 1.8 shows the results of the Thermographic NDT process after the images have been stitched and registered to its location on the fuselage structure. Two 3D views of the processed and registered thermal data from a complete inspection of the composite fuselage are shown ([2]).

1.2.1.3 Case Study 3: Ultrasonic Inspection of Panels

In this case study ([3]), we study the ultrasonic inspection of panels and its associated path planning process. Although the ultrasonic scanning is done at a point and is not a camera based technique, this case study is important because it provides an example of an automated coverage path planning approach.

In Figure 1.9, we see how the robot path is being planned. A slicing algorithm is used which uses the XY plane offset at various heights for the generation of the paths. The

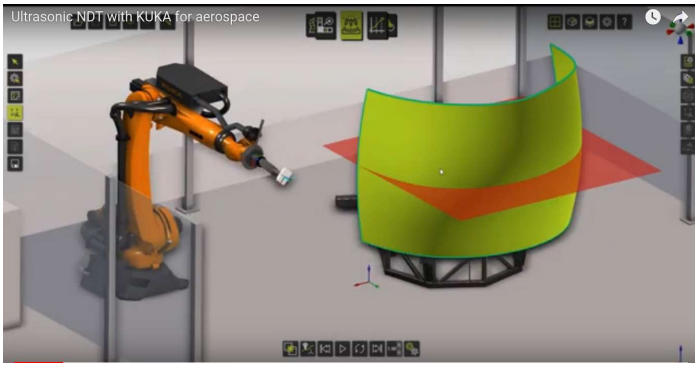


Figure 1.9: Ultrasonic Inspection using Robot - Path planning [3]

slicing approach is an example of geometry based technique for automated coverage path planning.

The simulation of the robot following this path is shown in Figure 1.10. The motion of the robot holding the end effector (Ultrasonic NDT device) is combined with the rotation of the base table in order to scan the panel according to the path planning process.

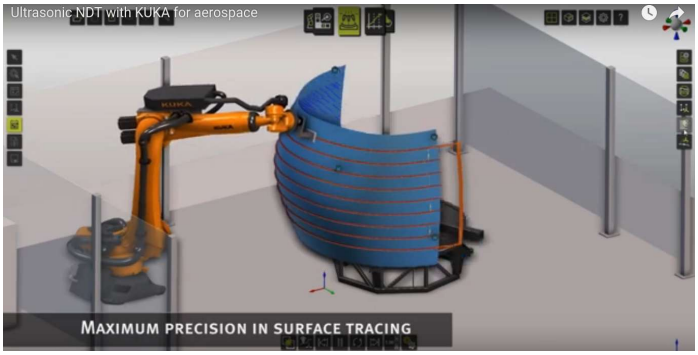


Figure 1.10: Ultrasonic Inspection using Robot - Simulation [3]

It is possible to change the slicing plane angle in order to generate a different path. This process is shown in Figure 1.11.

Hence, this case study is a demonstration of Coverage Path Planning of a Panel for ultrasonic NDT inspection using slicing techniques.

1.2.1.4 Case Study 4: Laser Shearography inspection using Robotics

The YouTube video [4] describes a Robotic arm based Laser Shearography system for the scanning of panels. There are not much details provided on the path planning or image stitching processes. The automated inspection in progress is shown in Figure 1.12.

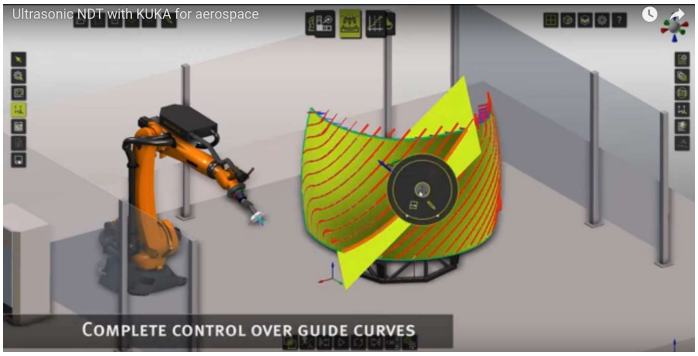


Figure 1.11: Ultrasonic Inspection using Robot - Changing the Slicing Plane [3]

Figure 1.12: Laser Shearography Inspection using Robot [4]

There are not much differences in the case of Laser Shearography as compared to that of Thermography, as both of them are camera based techniques. Here, a set of laser diodes is used to illuminate the panel. The distortion caused by loading the panel is captured by laser interferometric techniques in order to find the defects. Here the loading is applied by putting the entire system into a vacuum chamber and applying a vacuum load. For the remainder of the present work, we refer only to Thermography, but the process described applies equally well to Laser Shearography as well. Hence this case study is mentioned.

1.2.1.5 Case Study 5: Bicycle Inspection using Thermography

Peeters et al [5] describes a method of inspecting a composite bicycle frame for defects using Thermographic NDT. The process is shown in progress in Figure 1.13. A coverage path planning method has been described in [5] for this application. In that method, an optimization approach using genetic algorithms has been used to optimize the viewpoints

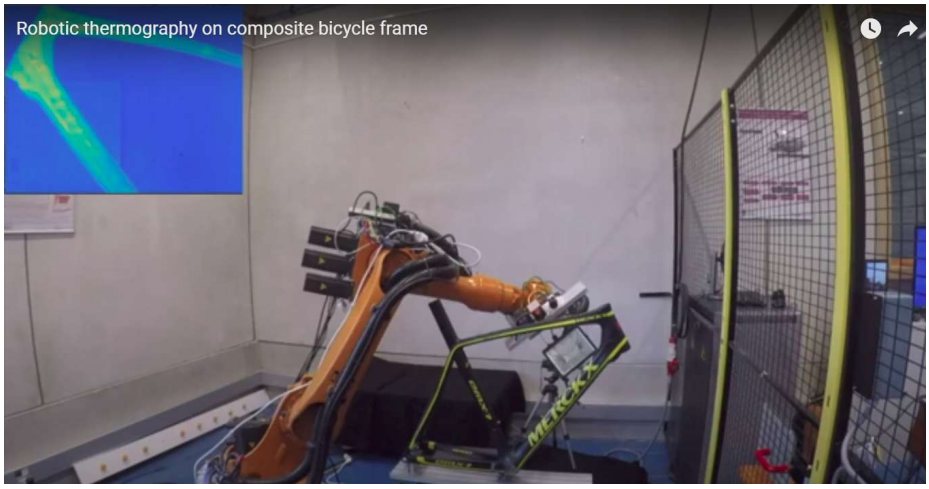


Figure 1.13: Bicycle Inspection using Robotic Thermography system [5]

for coverage of the entire bicycle frame. The approach is illustrated well in the YouTube video [16].

The process described however appears to be overly complicated. The application is also dubious as the inspection of a bicycle is not an activity important enough to warrant the installation of an expensive robotic arm and Thermographic NDT apparatus. However, this case study appears to be unique in the sense of having a published literature for coverage path planning using Thermographic NDT. Hence it is mentioned for completeness sake.

1.2.2 Motion Tracking Systems

In this section, we will explore various examples in which a motion tracking system has been used for the augmentation of Non-Destructive Testing.

1.2.2.1 Case Study 1: NASA Inspection of Aircraft Fuselage

[6] demonstrates an example where motion tracking has been used for Thermographic NDT. In this case study, the Thermographic NDT System consisting of the cooled Infrared camera and heating system has been mounted on a boom crane (Figure 1.14). This crane or jib consists of an arm which can be stretched out and a counterweight is mounted at the end in order to offset the weight and make it easier for the operator to manually move it. This system is conceptually similar to Figure 1.1b, except that the pose is captured by motion tracking and not through joint encoders.

Figure 1.15 shows a close in (zoomed) view of the motion tracking target mounted to the hood of the flash thermography system. We can see that fiducial markers similar to

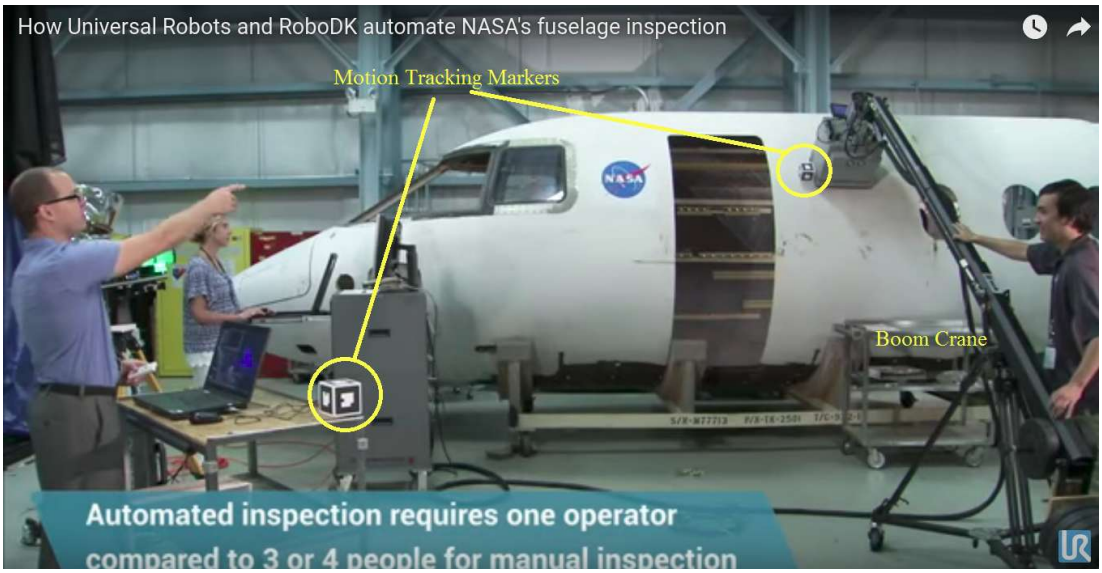


Figure 1.14: Aircraft Fuselage Inspection - Motion Tracking [6]

April Tags are being used.

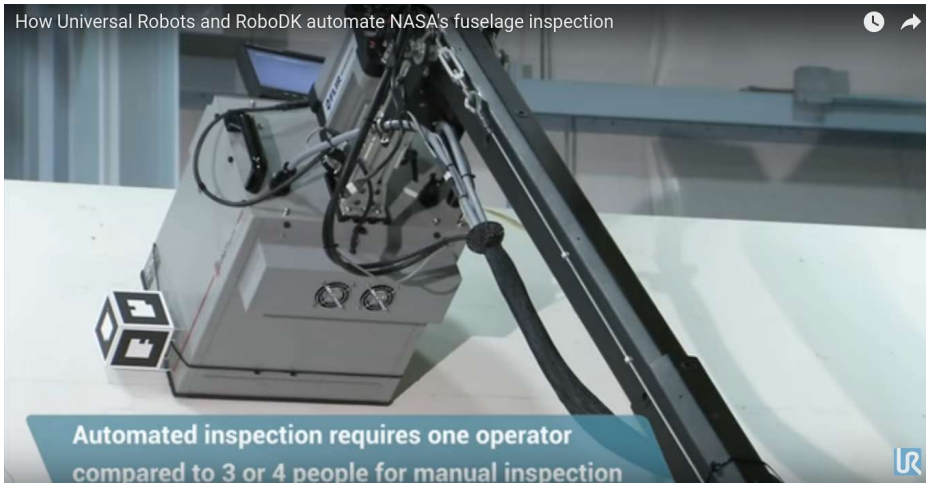


Figure 1.15: Aircraft Fuselage Inspection - Motion Tracking (Zoomed) [6]

Unfortunately, there seems to be no paper explaining how this Motion tracking system works or how it is integrated with the Thermographic NDT system. [6] gives no such details.

1.2.2.2 Case Study 2: TecnaTom WIIPA Cell

In this case study ([7]), we see how a motion tracking system has been combined with a handheld ultrasonic sensor to create a "WiiPA Cell". Figure 1.16 shows how motion

Figure 1.16: WIIPA Cell - Motion Tracking [7]

tracking cameras are mounted on the structure and observing the movement of the scanning done by the operator. The ultrasonic scanner has spherical motion tracking targets mounted to it.

Figure 1.17 shows the model of the motion tracking targets in a 3D viewer as it is being captured by the motion tracking cameras. Multiple view geometry is applied to compute the pose information.

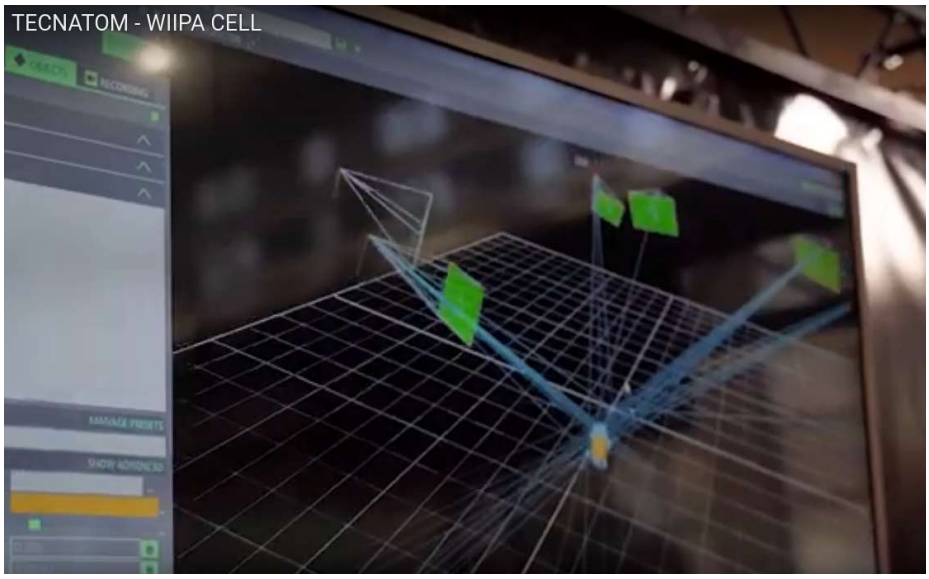


Figure 1.17: WIIPA Cell - Motion Tracking (Virtual model) [7]

Figure 1.18 shows a closer view of the ultrasonic sensor with the motion tracking targets mounted. We can also see how the data from the ultrasonic sensor is combined using the

3D model of the panel and displayed on the screen.

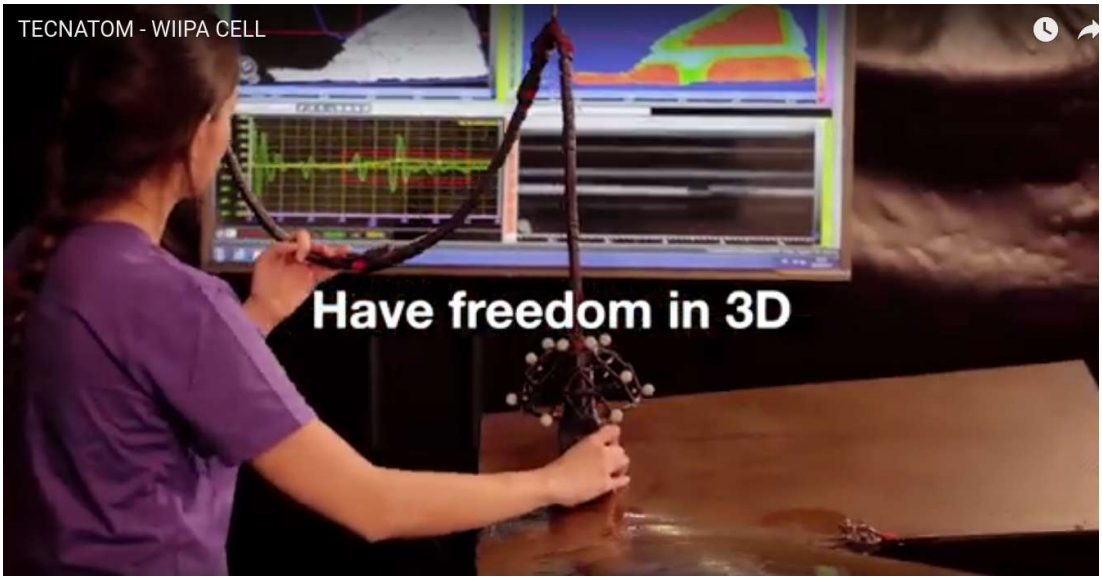


Figure 1.18: WIIPA Cell - Motion Tracking (Data viewing) [7]

Unfortunately, there is no information available as to how this Motion tracking based augmented Ultrasonic NDT System has been implemented by Tecnatom. The knowledge appears to be proprietary.

1.2.2.3 Case Study 3: Tecnatom Wiipa Lite Portable

This case study ([8]) demonstrates how a motion tracking system can be used for in-situ or field inspections. The Inspection system is provided as a kit (Figure 1.19) consisting of the ultrasonic NDT equipment with motion trackers mounted to it, a tripod for mounting the motion tracking cameras, computer and electronic equipment for processing the data from the motion tracking system and the NDT system and displaying it on the screen.

Figure 1.19: WIIPA Lite - Inspection Kit [8]

Figure 1.20 shows inspection in progress using the WIIPA Lite Kit. Here we can see the operator manually scanning the steel surface as the motion tracking is being done by the tripod mounted background cameras. The results are being displayed on the laptop screen using 3D viewers.

Unfortunately, there is no information available as to how this Motion tracking based portable augmented Ultrasonic NDT System has been implemented by Tecnatom. The knowledge appears to be proprietary.

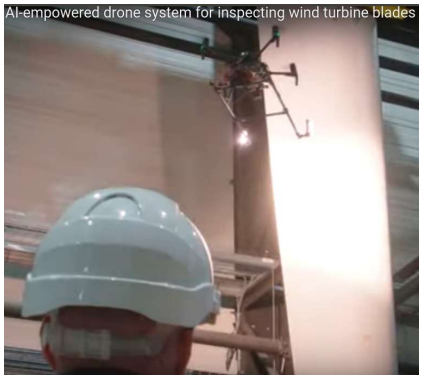
1.2.3 Drone based Inspection

While Drone based visual camera inspections are quite common, the concept of a drone carrying out Non-Destructive Testing using Thermography is quite rare. [9] gives an example of a drone based active thermography system named 'Firefly' for inspection of aero structures and wind turbine blades.

Figure 1.21a shows the Firefly Drone. Figure 1.21b shows the drone inspecting a wind turbine blade. Infrared lights are mounted on the drone for active thermography.

Figure 1.20: WIIPA Lite - Inspection in Progress [8]

(a) Firefly Drone [9]



(b) Inspection of Wind Turbine [9]

Figure 1.21: Firefly Drone System

The Firefly Drone System also incorporates a motion planning system using a 3D viewer to plan the viewpoints at which the drone will stop and scan the panel for defects. This can be seen in Figure 1.22 where the path planning is being done to cover an aircraft wing structure.

The Firefly Drone can also be used for in-situ inspection of aircraft. This can be seen in Figure 1.23.

Unfortunately, there are no papers or additional information explaining how the motion planning for the drone is being done. The knowledge appears to be proprietary.

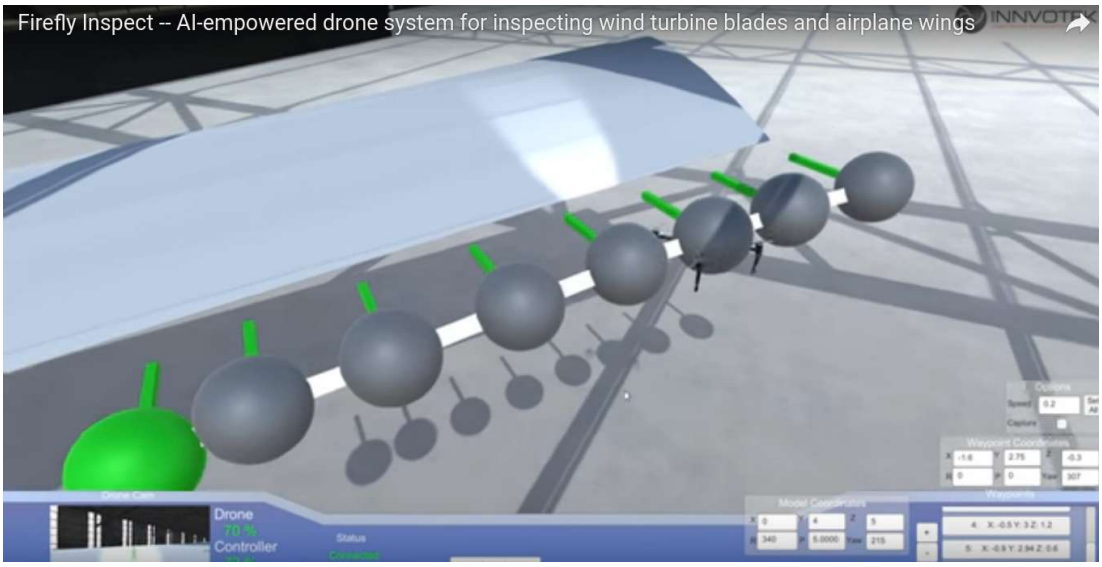


Figure 1.22: Firefly Drone - Motion Planning [9]

Figure 1.23: Firefly Drone - Inspecting Aircraft [9]

1.3 Summary

We have covered various examples of the augmentation of Non-Destructive Testing using Thermographic and Ultrasonic techniques. These case studies involve the use of Robots/Drones, Motion Tracking systems and have sophisticated algorithms for planning the paths and displaying the recorded data. In most of the above case studies, the exact methodology is rarely disclosed. Therefore we cannot be sure how complete or versatile these solutions

are.

Thus, the motivation behind the current study is to develop and demonstrate methods of NDT augmentation for Camera based NDT systems and document the thought process and algorithms behind their working.

1.3.1 Approach of the current study

The case studies covered in the literature survey all use professional, commercially available systems for Non-Destructive Testing, Robotics, Drones and Motion tracking systems. The 'solution' thus obtained is a system integration provided for a particular application, using this commercially available industrial equipment.

We depart from this approach and instead developed prototype hardware and equipment for the demonstration of the algorithms. The reasons for this is mainly budgetary, as the required industrial grade robots, NDT Systems and Motion tracking systems were not available to us. However, the construction of such prototype equipment enabled us to get a deeper understanding into the workings of such systems and also the limitations of using such prototype hardware. This would enable us to make better decisions in construction of an industrial grade system or acquisition of the same.

For the remainder of the thesis, the hardware to be used is as follows:

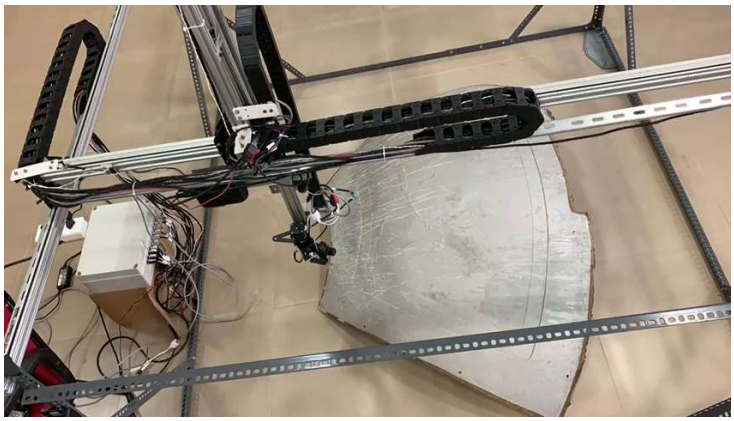
1. NDT System: An ELP IP Camera is used as a stand in for Camera based NDT Systems like Thermography and Laser Shearography

2. Robotic system: A prototype 5-axis cartesian robot is constructed which can scan over a panel of size up to 2m x 2m x 1m. This is described in more detail in Chapter 2 and can be seen in Figure 1.24a.

3. Motion Tracking system: A prototype motion tracking system was constructed using two ELP USB cameras mounted on a tripod in a stereo configuration (Figure 1.24b). April Tags are used as fiducial markers for tracking. It is also described in more detail in Chapter 3.

The Software is developed in C++ and OPENGL. OpenCV is used for computer vision/motion tracking. No proprietary softwares are being used.

The work done is presented as follows. Chapter 2 describes the prototype 5-axis cartesian robot constructed for the demonstration of algorithms. Chapter 3 describes the hardware and software for motion tracking and its integration with the NDT system. Chapter



(a) 5 Axis Cartesian Robot

(b) Motion Tracking Setup

Figure 1.24: Prototype Hardware used in the Project

4 describes the process of manually planning the viewpoints in the virtual 3D model and stitching the images thus obtained onto the 3D model of the panel. Chapter 5 describes the process of automatically planning the viewpoints for full coverage of a panel. Finally Chapter 6 summarizes and concludes the thesis.

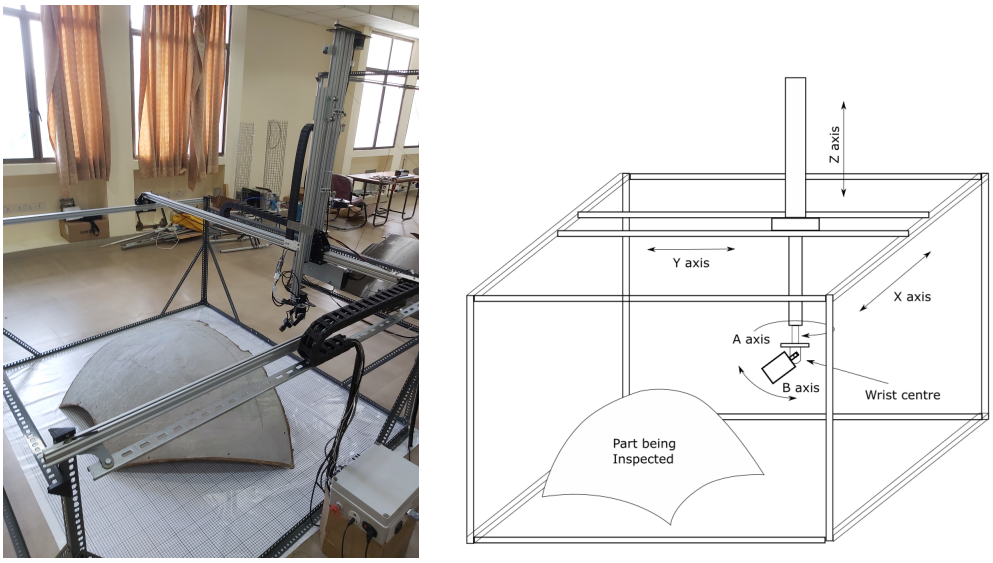
Chapter 2

Development of the Prototype 5 axis Cartesian Robot for scanning

This chapter describes the design and development of the 5 axis prototype robot that is used for scanning over a panel using a camera. The aim was to develop a mechanism which can scan over the required workspace (2m x 2m x 1m) using a camera which records visual images. The camera is a stand in for the actual NDT imaging setups. The robot can be controlled by a CNC controller which accepts G-code. It is to be noted here that we have used low budget equipment to develop the robotic system and also the CNC controller is also self developed. This was done in order to keep the cost down. Subsequently, the robot realized is good for demonstrating the proof of concept (it is used further in Chapter 4 - Viewpoint Planning and Image Stitching), but has some severe limitations when it comes to speed, accuracy and reliability. We also explore the limitations of the robot in this chapter in order to develop a better quality robotic system in the future. Section 2.1 describes the design and operation of the 5 axis cartesian robot and Section 2.2 explores the accuracy limitations of the robot using metrological methods.

2.1 Design and Development of the Cartesian Robot

The 5-Axis robot shown in Figure 2.1a was designed, developed and tested. Figure 2.1b shows the conceptual design of the robot. This 5-Axis robot has motion in all 3 cartesian axes (X,Y,Z) and 2 rotation axes at the end effector (A,B). It can inspect a panel of dimensions within the volume of 2mx2mx1m. The end effector carried by the robot is a small IP camera which is used to represent camera based NDT techniques such as thermography and laser shearography. This robot is a proof of concept to demonstrate our various algorithms and techniques.



(a) 5 Axis Cartesian Robot **(b)** Robot concept diagram

Figure 2.1: Cartesian Robot

The alignment and accuracy testing of the robot was a point of particular concern. We had to check and fix the parallelism of the X-axis motions and straightness of Y and Z axis. The current structure has significant compliance in the Z direction. We also tested the perpendicularity of XZ and YZ axes using line laser. The accuracy of the robot is also poor, with an error of the order of 1cm per metre of travel. This is due to the low power and inertia ratio of the motor and motor driver being used.

The 5-axis cartesian robot developed is used for the demonstration of motion planning and image stitching algorithms developed in the later chapters (Chapter 4 - Viewpoint Planning and Image Stitching).

2.1.1 Hardware design

The hardware or mechanical structure of the robot is constructed out of aluminium frames using T-slot 2040 profiles of size 20 mm x 40 mm. Also some bent angle steel rails (typically used for shelving) are used for constructing the base of the cartesian robot. The cartesian axes of the robot are sliding on T-slot roller wheels and are actuated by Nema 17 stepper motors. The X and Y axes are driven by a belt and pulley setup while the Z axis is driven by a leadscrew. The two rotary axes (A and B) are direct driven. The wiring is routed through three cable drag chains attached to the various axes and limit switches are mounted for homing the X, Y and Z axes. These machine components are used in desk-

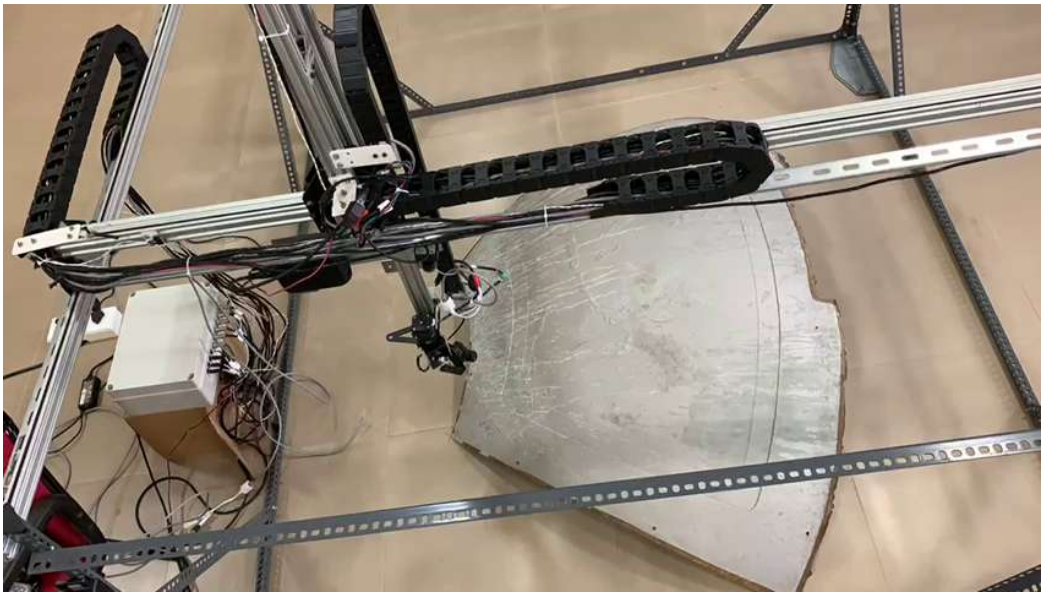


Figure 2.2: Cartesian Robot Top View

top sized and hobbyist machine building and can be readily procured from online vendors. The camera used is an ELP IP camera with resolution 1260x768 (camera parameters are described further in Chapter 4 - Viewpoint Planning and Image Stitching). The ELP IP Camera was used as it was readily available and not expensive.

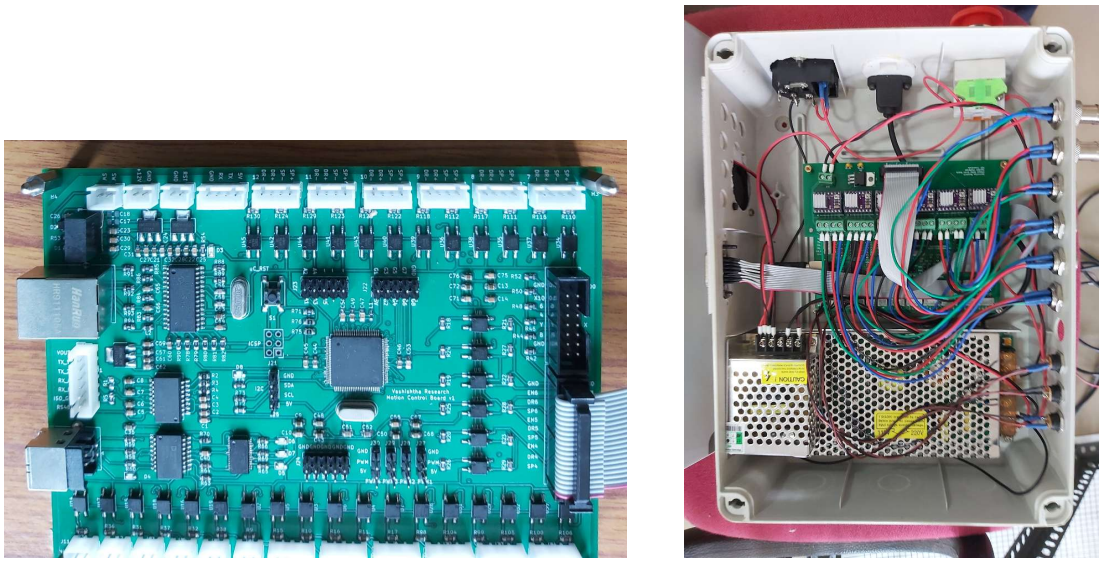
The top view of the robot as it is scanning over the panel is shown in Figure 2.2.

2.1.2 Electronics and Firmware development

We have developed a motion control board (Figure 2.3a) which can control up to 12 axes of motion independently and also has additional features such as MPG port, limit switches and RS485/ethernet communication. This control board is connected to stepper drivers via a stepper driver board (6 motors) for controlling the Nema 17 motors in the cartesian robot. The motion control board and stepper drivers are placed into a custom made enclosure (Figure 2.3b).

2.1.3 Operation of the Cartesian Robot

The cartesian robot can be controlled in two ways. It accepts G-code via the serial port and executes the simultaneous 5 axis motion per block of the G-code. The G-code can also be used to do the homing of the cartesian axes and to set the offsets. Another method of operating the robot is to use the MPG pendant for jogging. This is used for making small



(a) Motion Control Board **(b)** Enclosure

Figure 2.3: Electronics Development

motions and primarily used for the workpiece registration process.

2.1.3.1 Running G-Code

G-code can be generated by higher level path planning algorithms (this is described in subsequent chapters 4 and 5). In order to control the robot, G-code can be sent to the CNC controller using the Serial port. A standard laptop is connected to the controller via a USB cable and the G-code can be sent line by line to control the robot.

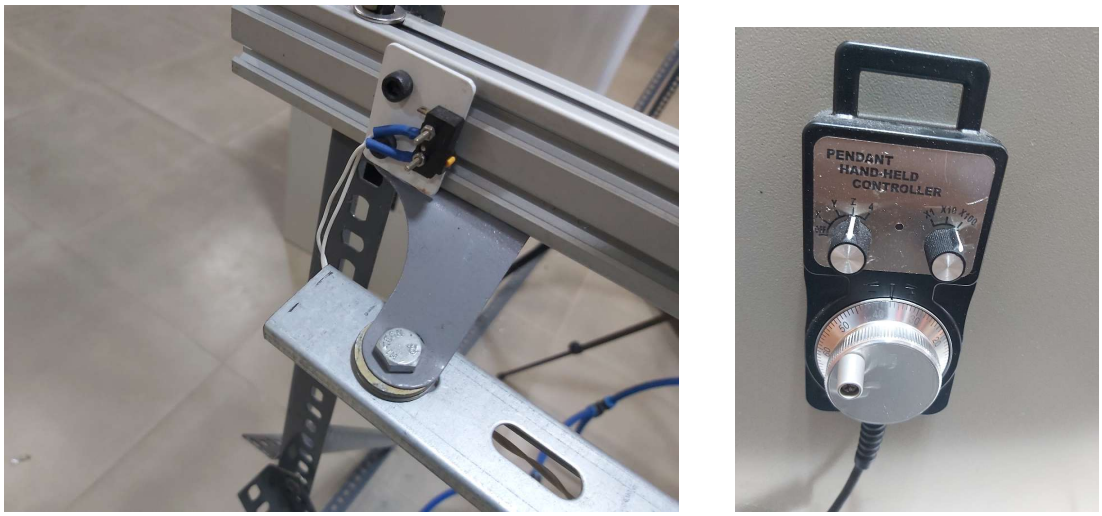
An example of the G-code being sent to the robot is shown below. This G-code corresponds to 14 viewpoints. The G-code is sent one line at a time and the robot is stopped at each viewpoint to record the image. Note that these viewpoints can be manually generated using the process in Chapter 4 - Viewpoint Planning and Image Stitching, or automatically generated using the process in Chapter 5 - Semi Automated Coverage Path Planning.

```
G1 X352.726 Y-467.058 Z525.336 A118.285 B32.0049
G1 X484.301 Y-310.149 Z556.448 A132.84 B29.2058
G1 X560.003 Y-111.854 Z593.692 A148.67 B26.9281
G1 X636.424 Y80.3811 Z597.466 A165.944 B26.6291
G1 X612.651 Y306 Z620.113 A-173.971 B24.0599
G1 X473.815 Y240.77 Z675.506 A175.281 B18.4046
G1 X518.419 Y64.7916 Z645.843 A159.919 B22.4305
```

```
G1 X418.561 Y-157.278 Z633.419 A138.859 B22.4277
G1 X344.812 Y-339.43 Z589.553 A122.896 B26.5552
G1 X180.365 Y-307.811 Z637.245 A112.019 B22.5205
G1 X318.043 Y-155.081 Z661.977 A127.904 B20.8013
G1 X384.481 Y89.3397 Z693.141 A157.772 B16.3731
G1 X371.09 Y311.243 Z706.341 A-172.454 B14.6314
G1 X255.849 Y-47.0172 Z701.889 A134.172 B14.802
```

2.1.3.2 Homing using limit switches

Limit switches (eg. X axis limit switch as shown in Figure 2.4a) are used to set the home position of the robot. The robot is moved at the homing speed rate until it hits the limit switch, then it moves backward and then forward at a very slow speed until the limit switch is precisely triggered. This enables us to set the zero or home position of the robot, which can be very useful if we want to move the robot with reference to its home position. In practice this technique is rarely used, since we want the robot to move relative to the work-piece.



(a) Limit Switch **(b)** MPG Pendant

Figure 2.4: Control of the Cartesian Robot

2.1.3.3 Jogging using Pendant

The MPG (Manual Pulse Generator) Pendant used is shown in Figure 2.4b. This pendant can be used to manually give small motions to the robot. We select the axis of motion

(X,Y,Z,A,B), the distance to be moved per revolution of the handwheel (1mm, 10mm, 100mm), and then rotate the handwheel manually to give small adjustments to the position of the robot. In practice, this is mainly used in the workpiece registration procedure.

2.1.3.4 Workpiece registration procedure

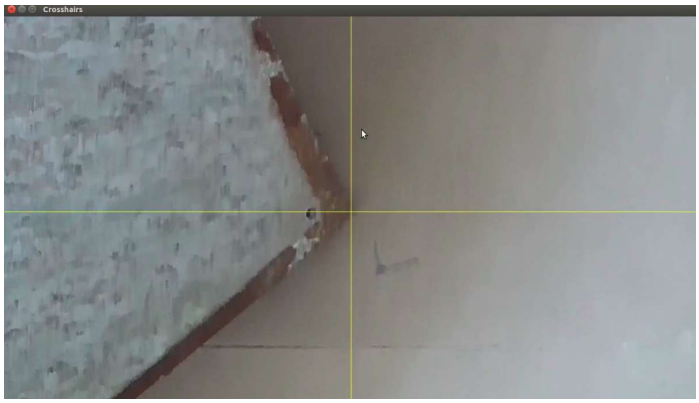


Figure 2.5: Workpiece Registration - Crosshairs view

In this process we align the workpiece (ie. panel) to the coordinates of the robot workspace and vice versa. For this process live view from the camera with crosshairs superimposed on top of it is streamed to the user (Figure 2.5). We put the camera pitch axis vertically down and align the yaw axis so that the crosshairs are in the same direction as X and Y axis. This is needed to zero the rotary axes. We then manually place the workpiece panel into the robotic cell such that the workpiece corner point is exactly positioned at the centre using crosshairs. Manual Pulse Generator (MPG) pendant is used for fine motion for this purpose.

We will then obtain the offset coordinates ie the current X, Y, Z positions of the robot. These offset coordinates can be fed to the controller using G-code. We can then move the robot end effector to the expected location of another corner point and gently rotate the panel (without shifting the first point) until that corner point comes into position. Since this workpiece rests on a plane, 2 points are sufficient for workpiece registration. Similar processes can be done easier if the robot is equipped with a probe or laser displacement sensor. This process is needed to make sure that the robotic system with the panel conforms to the digital models that are used for path planning in Chapter 4 (Viewpoint Planning and Image Stitching).

2.2 Metrology analysis of the Cartesian Robot

The following analysis was done to ascertain the metrological accuracy of the 5-axis Cartesian robot. The various metrology parameters were quantified and and efforts were taken to correct the errors. The reasons for poor performance were identified so that they can be corrected in the design of an actual professionally made robot. This section summarizes dimensional metrology (See [17] for a more thorough understanding) as applied to the design and analysis of Machines.

2.2.1 Measurement Parameters Techniques used

The robot currently has a great deal of swing in the Z-axis which makes contact measurements using a dial gauge and reference plates/blocks unreliable. Hence currently we have used less accurate means such as measuring tape and line lasers. Since the next version of the robot will be more rigid, we will be able to determine metrology parameters and alignment to a much greater accuracy than the current methods.

2.2.1.1 Straightness

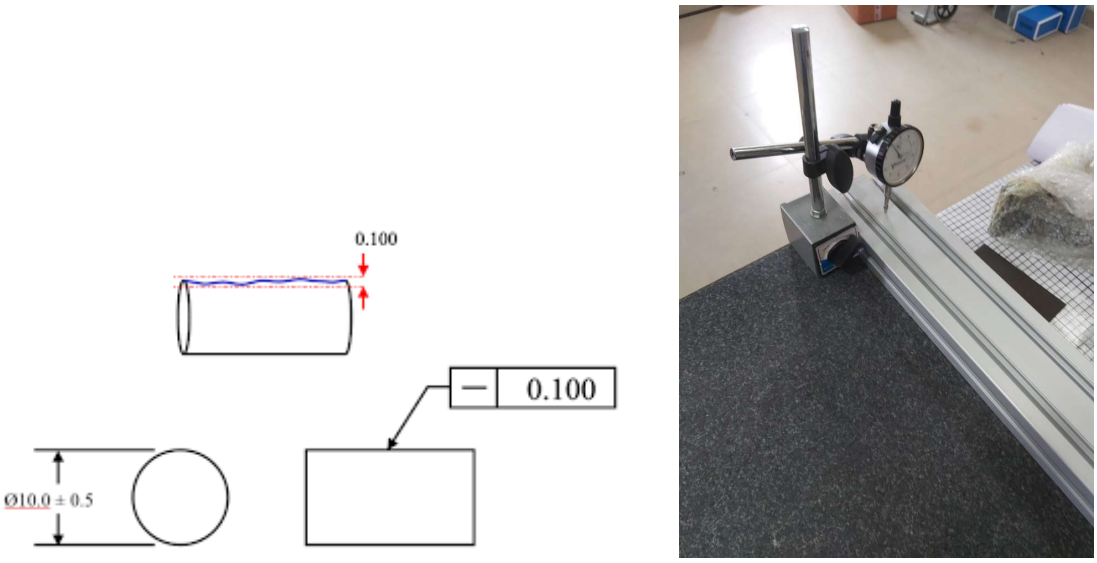
Straightness is a measure of how much a rod or column deviates from an ideal straight line throughout its length. The distance between the two fitting lines (maximum and minimum deviation) gives us the value of straightness. Figure 2.6a shows us how straightness is defined and the desired tolerance on straightness is indicated.

Measurement of straightness

We do not have a direct way to measure the straightness of the axis due to low rigidity of the robot and unavailability of a long enough reference. Instead, we measure the straightness of the structural elements (beams) that are used in the X, Y, Z machine frame. A dial gauge is fixed on a granite surface plate and the column is fed through along its length (Figure 2.6b). The deviation of the dial gauge (maximum – minimum value) is observed and noted.

Type of column (structural element)	Straightness (mm)
4080 AL channel	0.03
2040 AL channel	0.03

Table 2.1: Straightness measurement



(a) Straightness Definition **(b)** Measuring Straightness

Figure 2.6: Straightness

The value of straightness was observed to be within 0.03mm for a structural element length of 600mm. (Table 2.1) This value is very low compared to other deviations we will observe. There will be an increase in bending if load is applied to the beam. The deflection can be predicted using finite element analysis and it is based on the load applied. We should choose the structural element according to how much deflection is predicted (from FEM) and whether it is in the tolerance zone.

2.2.1.2 Flatness

Flatness is defined for a surface, usually XY-plane. It is a measure of how flat the surface is compared to an idealized plane. This distance between the two fitting planes (containing the whole surface) gives us the value of flatness (Figure 2.7).

Measurement of flatness

Flatness of XY-plane should be measured with a dial gauge with reference to a flat surface (eg. granite surface plate), but the low rigidity of the robot prevents such a measurement. Instead we calculate flatness in a round-about way – we assume that the ground is flat, and take measurements of the 2 X-axis rails and the Y-axis rails from it using measuring tape. We also adjusted the levelling on the floor to make these variations as small as possible.

Table 2.2 shows the measured values of flatness ie the variation in value according to our measuring technique. We can conclude that the flatness of the XY-plane of the robot is

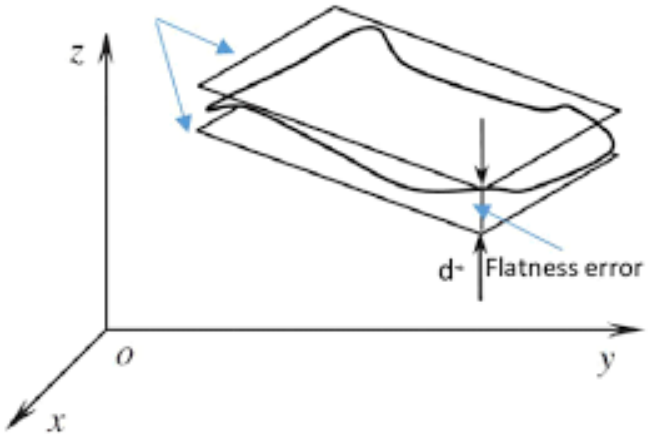


Figure 2.7: Definition of Flatness

Flatness Measurement	Variation in value (mm)
X axis 1 from ground	4
X axis 2 from ground	5
Y axis from ground	3

Table 2.2: Flatness measurement

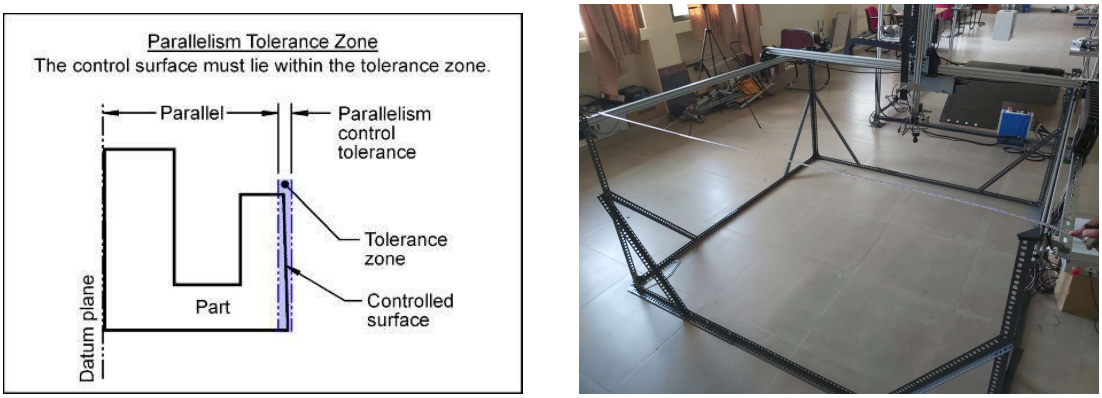
of the order of 5mm. This is a fairly high deviation and contributes to the inaccuracy of the robot (especially with reference to the workpiece).

2.2.1.3 Parallelism

Parallelism is a measure of how much two straight lines run parallel to each other. In the context of this robot, it is in our interest to make sure that the two X-axis rails run completely parallel. Taking one line as a reference, the distance between the two lines that fit the second line completely gives us the measurement of parallelism (Figure 2.8a).

Measurement of Parallelism

We use the measuring tape to measure the distance between the two X-axis rails at regular intervals (Figure 2.8b). The pairs of points were identified by measuring from corresponding ends and marking points at regular intervals. The maximum variation in this distance (thus parallelism) is estimated as **8mm**. We can adjust two crossbars at the end of the X-axis rails to improve on the parallelism value.



(a) Parallelism Definition **(b)** Measuring Parallelism

Figure 2.8: Parallelism

2.2.1.4 Perpendicularity

Perpendicularity is measure of how close the angle between two lines (or planes) is to a right angle (Figure 2.9. In a Cartesian robot, our interest is to keep the X, Y and Z axes all mutually perpendicular to the greatest extent possible. Perpendicularity is also sometimes called squareness.

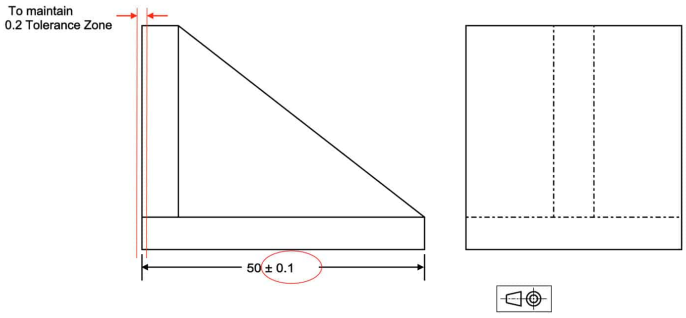


Figure 2.9: Definition of Perpendicularity

Measurement of Perpendicularity

Square edge with dial gauge is typically the method of estimating perpendicularity but we cannot use this method due to the low rigidity of the system. Instead we are using Bosch GLL 3x Line laser which projects two laser lines as a cross. This can be used as the reference to measure or adjust the perpendicularity of the system.

Values of perpendicularity were not noted. It was observed to be reasonably perpendicular. The YZ perpendicularity was adjusted using the location of the wheel bearings. The XY perpendicularity was adjusted by moving one side of the gantry. The XZ perpendicularity is not amenable to adjustment owing to the high swing in the system.

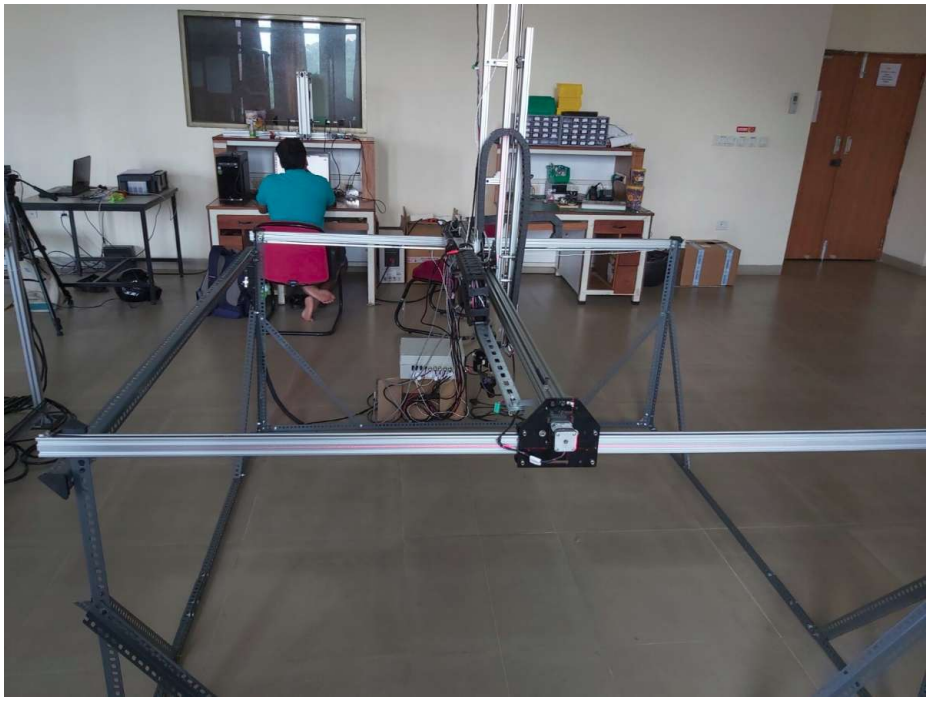
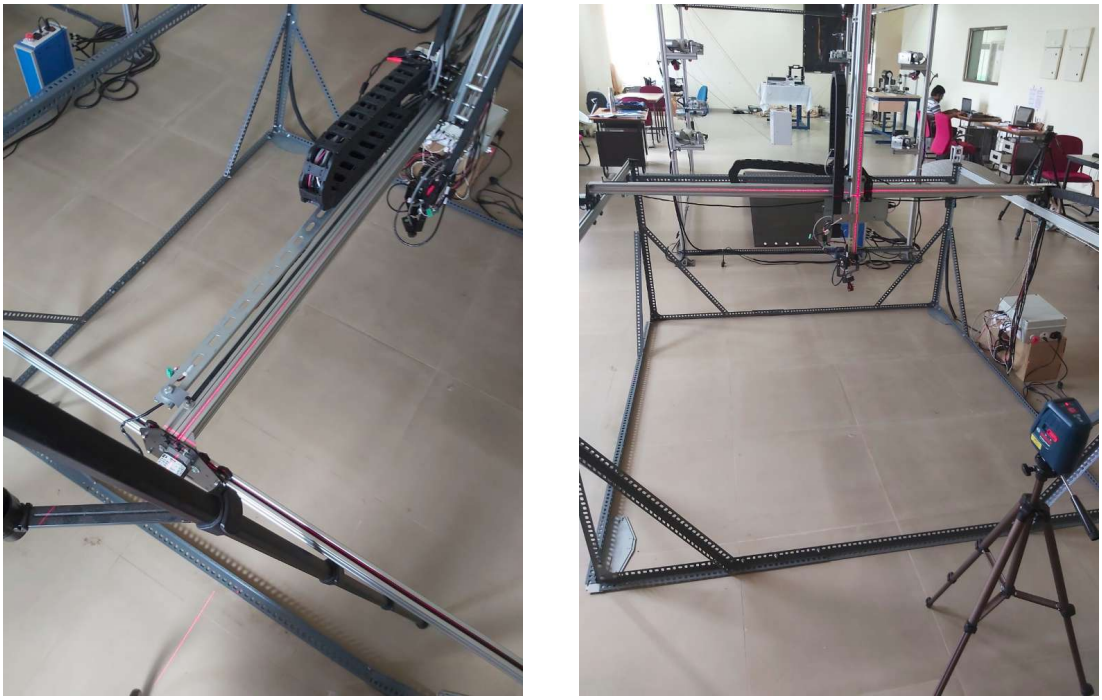


Figure 2.10: Measuring Perpendicularity of X and Z axes



(a) Alignment of X and Y axis **(b)** Alignment of Y and Z axis

Figure 2.11: Measuring perpendicularity of XY and YZ axes

2.2.2 Accuracy and Repeatability

In a set of measurements, accuracy is closeness of the measurements to a specific value, while repeatability is the closeness of the measurements to each other. Figure 2.12 demonstrates the concept of accuracy and repeatability.

We can measure the accuracy and repeatability of the motions in a Cartesian robot. For example, the X-axis can be moved by a set distance, and the actual distance moved can be measured. The difference will give us a measure of the accuracy. Repeatability means if an axis is moved by a certain distance, and then moved back by the same amount, the difference from the starting point would be a measure of repeatability.

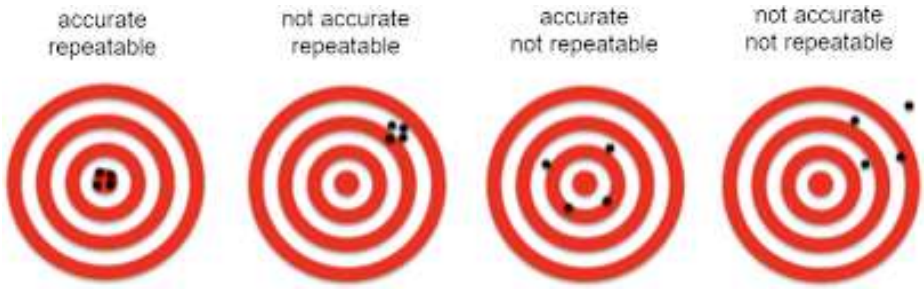


Figure 2.12: Accuracy and Repeatability

In the professional machine tool/ robotics industry, laser trackers are used to estimate accuracy and repeatability. However, since this was not available, a measuring tape is used as the reference to find the distance moved. This however only has an resolution of 1mm so we cannot get readings finer than that. Accuracy is estimated by taking measurement from a reference (such as the end of the rail) after moving the axes by a certain distance. Repeatability is estimated by moving the axis back and forth by a known distance and measuring the deviation. These experiments are repeated a number of times for each axis of motion. The accuracy of the rotary axes is not considered as we do not have a good way of measuring that. The overall accuracy of the robot is also not estimated because that would require a sophisticated sensor like a laser tracker.

2.2.2.1 X axis

Figure 2.13 demonstrates the method of determining accuracy and repeatability of X and Y axes. A measuring tape is used to measure the distance between the end point of the base structure and a specific point on the moving axis. Before motion, the initial distance is recorded as the start position, and after giving a specific commanded distance (via g-code),

we record the distance measured as the end position. The difference between the start and end position gives us the distance moved, and we can compare this with the commanded distance to get the deviation. Accuracy would be measured by dividing this deviation (in mm) by the distance moved (in m) and the value is given in units of mm/m .Table 2.3 records the data for the X-axis accuracy measurement experiments.

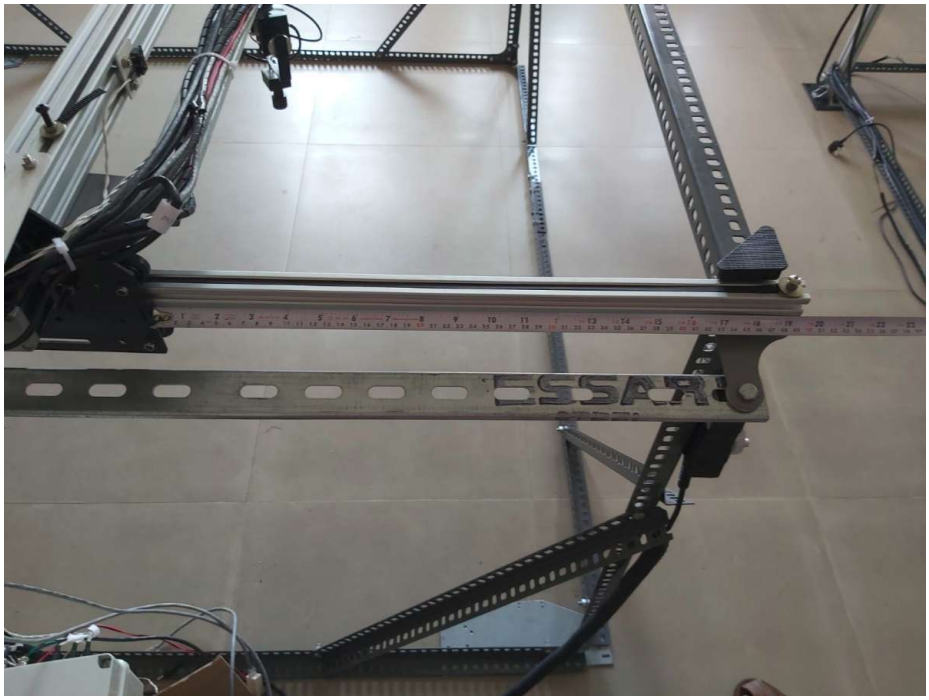


Figure 2.13: Measuring Accuracy and Repeatability of the X-axis

Start position (mm)	End position (mm)	Distance moved(mm)	Distance commanded(mm)	Deviation (mm)	Accuracy (mm/m)
704	806	102	100	2	20
806	907	101	100	1	10
907	1009	102	100	2	20
1009	1111	102	100	2	20
387	796	409	400	9	22.5
796	1203	407	400	7	17.5
1203	393	810	800	10	12.5

Table 2.3: X axis accuracy measurement

The average accuracy of X axis motion is thus 17.5 mm/m

For repeatability, axis is moved forward by 100mm and back by 100mm. The deviation in the start and end value indicates the repeatability. The results of the repeatability test are shown in Table 2.4.

Start position (mm)	End position(mm)	Deviation (mm)
393	396	3
396	398	2
398	398	0

Table 2.4: X axis Repeatability

The average repeatability of X-axis is thus 1.67 mm.

2.2.2.2 Y axis

The Y-axis measurements are made in the exact same manner as that of the X-axis. The results of the Y axis accuracy testing are shown in Table 2.5, and that of repeatability are shown in Table 2.6.

Start position (mm)	End position (mm)	Distance moved(mm)	Distance commanded(mm)	Deviation (mm)	Accuracy (mm/m)
517	621	104	100	4	40
621	721	100	100	0	0
721	823	102	100	2	20
823	924	101	100	1	10
924	823	101	100	1	10
823	721	102	100	2	20
721	619	102	100	2	20
619	516	103	100	3	30
414	918	504	500	4	8
918	408	510	500	10	20

Table 2.5: Y axis accuracy measurement

The average accuracy of Y-axis is thus 17.8 mm/m

For repeatability, axis is moved forward by 100mm and back by 100mm

The average repeatability of Y-axis is thus 1.0 mm

Start position (mm)	End position(mm)	Deviation (mm)
408	409	1
409	410	1
410	409	1

Table 2.6: Y axis Repeatability

Figure 2.14: Measuring Accuracy and Repeatability of the Z-axis

2.2.2.3 Z axis

Figure 2.14 shows the method of measuring the accuracy and repeatability of the Z-axis. As before, the deviation from the commanded distance to the actual distance moved is used to estimate the accuracy. The results of the Z axis accuracy testing are shown in Table 2.7, and that of repeatability are shown in Table 2.8.

The average accuracy of Z axis motion is thus 6.6 mm/m. However it must be kept in mind that the measuring tape can only measure upto 1mm resolution. In practice the difference between commanded and measured distance is not really measurable by the tape.

The repeatability is too small to be measured using the measuring tape.

Start position (mm)	End position (mm)	Distance moved(mm)	Distance commanded(mm)	Deviation (mm)	Accuracy (mm/m)
135	185	50	50	0	0
185	236	51	50	1	20
236	286	50	50	0	0
286	236	50	50	0	0
236	186	50	50	0	0
186	135	51	50	1	20

Table 2.7: Z axis accuracy measurement

Start position (mm)	End position(mm)	Deviation (mm)
135	135	0
135	135	0
135	135	0

Table 2.8: Z axis Repeatability

2.2.3 Observations and Summary

The Z-axis is fitted with a lead screw and pulley gear reducer. This may explain the high level of accuracy observed compared to X and Y axes. The X and Y axis use a belt drive which is more inaccurate. However the main reason for the inaccuracy is the inadequate motor and motor driver used. The NEMA 17 motor used is too small for such a large robot causing the inertia ratio to be too large. For the x axis the ratio of inertia of load to motor is around 28.6, while the maximum recommended for stepper motors is 10. Also the driver used (DRV8825) is too small for driving such a large load. These factors contribute to the poor accuracy and repeatability in the X and Y axes. Adjusting the motor current down was found to make the accuracy a little better but not by much.

Estimation of accuracy of A and B axes (rotary motions) was not done, due to lacking appropriate equipment. However since these axes are directly driven by the stepper motors and are relatively small and lightweight, they are expected to be reasonably accurate. Errors in these axes may come due to incorrect homing as we have no sensor to set the zero position.

The robot in its current form is highly inaccurate. However, it is good enough for a very rough proof of concept/demonstration purpose. For developing a professional version of the robot, it will be designed to eliminate all the observed problems. Heavy structural

elements and rigid design will remove swing and deflection in the system. Accurate adjustment for the alignment parameters will be made possible and appropriately heavy sized NEMA34 motors and motor drivers will be provided to meet the requirements for speed, torque and inertia ratio. We should also design metrological methods to measure and calibrate the rotary axes as well as the overall performance of the robot.

Chapter 3

Motion Tracking System for Augmentation of Camera based Non-Destructive Testing

In this chapter, we explore the methodology for the augmentation of Camera based NDT using motion tracking systems. This is an augmentation provided for handheld NDT systems that are to be used in-situ, eg. in aircraft hangers or launch vehicle assembly bays. Here, a robotic cell would not be feasible. Hence, the NDT System has to be carried by hand. In the case of Optical Non-Destructive Testing (NDT) methods such as Thermography and Laser Shearography these methods cover a small area of the structure being inspected and especially for curved surfaces there is no obvious method to relate a defect's pixel location in an image with the actual location of the defect on the panel. In order to cover a large structure, data must be taken at multiple viewpoints of the structure and it is often a challenge to locate a defect in a specific image with the actual location on the structure.

We can augment the basic manual process of NDT inspection by attaching motion tracking tags to the NDT instrument and the structure being inspected, while the inspection process is being monitored by background cameras. Here the background cameras will be used to track the inspection process being done and map it on to the digital model of the panel. A method is desired whereby the NDT system's position and orientation is tracked and this pose is used to determine the linkage between the defect image points and the physical object points. In this chapter, we explore a stereo vision based motion tracking system that locates the NDT system (here represented as a camera) in relation to the part being inspected and presents it in a user friendly 3D viewer. This would greatly aid the operator to relate defects found in the images to their actual location on the panel. Since

the pose data can be recorded and NDT data saved as well at every viewpoint, it can also be used for image stitching for making a mosaic of the multiple viewpoints NDT data on the virtual model of the panel.

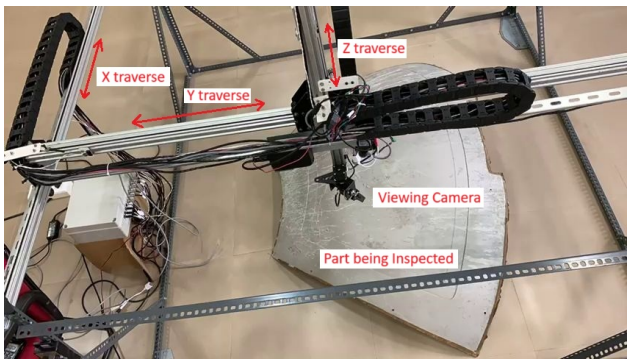
We discuss the development of prototype motion tracking systems using commercial grade USB cameras and April Tag fiducial markers. It is important to understand the accuracy of this system and how it differs from professional motion tracking systems. Hence, work has been carried out to characterize the errors in the system and deduce how one might proceed if a professional motion tracking system were to be designed.

3.1 Introduction

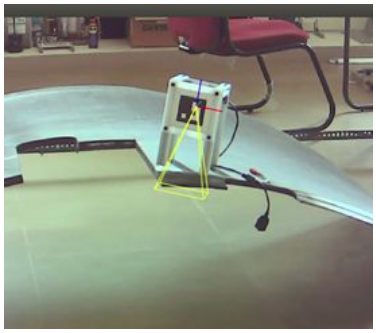
In the case of camera based NDT systems such as Thermography and Laser Shearography, the defect data is gathered by analyzing image data that is taken by a camera and this data is gathered from the panel according to how the camera's viewing frustum is positioned and oriented with reference to the panel. An example of pose estimation of the NDT system is shown in Figure 3.1b using a motion tracking system with April Tag marker. The pose of the camera is estimated and its viewing frustum is superimposed on the image as yellow lines. In the absence of such pose information, a defect may be detected using the image analysis techniques of the NDT instrument, but the operator would find it difficult to relate the position of the defect from the image position in pixels to the actual position on the structure. This creates problems when this defect requires further analysis or repairs. Furthermore it may be difficult to combine NDT image data from multiple viewpoints using image stitching techniques if we don't record pose data from the viewpoints at which this data was taken from.

Experts in the domain of NDT have long recognized that the detection of a defect using an NDT instrument does not by itself give an indication of where the defect actually is and have thus addressed this problem by augmenting the NDT or measuring instrument with some form of positioning information. The most common methods use robotic arms or cartesian robots (or other mechanism) to scan over a structure. This is an automated way to scan over a structure and also record the corresponding NDT information and the pose of the viewpoints from which the NDT data is being recorded. This method is also elaborated on in Chapters 2 and 4 which show a proof of concept for a prototype robot scanning the panel. The problem with this method is that a robot of this size is expensive and may not be portable. It can be used at the manufacturing stage where the structures or panels are brought to the robotic cell for inspection, but it may be inappropriate be used for

on-site inspection or maintenance activities for a fully assembled structure like an aircraft or launch vehicle. Figure 3.2a shows the schematics for a 5 axis cartesian robotic cell for inspection and Figure 3.1a shows a prototype cartesian robot cell developed for inspecting a panel.



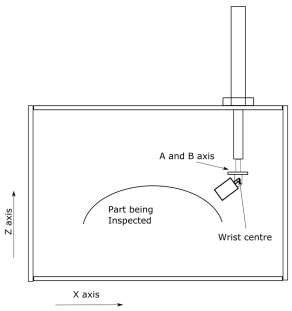
(a) Robotic system inspecting panel



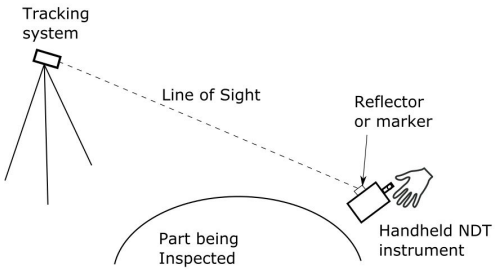
(b) Motion tracking and superposition of viewing frustum

Figure 3.1: Camera based NDT Augmentation methods

While robotic arms mounted on ground vehicles can be used for on-site inspections, it may be inconvenient or expensive to do so. An alternative to a Robotic System is a motion tracking based solution for Handheld NDT systems. Figure 3.2a shows the schematic for a Robotic solution while Figure 3.2b shows the schematic for a Motion tracking system.



(a) Robotic System for NDT



(b) Motion tracking system

Figure 3.2: Schematic diagram for Camera based NDT Augmentation methods

Motion tracking pose estimation systems have been proposed in US patent [18] but the exact implementation has not been disclosed there. Essentially it consists of either a laser tracker or a camera based system in the background and a reflector or fiducial markers are mounted on the object being tracked. The tracked object has to always be in the line of sight of the tracking system. The camera system can be a monocular, stereo, or multi camera sys-

tem. Laser trackers are available from companies like Faro and Leica but are prohibitively expensive for this application. Camera based tracking systems are available commercially from Vicon and Optitrack. Another stereo vision based tracking system called C-track system is included in Creaform's metrascan 3D laser scanning system (Figure 3.3). However the exact details of implementation are not disclosed in any of the commercial products. Furthermore these methods or products, to our knowledge, have never been applied for the purpose of camera based Non-Destructive Testing.

Figure 3.3: Creaform Metrascan system for 3D scanning

In this chapter we explore the techniques of camera based tracking for the purpose of tracking the motion of a handheld Non-Destructive Testing (camera based) instrument over a panel being inspected, the primary application of it being in-situ inspection. The accuracy and repeatability of the camera based tracking system is crucial to the success of the pose estimation of the NDT device. Hence this is explored at length. Section 3.2 investigates the effect of camera and lens hardware to see what kind of hardware should be used for motion tracking. Section 3.3 investigates the various algorithms and computer vision techniques to find which gives the best results for tracking purposes. Section 3.4 discusses the technique of combining pose information from motion tracking for the handheld NDT system and the panel and displaying it in a 3D viewer. Finally section 3.5 summarizes and concludes this chapter.

3.2 Camera and Lens Hardware

Understanding what kind of camera and lens system is most desirable for the motion tracking system is extremely important in order to maximize our performance. We need to perform standardized tests with metrics to determine what type of camera and lens system is most appropriate for such an application.

The testing we carried out is MTF (Modulation Transfer Function) testing (to estimate image sharpness), and noise testing (to estimate image noise). We carried out this testing on an ELP camera, Basler camera (model acA 1600) and PTZ camera with an 8mm ELP lens, a 25mm Pentax Lens and the PTZ (Pan-Tilt-zoom) camera respectively. The ELP camera can also be fitted with the 25mm Pentax lens using a C-CS mount adapter. Hence this is also used for testing. We then determine the standard deviation in pixels for April Tag detection and reach some conclusions as to what kind of hardware is appropriate. We also explored the need of setting depth of field and using a custom illumination system.

3.2.1 Modulation Transfer Function (MTF50) Testing

The Modulation Transfer function, or MTF is a measurement of the camera system's ability to transfer contrast at a particular resolution from the object to the image. Lens acts as a low pass filter, therefore at higher frequencies, the ability to distinguish between dark and light intensity will be diminished. [19] gives an overview of Modulation transfer function and its uses. Since the detection of our fiducial markers depends greatly on the contrast between high and low intensity pixels in the grayscale image, we expect that a sharper image (higher MTF) will be better for motion tracking.

We use the open source software **MTF Mapper** [20] for characterization of system MTF, which includes camera and lens MTF. This software comes with it's own test charts, that we need to print on paper, take snapshots of using the respective camera and lens, and then upload into the software for analysis. Figure 3.4 shows the MTF curve for a single black-white transition. The chart used has these kind of squares which are placed radially outward. Thus we can get the MTF for the whole field of view in meridional (radial) and sagittal (tangential) direction.

3.2.1.1 MTF test setup

The MTF test chart is printed on one A4 sized sheet and an A0 sized sheet. The large sized A0 sheet is required for wide field of view cameras and the small A4 sized sheet

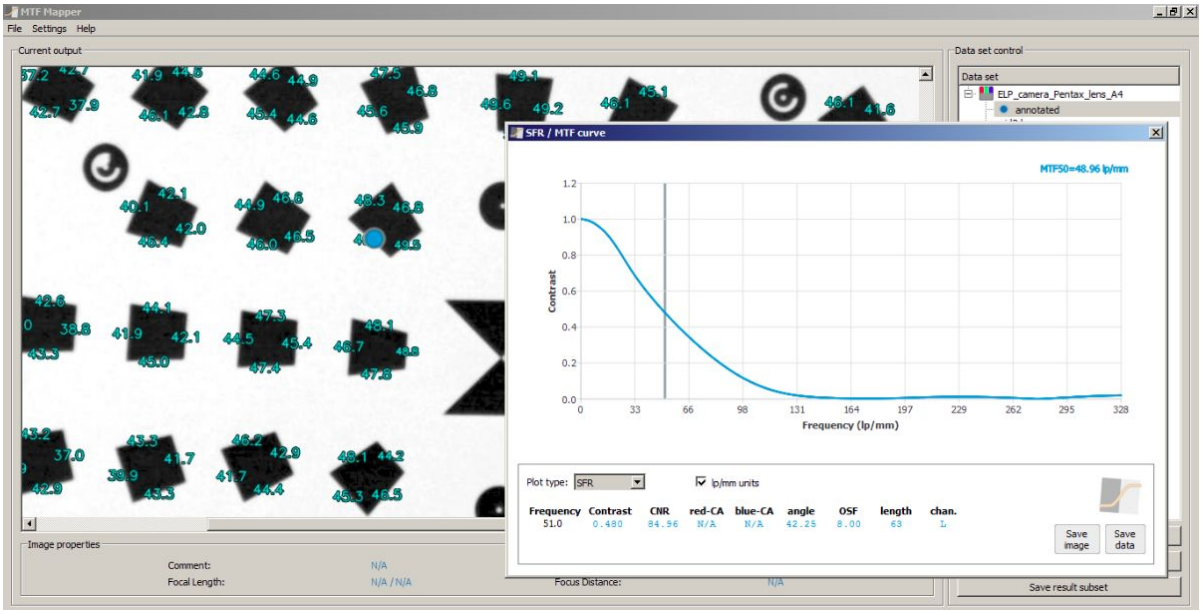


Figure 3.4: MTF testing using MTF Mapper software

is needed for narrow field of view cameras. This allows the cameras to be placed at at a reasonable distance so that the test chart fills the entire field of view. Figure 3.5 (a) shows the wide FOV 8mm ELP lens placed at a correct distance with the A0 sized chart, and with A4 sized chart (Figure 3.5 (b)) it would be too close and distorted by spherical aberration. Conversely the narrow FOV 25mm Pentax Lens would have to be placed very far away in order to view an A0 sized chart (Figure 3.5 (c)), but is at the right distance with an A4 sized chart (Figure 3.5 (d)). After setting it up, we take images, making sure that the camera sensor is as parallel to the chart as possible. MTF mapper has an in-built function to check this (see Figure 3.6) and it is recommended that the roll, pitch and yaw measured here are less than 5 degrees.

The images are taken for four cases:

1. ELP camera with 8 mm ELP lens

2. ELP camera with 25 mm Pentax lens

3. Basler camera acA1600 with 25 mm Pentax lens

4. PTZ camera with zoom value 1

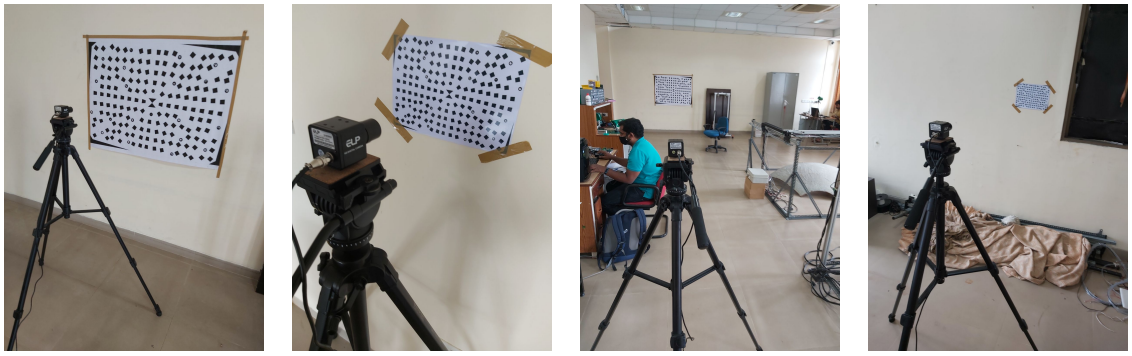


Figure 3.5: MTF chart sizes and setup: (a) Camera with 8 mm ELP lens viewing A0 chart, (b) Camera with 8 mm ELP lens viewing A4 chart, (c) Camera with 25 mm Pentax lens viewing A0 chart, (d) Camera with 25 mm Pentax lens viewing A4 chart

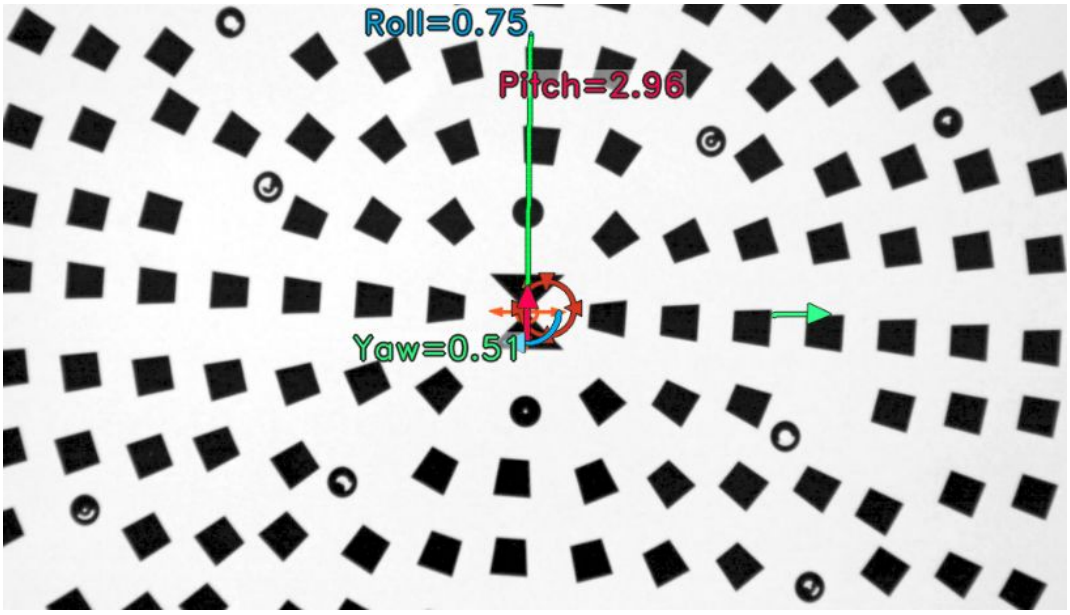
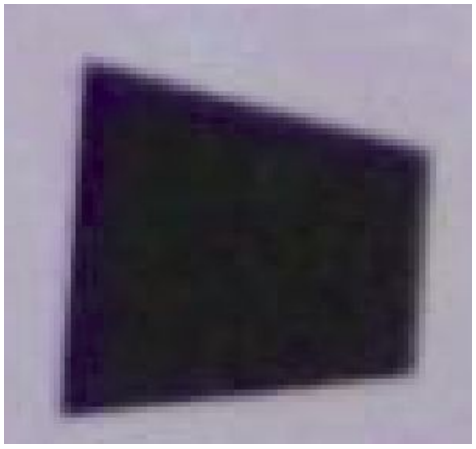


Figure 3.6: Chart Orientation using MTF Mapper software

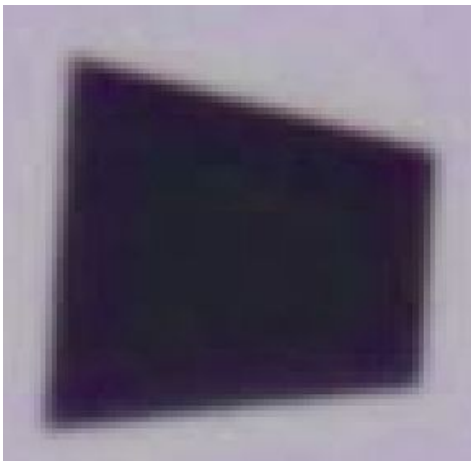
3.2.1.2 Results

An MTF curve is obtained for every black-to-white transition line in the MTF chart image. To keep things simple, we report the value of only MTF50 (Spatial frequency where MTF is 50 of the low (0) frequency MTF), for each line. MTF mapper calculates this and also plots the variation of this value of MTF over the entire chart (lens field of view).

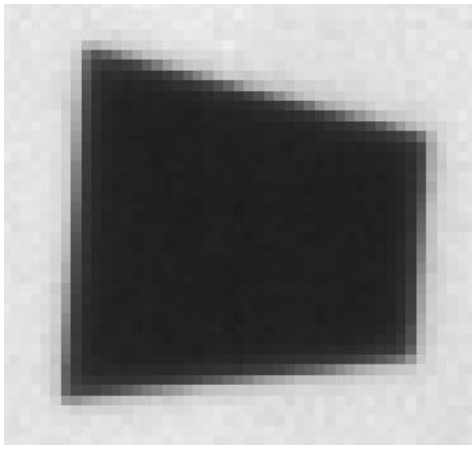
We observed that MTF mapper cannot detect the black to white transition for the PTZ camera (Figure 3.7d). This is because the image is compressed and of very low resolution while it is sent over the ethernet cable. Thus, even if the lens optics may be good, we cannot measure the MTF this way. It is also not suitable for tracking purpose.



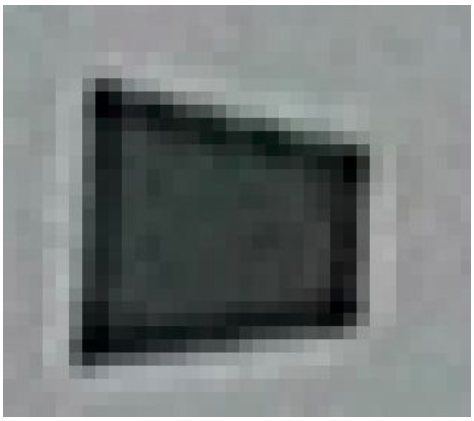
(a) Sample square, ELP camera with ELP lens



(b) Sample square, ELP camera with Pentax lens



(c) Sample square, Basler camera with Pentax lens



(d) Sample square, PTZ camera

Figure 3.7: Sample squares taken from images of MTF charts

The value of MTF50 is measured in Line pairs per millimetre (lp/mm). In order to get this value, we also need to take the size of the sensor and resolution into account to get the pixel size. This is done for the 2 different cameras (ELP and Basler). For the case of the ELP camera we also considered two different lenses (8mm ELP lens and 25 mm Pentax lens).

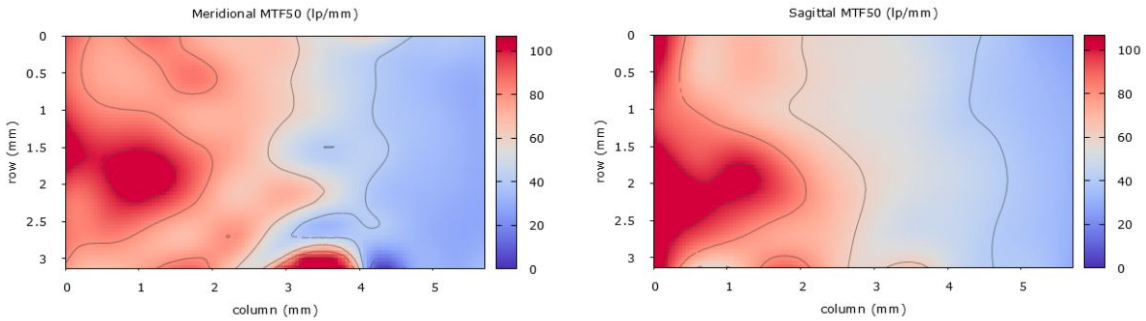


Figure 3.8: Variation in MTF using ELP camera and 8mm ELP lens

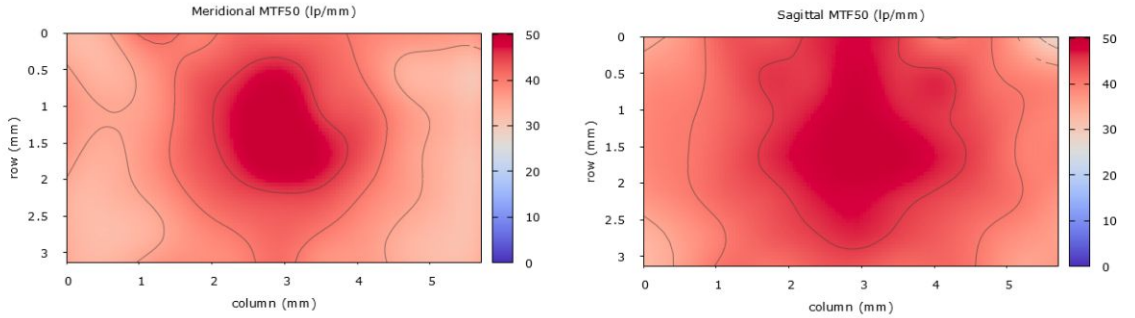


Figure 3.9: Variation in MTF using ELP camera and 25mm Pentax lens

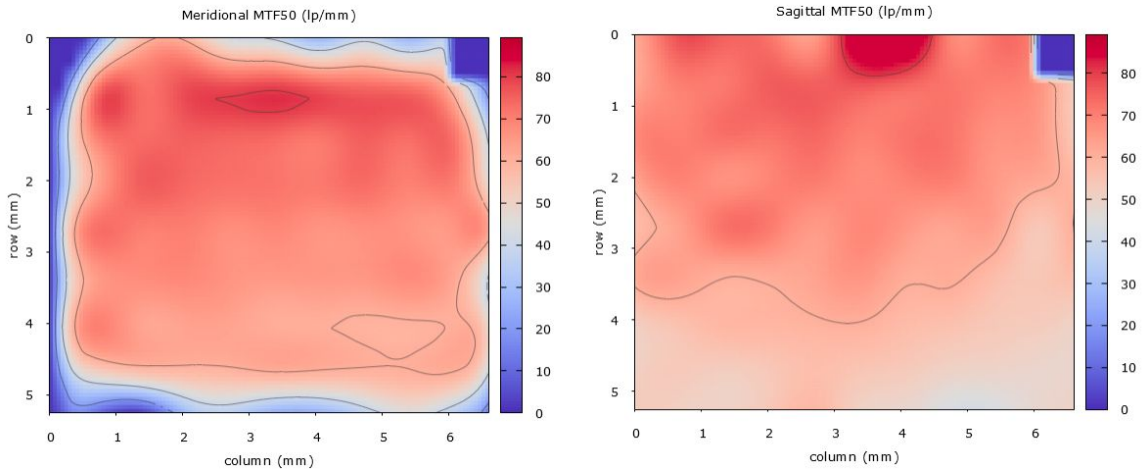


Figure 3.10: Variation in MTF using Basler camera and 25mm Pentax lens

We observe from Figure 3.8 and Figure 3.9 that while the MTF50 value in the centre of the image is comparable for both cases, but the variation in the ELP lens case is very large. This suggests that this lens is highly non-uniform. In fact we can see the variation in sharpness of the black to white transition line between the left parts and right parts of the image in Figure 3.11.

Also we observe that using the Basler camera with the Pentax Lens (Figure 3.10) gives similar results as when the same lens is used with the ELP camera (Figure 3.9). This suggests that MTF is primarily a property of the lens rather than that of the camera.

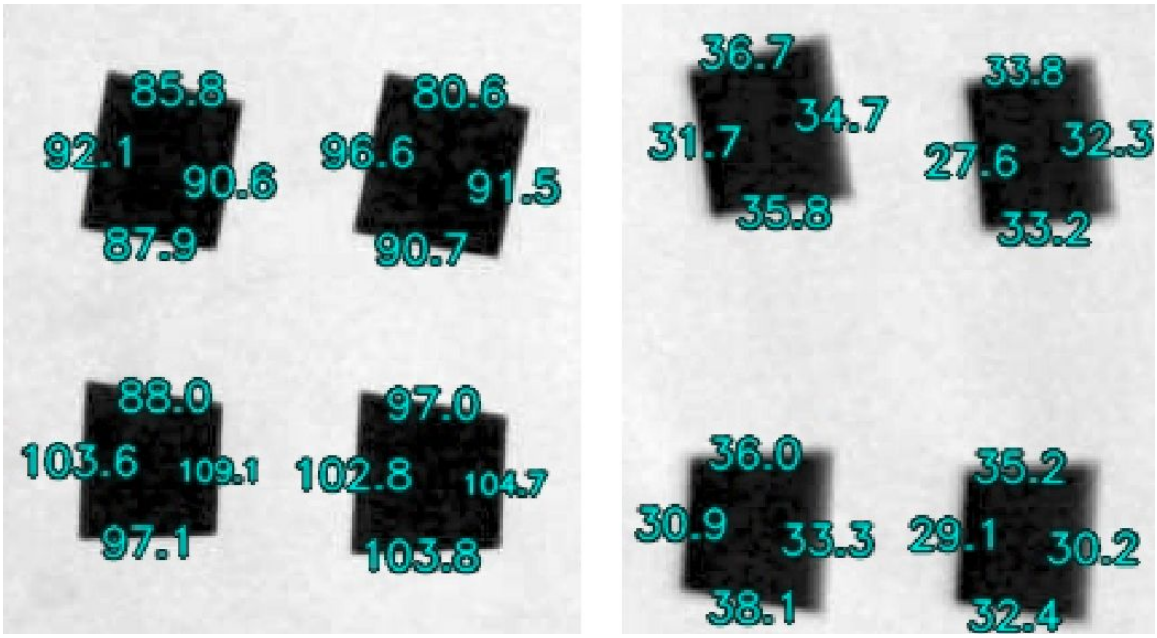


Figure 3.11: Image left and right portion using ELP lens (Note: the numbers denote MTF50 value of the black to white transition zone)

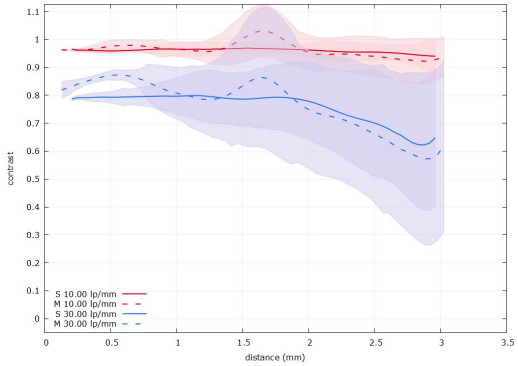
MTF mapper also has the facility to plot lens profiles. We observe from Figure 3.12 that the ELP lens has a very broad spread, while the Pentax lens has a very narrow spread for both the cameras using it. This reinforces our conclusion that the ELP lens is very non-uniform as compared to the Pentax lens.

The MTF50 values that we observe near the centre of the image are given in Table 3.1.

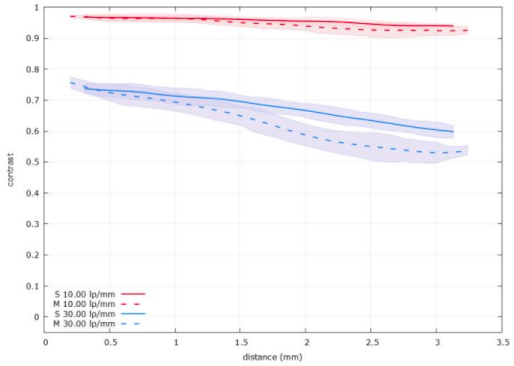
Camera	Lens	MTF50(lp/mm)
ELP camera	8mm ELP lens	56.2
ELP camera	25mm Pentax lens	48.8
Basler camera	25mm Pentax lens	64.1

Table 3.1: MTF50 values at centre of images

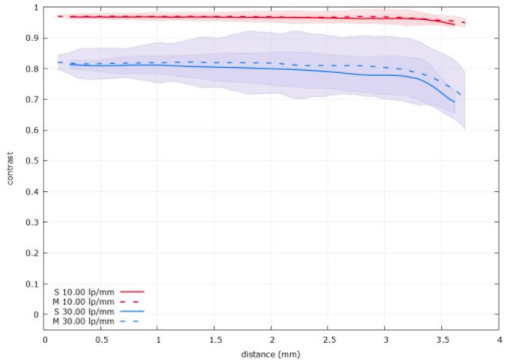
While all the values are similar, this does not tell us the true picture, the ELP lens is far less uniform compared to the Pentax lens and should not be used.



(a) Lens Profile, ELP camera with ELP lens



(b) Lens Profile, ELP camera with Pentax lens



(c) Lens Profile, Basler camera with Pentax lens

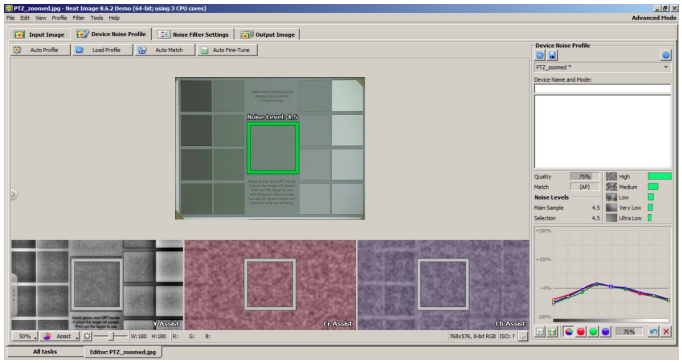
Figure 3.12: Lens Profiles from MTF charts

3.2.2 Noise Estimation

All digital cameras have noise. This noise is actually very complex, and is a mixture of intensity varying noise, shot noise and random noise. [21] and [22] give an overview of how to characterize and remove camera noise from an image. As the noise is responsible for disruption in our image, we expect it to have an impact on motion tracking performance. In this work, we do not develop any noise estimation or correction algorithm, instead we use the software Neat Image v8 (https://www.neatimage.net/) to carry out noise analysis of our camera. Here there is a calibration target which has different intensity patches which we can build a noise profile from. The setup is shown in Figure 3.13a and the Neat Image analysis window is shown in Figure 3.13b.

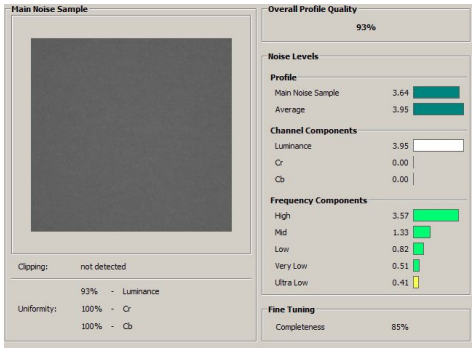
We will get a value of noise for every square (uniform intensity region), and thus get the variation in noise as per the intensity level of the camera. This will allow us to build the noise profile of the camera.

(a) Noise estimation setup

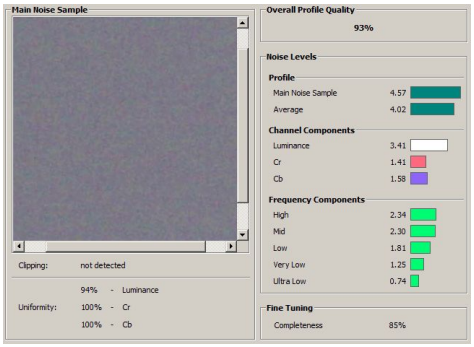


(b) Neat Image window

Figure 3.13: Noise Estimation using Neat Image



(a) Basler Noise Profile



(b) ELP Noise Profile

Figure 3.14: Noise Profiles

In Figure 3.14 we observe that in addition to an overall 'value' of this noise, there is also a qualitative difference in the frequency components of the noise. For the ELP camera, the noise is fairly evenly distributed across the frequencies, but the Basler camera has a preponderance of high frequency noise. This suggests that it is easier to de-noise by using a low pass filter. Image de-noising can be carried out by Neat Image software (by using the camera noise profile built), but since we cannot integrate this into our motion tracking pipeline, we leave it for a future work. Here Neat image is being used only for noise characterization and not correction.

Figure 3.15 shows the calibration target imaged through the Basler camera with 25mm lens. Each square denotes an area of uniform intensity and we have labelled them from 1 to 17. The Neat Image software is used to find the noise level for each labelled square.

We consider the following use cases and determine the noise value according to the

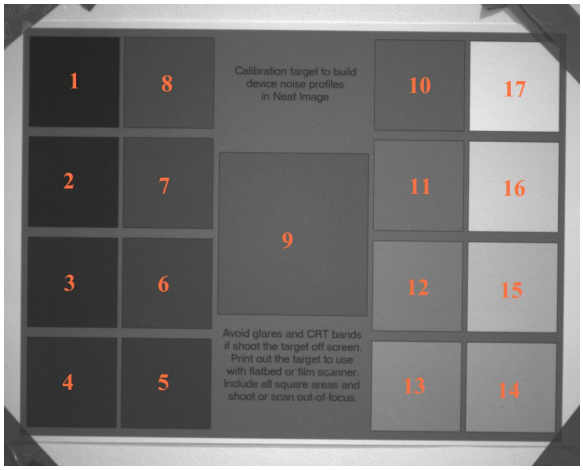


Figure 3.15: Neat Image calibration target image labelled

intensity level from 1 to 17 (1 being lowest intensity or darkest and 17 being the highest intensity or brightest). These 17 patches are corresponding to the calibration target used. Note that the Basler acA1600 being a scientific camera we can change the integration time of the image acquisition.

1. Basler Camera with 5ms integration time
2. Basler Camera with 7ms integration time
3. ELP Camera with ELP lens
4. ELP Camera with Pentax Lens
5. PTZ camera with zoom level 1
6. PTZ Camera with zoom level 4

Figure 3.16 displays the noise value as measured by the Neat Image software for each square patch (intensity level) as imaged by the above six camera setups.

We observe that the Basler camera has significantly lower noise level. Lower integration time results in lower intensity (and thus noise). Also, for Basler camera noise increases with intensity, for the others it decreases or remains nearly same. Zoom level and lens used doesn't have much effect.

The conclusion from this section is that a low noise scientific camera used at a low intensity setting will lead to ideal noise conditions and also can be de-noised more effectively.

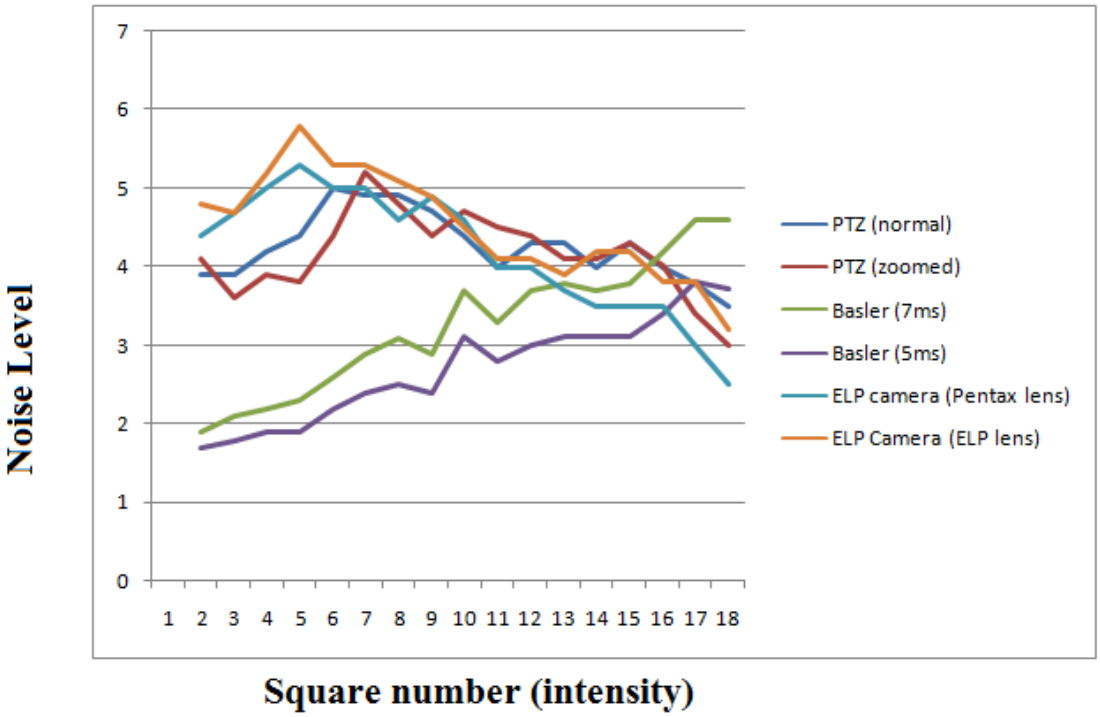


Figure 3.16: Noise Estimation Results

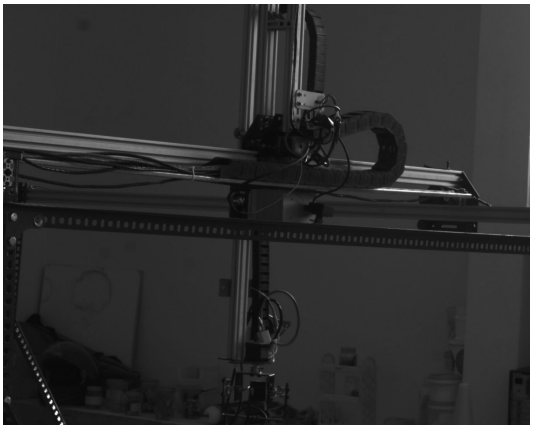
3.2.3 Depth of Field

In motion tracking, the object being tracked can vary significantly in its distance from the camera, as the user moves it about. This suggests that we need a large depth of field (distance range at which the focus is maintained) for the camera and lens system. Generally a machine vision system is used at a specific working distance (say at 300 mm) and the lens used is focused to that specific working distance. If the imaging system has a small depth of field, then objects placed closer or further from the working distance would quickly go out of focus. This may not be a problem for an application where the object is nearly at a constant distance (eg. scanning objects as they are moving on a conveyor belt), but for a tracking system, this would obviously be a problem. We also cannot manually refocus the lens as the tracked object is moved closer or farther from the camera. Hence the large depth of field is desired.

The most common way to increase depth of field is to reduce the aperture or iris of the lens. This results in less light falling on the camera, which makes the image darker. This can be compensated for by increasing the exposure time so that we can get an image of the same intensity. Note that the aperture is set manually and not changed during motion

tracking.

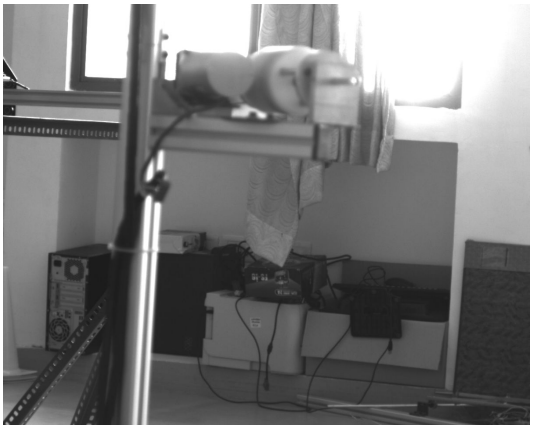
(a) Narrow aperture close distance image



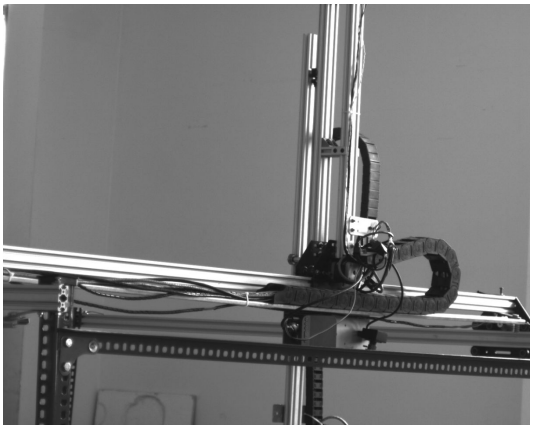
(b) Narrow aperture far distance image

Figure 3.17: Narrow aperture depth of field

Figure 3.17 shows images taken with a narrow aperture (f/8 setting), imaging objects located at distances of approximately 1m and 3m away from the camera. At aperture setting of f/8, the sensor receives 16 times less light than when it is at an aperture setting of f/1.4 . To compensate, the exposure time of the image is increased to 15ms. We can see that the close object and the distant object are both maintained in focus.



(a) Wide aperture close distance image



(b) Wide aperture far distance image

Figure 3.18: Wide aperture depth of field

Figure 3.18 shows images taken with a wide aperture (f/1.4 setting), imaging the same objects located at distances of approximately 1m and 3m away from the camera. At this aperture, the exposure time is set at 5ms. We can see that it is impossible to focus on both

close objects and distant objects at the same time. Figure 3.18b is in focus on the distant object, Figure 3.18a is distinctly out of focus for the nearby object.

Hence we have demonstrated that a narrow aperture size (and higher exposure time to compensate for lower light) achieves a high depth of field. The Pentax lens is highly suitable for this as it is easy to set the f stop number (aperture), while the ELP lens is mechanically unwieldy and has no reading for the aperture. It is also better to use a scientific camera whose exposure time and other settings can be controlled instead of relying on the automatic exposure of the ELP camera.

3.2.4 Illumination, Reflector and Filtering

So far, we have described a motion tracking system that tracks a black and white fiducial marker by means of natural lighting. It takes the image and a thresholding process is used to identify the fiducial marker.

However, the contrast between the black and white areas of the image could be dramatically increased by using a custom illumination, reflector and filtering system.

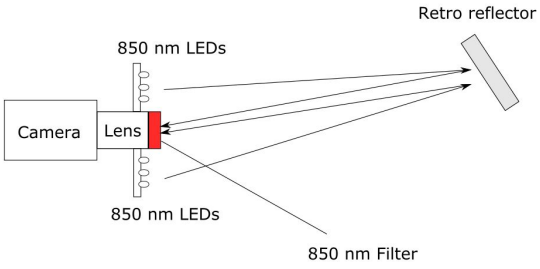


Figure 3.19: Illumination system

Figure 3.19 shows a schematic of a custom illumination system that uses a specific wavelength of light (850 nm IR LEDs in this case) for illumination. These LEDs will usually be mounted in a circular disc around the camera lens. We will use a retro reflective tape as our fiducial marker. This kind of tape is more reflective at certain wavelengths (typically in the IR region) and are retro reflective ie. they will reflect the light rays towards the source (directly into the camera). Then, a special band pass filter is fitted to the lens which will transmit only wavelengths in a narrow band around the 850nm region. Thus, only the illumination lighting reflection will be captured and the natural lighting will be filtered out. This makes our system independent of natural lighting which in any case is highly variable and should not be relied on.

We have not tested the custom illumination system proposed in Figure 3.19 so far. However we carried out a preliminary test using a red line laser (632nm) projected against a wall and imaged through a red colored glass filter. The imaging is done using Basler camera which is monochromatic (and does not capture color information).

(a) Red Laser line using colored glass filter **(b)** Red Laser line using no filter.

Figure 3.20: Effect of custom illumination and filtering

We observed that there is a dramatic improvement in using the filter (Figure 3.20a) as compared to not using the filter (Figure 3.20b) as the background lighting which is normally washing out the image is being filtered out. We can assume even better results would be obtained using a proper optical laser filter which can be in a very narrow wavelength range rather than just a colored glass filter.

Since such kind of illumination, retro reflective tape and filters are already standard in the security camera industry (they are used for better night vision), they are readily available to be used in a custom machine vision application such as this one as well.

3.2.5 April Tag detection and repeatability

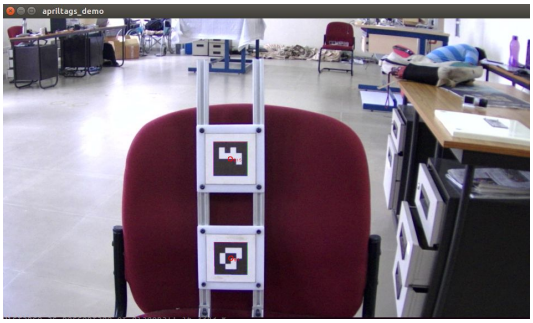
We conducted a test of fiducial marker (April Tag) detection using the different camera hardware. The April Tag algorithm takes an image as an input and returns the tags detected (4 corners and centre position in pixel values) as outputs. We take two different tags and calculate the distance between the centres in pixels. We then do this for 50 images captured (without moving the markers or the camera) and compute the standard deviation of the distance between centres in terms of pixels. This will give us an indication of the repeatability of the motion tracking system.

The following imaging setups are tested:

1. Basler Camera with 25mm Pentax Lens
2. ELP Camera with 8mm ELP lens
3. ELP Camera with 25mm Pentax Lens
4. PTZ camera with zoom level 1
5. PTZ Camera with zoom level 4

Figure 3.21 shows the setup with 2 April Tags being imaged using the Basler camera with Pentax Lens and ELP camera with ELP lens.

(a) April Tags detected using Basler Camera



(b) April Tags detected using ELP camera and ELP lens

Figure 3.21: April Tag detection

Since a wide variety of focal lengths and resolutions are involved, we do the following to normalize the results. We keep the April Tags at a position where the distance between the centres is at approximately 16% of the diagonal resolution in pixels. This will ensure that the angle subtended by the tags is approximately the same for the different imaging setups. Then, since the cameras have different resolutions, we find the mean pixel distance between the centres and find a multiplying factor to normalize it to 300 pixels. Then we apply the same multiplying factor to the standard deviation to get a normalized standard deviation in pixels.

The results for the normalized standard deviation are as listed in Table 3.2.

We observe that the best value is observed for the case of Basler Camera with Pentax lens.

3.2.6 Conclusions

From our testing, we conclude that the following are necessary for good motion tracking hardware

Camera and Lens	Resolution	Distance (as %of diagonal)	Mean Distance (pixels)	Standard Deviation (pixels)	Std deviation normalized for mean distance 300px
ELP camera ELP lens	1920 x 1080	16.3317	359.779	0.01633	0.0136
ELP camera Pentax lens	1920 x 1080	16.298	359.071	0.0234	0.0195
Basler camera Pentax lens	1280 x 1024	16.1075	264.032	0.00758	0.0086
PTZ camera (Zoom 1)	768 x 526	16.3681	157.13	0.00745	0.0142
PTZ camera (Zoom 4)	768 x 526	16.3493	156.948	0.00692	0.0132

Table 3.2: Normalized standard deviation of distance between centres of the two April Tags

- A scientific camera which provides high resolution, low noise and ability to set exposure time and other parameters
- A good machine vision lens with uniform high MTF, aperture setting ,low distortion and high depth of field
- Illumination system consisting of LEDs, retro reflectors for fiducial markers and optical filter
- Communication system must give full frames without any image or video compression

We note that the above are all characteristics of the Creaform Metrascan system (Figure 3.3) and thus this is what we should aim for using in our computer vision hardware. The other systems explore (ELP camera and Lens, PTZ camera) fail on one or more of these parameters. However since they are significantly cheaper, they can be used during prototyping phase of the system development.

3.3 Image processing and Pose Estimation algorithms

In this section, we will explore the image processing and pose estimation algorithms pipeline which takes images as inputs and determines pose of the tracked object as the output. Two primary methods are explored here - monocular (using one camera) and stereo (using two cameras). We show the results obtained for repeatibility and accuracy of the pose estimation.

There is a lot of literature available in the general domain of Computer vision based tracking. [23] and [24] among others, generally explore the methodology of marker based tracking systems. [25] shows the accuracy results of a commercial Vicon tracking system by using robot motion for comparison. [26] investigates the merits of using a stereo based motion tracking system over a monocular system.

3.3.1 April Tag detection

In order to do motion tracking, we must make use of features in the image that we can derive some information from. Natural features include shapes that can be recognized (typically by machine learning algorithms) such as people, trees, cars, faces and so on. These are good for identification (eg. face recognition, gesture recognition, etc.) but their accuracy for pose estimation is poor. Artificial features, also called fiducial markers are objects deliberately introduced into the image scene for the purpose of pose estimation. These fiducial markers are attached to the object being tracked and there are special image processing algorithms for their detection.

The entire range of fiducial markers proposed and tested in existing literature are summarized in Figure 3.22. [24] gives an overview of the various tags used. In this work, we do not test all kinds of tags, rather we carry out all experiments only using April Tags. April Tags [27] are square in shape, represent a binary number corresponding to tag id and family and the image processing algorithms to detect them are open source and readily available. We note that AR tags like April Tags are primarily used for Augmented Reality applications rather than pose estimation, and the commercial systems like Creaform and Vicon use simple circular fiducials. Nevertheless the simplicity of April Tag detection makes them worthwhile to explore for the purpose of pose estimation.

The April Tag tracking pipeline is given in Figure 3.23 . The input to this image processing algorithm is a grayscale image (if the image is taken by an RGB camera, we convert it to grayscale). We also specify the tag family used (in this case, tag family 16h5) The output

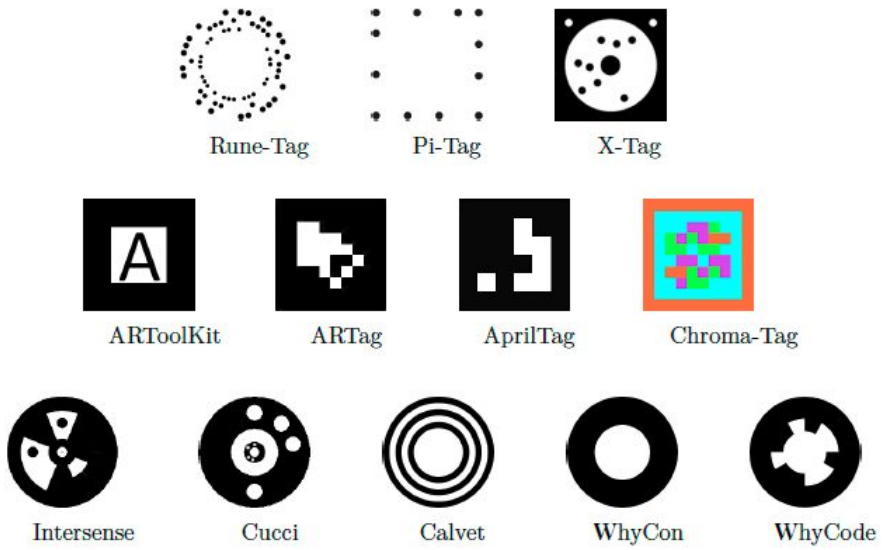


Figure 3.22: Fiducial Marker Tags

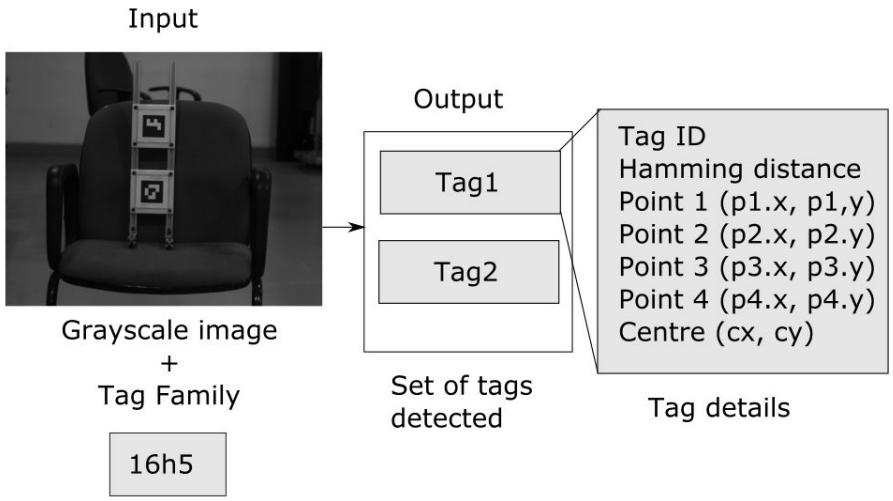


Figure 3.23: April Tag Detection

is a set of tags, and each tag is a data structure which holds the tag ID, hamming distance, four corner points and centre point (in pixel values). The Hamming distance is a measure of how close the detected tag's binary coding is to its ideal value. Here, we reject any tag with Hamming distance > 0 as it would usually be a false detection. The four points are identified in the sequence of p1 = upper left, p2 = upper right, p3 = lower right, p4 = lower left corners of the tag. The centre point is merely the average of all these four points. Once these tags are identified using the April Tag detection algorithm, the image processing sec-

tion is over, and the tag data is fed through the pose estimation pipelines described in the next sections.

3.3.2 Monocular Pose Estimation

As the name implies, Monocular Pose estimation denotes pose estimation using data from a single camera. This requires us to know the camera matrix, ie. we should carry out a camera calibration first (this is described in more detail in section 3.3.3.1). We mostly use the method of [28] for monocular camera calibration. Then, we need to know the object (or world) coordinates and image coordinates of the object being tracked. If for example, we assume that a single tag is being tracked and its object coordinate system has origin at the centre of the tag and axes along the tag's rectangular directions, then the object coordinates of the corners would be (-s,s,0), (s,s,0), (s,-s,0), (-s,-s,0) where s is half the size of the tag in metres. The image coordinates are determined by the April Tag algorithm as described in the previous section.

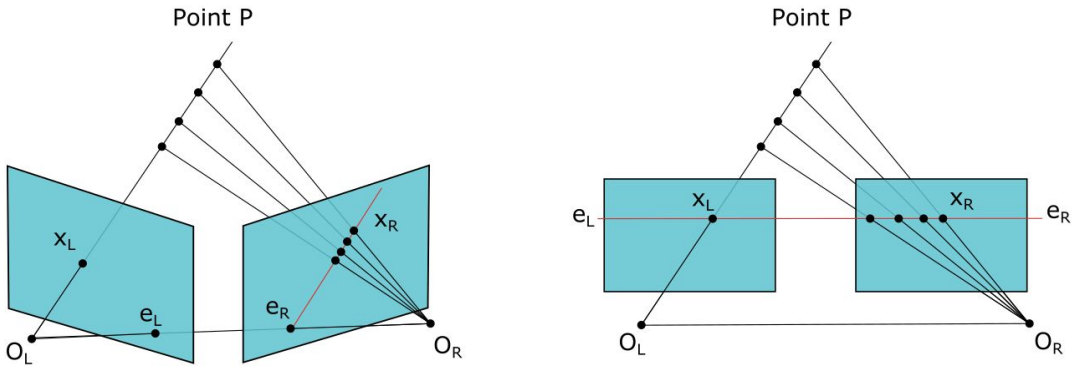
Then, we will need to solve the camera equation for the Rotation matrix $\mathbf{R}$ and Translation vector $\mathbf{t}$. This is seen in Equation 3.1, where $\mathbf{K}$ is the 3x3 camera calibration matrix, (u,v) are the image coordinates in pixels and (x,y,z) are the object coordinates in metres. We need to solve for $\mathbf{R}$ and $\mathbf{t}$ given the image and object coordinates for a set of points.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{3.1}$$

This problem is generically known as Perspective and Point (PnP) problem or the 3D to 2D correspondence problem. At least 3 points are needed to get the pose (rotation and translation) uniquely. If more than 3 points are available, a least squares solution is found. We use OPENCV which already has the built in function called 'solvePnP' to carry out the pose estimation. The effects of different sets of markers used and the algorithms used to solve these PnP equations are shown in the results section.

3.3.3 Stereo Pose Estimation

A stereo camera is a set of two cameras separated by some distance, called the baseline. Epipolar geometry is a term which covers the geometry of stereo vision, which relates the camera properties, points in 3D and points captured in the camera images.



(a) Epipolar Geometry (general case) **(b)** Epipolar Geometry (special case)

Figure 3.24: Epipolar Geometry

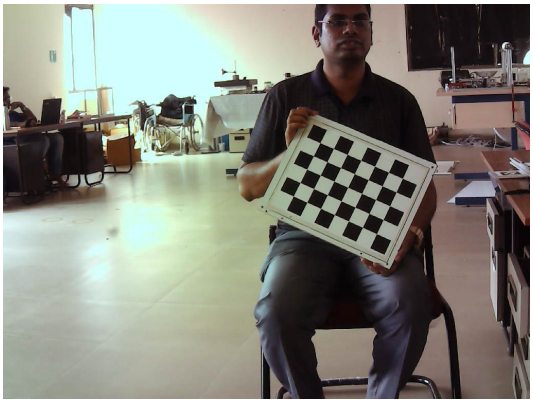
Figure 3.24a shows the general schematic for a stereo vision system. The centres of the cameras are denoted by O_L and O_R and the line joining them is called the baseline. The camera centres are imaged in the other camera's sensor frame at points e_L and e_R which are called epipoles . A point P in 3D space is imaged at point x_L in the left image and x_R in the right image. As the point P is moved along the ray from O_L to x_L, the image of the point in the left camera frame remains the same (this is why it is not possible to determine the depth of a single point from just one view point), but the image of the point in the right camera frame will move along a line called the epipolar line. Epipolar lines have the property that they intersect the baseline at the corresponding epipole (e_R in this case).

Figure 3.24b shows a special case of epipolar geometry which has some interesting properties. Here, the cameras are identical and their sensors are in the same plane separated only by a horizontal distance (in case of horizontal stereo) or a vertical distance (in case of vertical stereo). In this case the epipoles lie at infinity and the epipolar lines are parallel to one of the axes of the camera frame (x-axis in the case of the horizontal stereo). Thus, corresponding points in the two images will have the same y-coordinate, which simplifies calculations a great deal. Although this is a special case, this is usually the configuration in which stereo cameras are normally used. In the remainder of this chapter, we assume the stereo camera uses identical cameras placed in a horizontal stereo configuration.

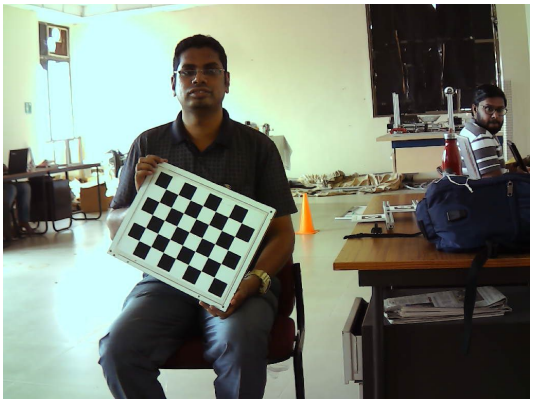
3.3.3.1 Stereo Calibration

Before we can run any computer vision estimation technique, it is necessary to determine the geometric parameters of the cameras and their positioning. This is known as calibration. Usually a printed pattern of known dimensions, such as a checkerboard is used for the

calibration, as seen in Figure 3.25. In the case of stereo calibration, the pattern is recorded by both the left camera and the right camera. Thus a dataset is created with the calibration checkerboard being held at different distances and angles for each image pair.



(a) Left image of chessboard pattern



(b) Right image of chessboard pattern

Figure 3.25: Stereo Calibration data

Stereo calibration in general is quite complicated [29] and an overview is provided in Figure 3.26. First, the camera intrinsics (Camera matrix and distortion coefficients) are individually estimated for the left and right camera. Then, stereo calibration is done to determine $\mathbf{R}$,$\mathbf{t}$,$\mathbf{E}$,$\mathbf{F}$ matrices. $\mathbf{R}$ and $\mathbf{t}$ are the rotation and translation of the right camera with respect to the left camera (so for an ideal horizontal stereo system, $\mathbf{R}$ will be identity matrix and $\mathbf{t}$ will be $\begin{bmatrix} B & 0 & 0 \end{bmatrix}^T$ where B is the baseline distance). $\mathbf{E}$ is Essential Matrix and $\mathbf{F}$ fundamental matrix which are related to $\mathbf{R}$, $\mathbf{t}$ and the camera matrices.

The values of $\mathbf{R}$ and $\mathbf{t}$ are used to determine the projection matrices $\mathbf{P_L}$ and $\mathbf{P_R}$, which are the projection matrices used to project points from the 3D space to the left and right images respectively. Stereo rectification parameters are also estimated such as Stereo maps $\mathbf{M_{1,L}}$, $\mathbf{M_{2,L}}$ and $\mathbf{M_{2,L}}$,$\mathbf{M_{2,R}}$ which are used for rectifying or undistorting the image as described in the next section. Note that since the individual camera calibration is also done here, there is no need to do monocular camera calibration separately (The monocular pose estimation uses only the left camera as input).

3.3.3.2 Stereo Rectification

In this step, we will rectify the raw images that are captured by the stereo camera so that the corresponding points in the two images have the same y-coordinate (ie. they are on the same epipolar line). The stereo mapping parameters $\mathbf{M_{1,L}}$, $\mathbf{M_{2,L}}$ and $\mathbf{M_{2,L}}$,$\mathbf{M_{2,R}}$ generated during the calibration process are used here. Figure 3.27 describes the rectification process.

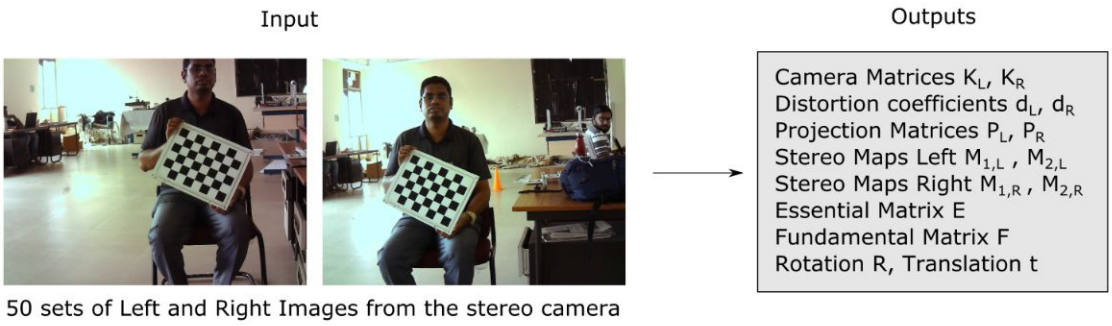


Figure 3.26: Stereo Calibration Process

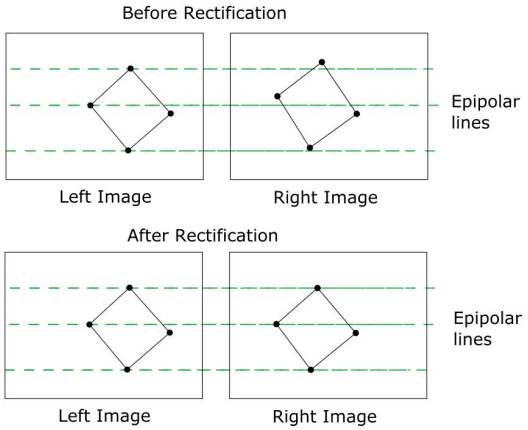


Figure 3.27: Stereo Rectification Process

Figure 3.28 shows the images captured by the stereo camera after rectification. Note that the image may be smaller than the original frame size after the rectification. In that case, we fill the remaining space with black color as can be noted in the right camera image in this figure.

3.3.3.3 Stereo Triangulation

After the rectification process we will run the images through the April Tag detector. Then we will have a set of point correspondences in which the y-coordinate is the same but the x-coordinate is different for the Left and Right images. The difference between the x-coordinates of the left and right image is known as **disparity**, and the disparity values over a number of points in the image constitute a disparity map.

We can get the depth, or z coordinate from the disparity map by a process called Stereo Triangulation.

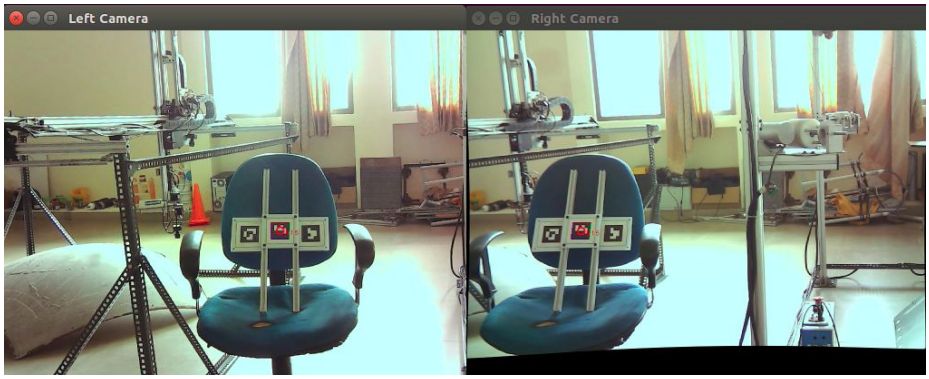


Figure 3.28: Camera images after Stereo Rectification

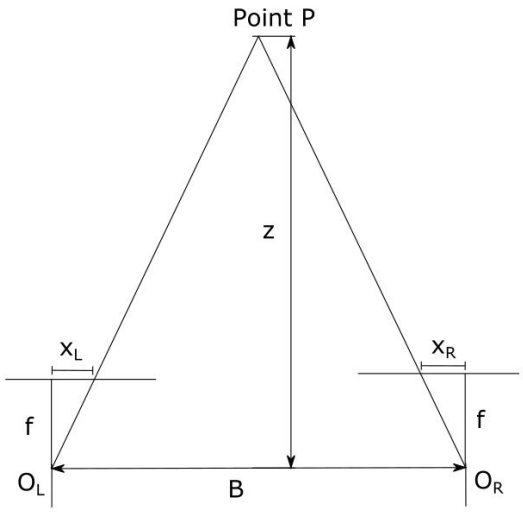


Figure 3.29: Stereo Triangulation

Figure 3.29 shows a basic scheme of stereo triangulation. O_L and O_R are the camera centres, x_L and x_R are the recorded distances on the camera sensor frames based on pixel values, f is the focal distance, B is the baseline (distance between the two cameras) and z is the distance of the point from the baseline. The disparity is $(x_L - x_R)$ and the depth z is given by Equation 3.2.

$$x_L - x_R = Bf/z \tag{3.2}$$

In OPENCV this is done in an automated manner by using the function cvTriangulate-Points which takes the projection matrices P_L and P_R into account along with the pixel positions in the left and right images. The output is the position (x,y,z) of the point in the camera coordinate frame.

3.3.3.4 Kabsch algorithm for pose estimation

From the stereo triangulation process, we have the position vectors of the points detected in the camera coordinate frame. We also know the position vectors of these points in the body frame of reference (ie. the frame of reference fixed to the tag). We thus need to find the Rotation matrix $\mathbf{R}$ and Translation vector $\mathbf{t}$ which constitutes the pose. This is described by Equation 3.3

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_C = \mathbf{R} \begin{bmatrix} x \\ y \\ z \end{bmatrix}_B + \mathbf{t} \tag{3.3}$$

This problem is known as the absolute orientation problem or 3D to 3D correspondence problem. When both $\begin{bmatrix} x, y, z \end{bmatrix}_C$ and $\begin{bmatrix} x, y, z \end{bmatrix}_B$ are known, there is an algorithm known as Kabsch Algorithm [30] which is proved to be optimal and efficient. We reproduce the algorithm here considering its great importance. Note that in the absolute orientation problem, the problem is usually stated as Equation 3.4 (general case of absolute orientation problem).

$$\mathbf{y_i} = \lambda \mathbf{R} \mathbf{x_i} + \mathbf{t} \tag{3.4}$$

Here we use the points in camera frame as y_i, points in world reference frame as x_i, and $\lambda = 1$ (so this is actually a restricted case of the absolute orientation problem which also includes scaling factor that we don't consider here).

Algorithm 3.1: Kabsch Algorithm for Absolute Orientation Problem

Input: Body points $\mathbf{x_i}$, Transformed points $\mathbf{y_i}$

Output: Rotation matrix $\mathbf{R}$, Translation vector $\mathbf{t}$

// Calculating mean of x_i, y_i

$\mathbf{x_0} \leftarrow (\sum_{i=1}^{n} \mathbf{x_i})/n$

$\mathbf{y_0} \leftarrow (\sum_{i=1}^{n} \mathbf{y_i})/n$

// Calculating H (3x3 matrix)

$\mathbf{H} \leftarrow \sum_{i=1}^{n} (\mathbf{y_i} - \mathbf{y_0})(\mathbf{x_i} - \mathbf{x_0})^T$

// Taking SVD

$\mathbf{U}, \mathbf{D}, \mathbf{V}^T = computeSVD(\mathbf{H})$

// Calculating R and t

$\mathbf{R} \leftarrow \mathbf{U}\mathbf{V}^\mathbf{T}$

$\mathbf{t} \leftarrow \mathbf{y_0} - \mathbf{R}\mathbf{x_0}$

return $\mathbf{R}$,$\mathbf{t}$

Algorithm 3.1 details the Kabsch algorithm for solving the absolute orientation problem. The computation of SVD (Singular Value Decomposition) of a 3x3 matrix into a orthogonal matrices $\mathbf{U}, \mathbf{V}$ and diagonal matrix $\mathbf{D}$ is well known ([30]) and not described here. The output of the algorithm is the Rotation matrix $\mathbf{R}$ and Translation vector $\mathbf{t}$.

The complete stereo vision processing pipeline is thus detailed in Figure 3.30. Note that we assume the stereo camera parameters are already known from the stereo calibration process (Figure 3.26) which only needs to be done once.

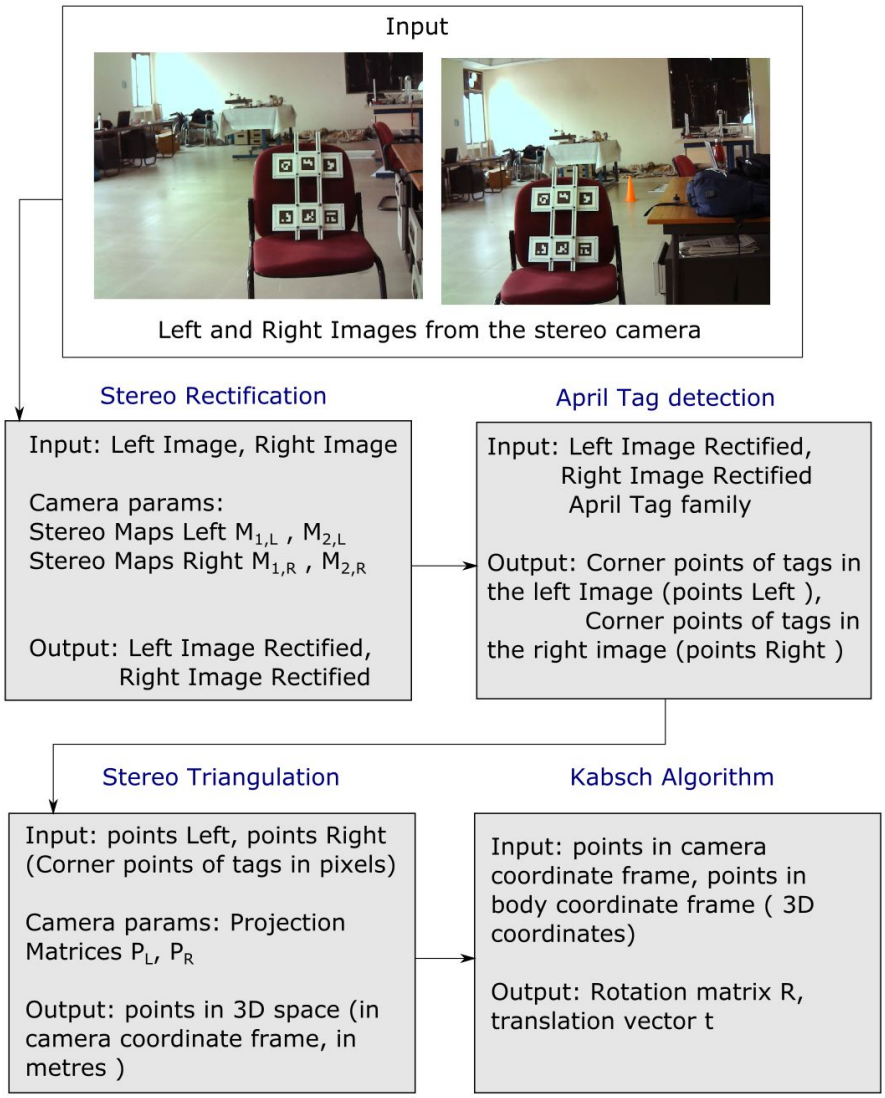


Figure 3.30: Stereo Pose Estimation Pipeline

3.3.4 Results

The results of motion tracking pose estimation for the various cases are tabulated here and conclusions are drawn.

3.3.4.1 Data acquisition and recording datasets

In this section we describe the process of data acquisition and storing data-sets for further analysis.

Preparation of targets

The tracking target to be imaged is composed of one or more April Tags. The tags are printed on paper, glued to acrylic plates and these plates are joined together using some aluminium channels and fasteners. The following configurations of April Tag targets were used as shown in Figure 3.31 :

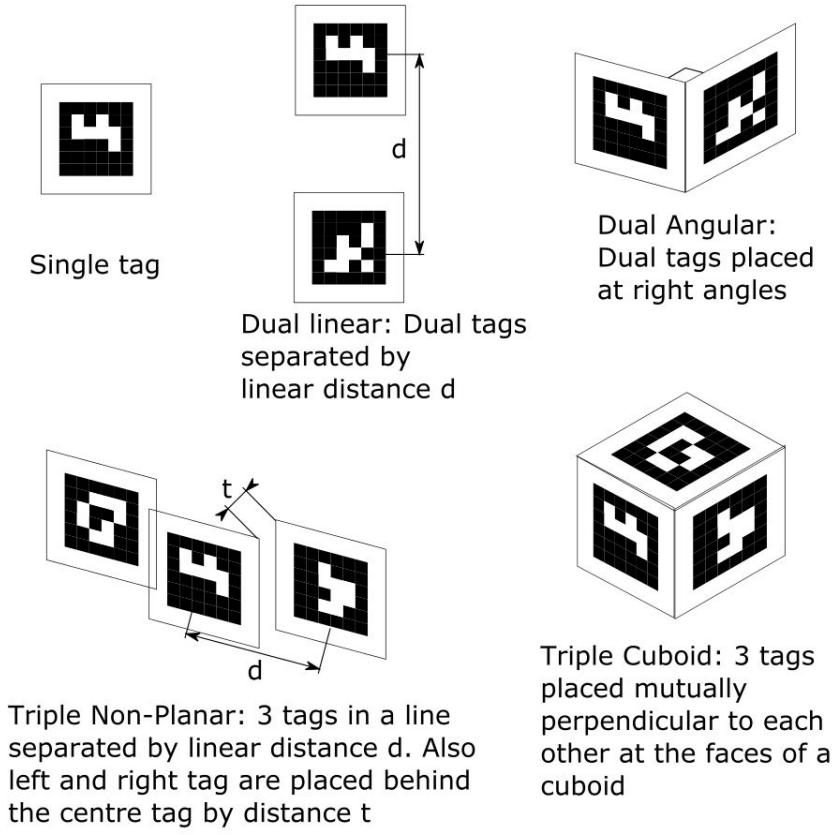


Figure 3.31: April Tag configurations for tracking

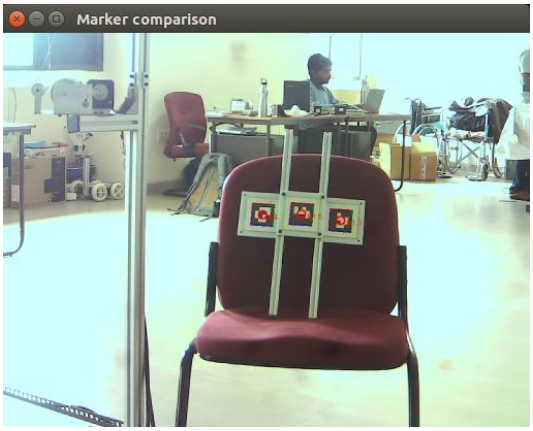
Five setups have been tested for the April Tag configuration:

- Single tag
- Dual Linear: Two tags separated by a distance
- Dual Angular: Two tags at right angles
- Triple Non Planar: Three tags along a line, the central tag is displaced above the other two

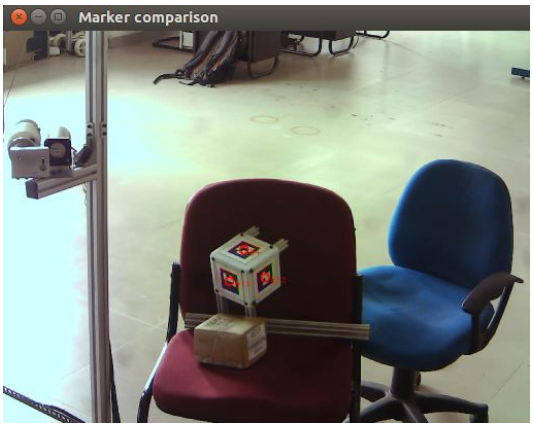
- Triple cuboidal: Three tags which are orthogonal to each other.

These motion tracking targets are used for pose estimation. For accuracy estimation the distance between two single tag systems are used. Note that when multiple markers make up a tracking target, all the points from all the tags are used at the same time for pose estimation using object points and image points. For example for the case of a single tag, the object points are at (-s,s,0) ,(s,s,0) ,(s,-s,0) ,(-s,-s,0) assuming s is half the tag size. For the case of the triple non planar, the object points are (-s,s,0) ,(s,s,0) ,(s,-s,0) ,(-s,-s,0) , (-s+d, s , -t) ,(s+d, s , -t) ,(s+d,-s , -t) ,(-s+d,-s , -t) , (-s-d, s , -t) ,(s-d, s , -t) ,(s-d,-s , -t) ,(-s-d,-s , -t), where t and d are defined as in Figure 3.31. The image points are the pixel locations obtained from April Tag detection.

As the number of tags which can be deployed in our case is limited, we did not investigate how the accuracy can be increased by increasing the number of tags.



(a) Triple NonPlanar target



(b) Triple Cuboid target

Figure 3.32: Motion tracking targets April Tags

Figure 3.32 shows the configuration of two targets composed of multiple April tags as they are imaged by the camera.

Stereo Camera setup

Identical ELP color cameras imaging at a resolution of 1024x768 pixels are used. The zoom level of the ELP lens are set to the same and the lenses are focused by using the focus ring. The two cameras are bolted to a bar 350 mm apart and placed pointing outward in the same direction.

Image acquisition and storage

The stereo camera is set up and pointed at the target so that the target is wholly visible in both camera's field of view after stereo rectification. Then, 50 pairs of images of the target are taken one after another without moving the cameras or the targets in any way. This takes a few seconds. The 50 image pairs are stored in a directory. We do this for various configurations of April tags and also distances to the camera. Once all the data is saved, we will run the monocular and stereo pose estimation pipeline on the stored dataset (so here we are not doing real time tracking). Note that for the monocular vision algorithms we use only the data from the left camera and the right camera is ignored.

3.3.4.2 Repeatability of Pose Estimation

The repeatability of pose estimation is estimated by using the standard deviation over the translation vector (x,y,z components) of the pose estimated over 50 images as saved in the dataset. We do this for the different types of targets and also for the monocular and stereo case. The target is kept at approximately the same distance of 2m from the camera. Results are given in Table 3.3 :

Target type	Tracking system	Mean Translation vector (x,y,z) in mm	Standard deviation (x,y,z) in mm
Single Tag	Monocular	(223.80, -54.79, 1968.14)	(0.071, 0.019, 0.658)
Single Tag	Stereo	(596.17, 11.53, 1881.09)	(0.013, 0.012, 0.074)
Dual Linear	Monocular	(221.85, -40.47, 1949.06)	(0.085, 0.025, 0.734)
Dual Linear	Stereo	(596.02, 12.05, 1880.85)	(0.015, 0.016, 0.068)
Dual Angular	Monocular	(-213.71, 105.08, -1544.86)	(0.044, 0.022, 0.269)
Dual Angular	Stereo	(-246.47, -29.95, 1521.79)	(0.012, 0.012, 0.052)
Triple Non-Planar	Monocular	(-223.33, 54.32, -1964.87)	(0.016, 0.010, 0.124)
Triple Non-Planar	Stereo	(596.32, 12.18, 1879.86)	(0.012, 0.040, 0.052)
Triple Cuboid	Monocular	(-95.36, 130.95, 3048.27)	(0.021, 0.040, 0.817)
Triple Cuboid	Stereo	(-241.99, -34.26, 1502.96)	(0.036, 0.022, 0.076)

Table 3.3: Repeatability Estimation

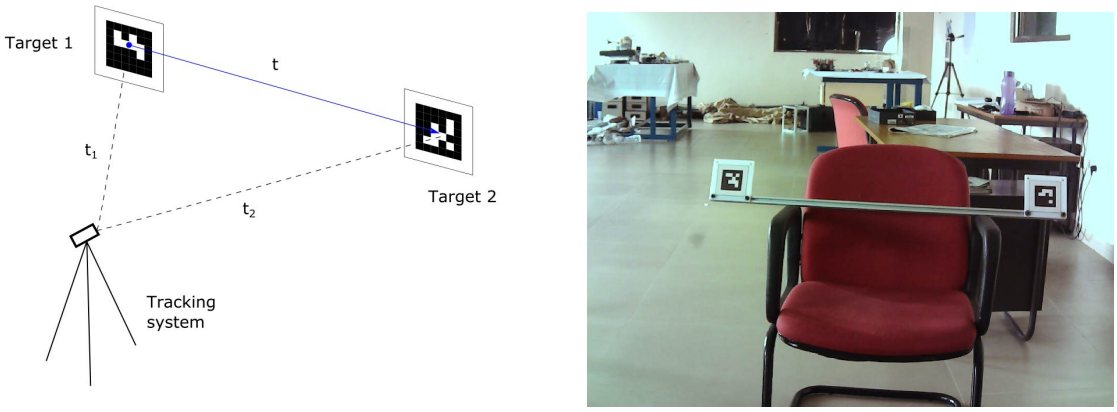
We can observe that the repeatability of pose estimation is weakest in the Z direction

(facing away from the camera), but the stereo repeatability is significantly stronger than the monocular one in every case, sometimes by nearly 10 times. This is consistent with what is expected in the literature (eg. [26]). Also, the more number of points (tags) used in the target, the better the repeatability becomes.

3.3.4.3 Accuracy of Pose Estimation

The accuracy of pose estimation is harder to determine than the repeatability. In order to determine accuracy we need to determine the real pose of the camera relative to the tracked object but we don't have any setup or instrument to do that. Hence, we measure accuracy in an indirect manner. We set up two tags separated by a known distance, and then take the difference of the translation vector as measured by pose estimation method. The norm of the translation vector (averaged over the dataset of 50 images) gives us the euclidean distance between the tags, and we can compare it to the actual distance between the tags to get the accuracy.

Figure 3.33 shows the schematic as to how the accuracy is estimated. If $\mathbf{t_1}$ is the translation vector from camera to tag1, and $\mathbf{t_2}$ is the translation vector from camera to tag2, then the translation vector from tag1 to tag2 is given by $\mathbf{t} = \mathbf{t_2} - \mathbf{t_1}$. The accuracy is determined as the percentage error of the estimated distance over the real distance.



(a) Accuracy Estimation scheme **(b)** Image acquired for Accuracy Estimation

Figure 3.33: Accuracy Estimation

The results of accuracy estimation are given in Table 3.4.

We can observe that the error percentage (accuracy) is more consistent in the stereo vision case while it varies from large to small in the case of monocular, especially for larger distance between tags.

Distance between tags(mm)	Tracking system	Mean Translation vector (x,y,z) in mm	Euclidean distance (mm)	Error %
240	Stereo	(-16.82, 229.63, -70.80)	249.74	4.06 %
240	Monocular	(-249.09, 1.21, -17.89)	244.48	1.87 %
500	Stereo	(-512.94, 5.94, -79.26)	519.06	3.81 %
500	Monocular	(-505.37, 5.17, -85.85)	512.64	2.53 %
880	Stereo	(-840.06, 88.42, -292.38)	893.87	1.58 %
880	Monocular	(-841.87, 119.82, -455.69)	976.08	10.92 %

Table 3.4: Accuracy Estimation

3.3.4.4 Effect of Distance from the camera

We expect that the distance of the target from the camera will have a large impact on accuracy and repeatability, because as the target gets smaller, pixellation and noise effects get more pronounced. Also for the monocular case it subtends a smaller angle to the camera which is not good for PnP algorithms and in the stereo case the depth being large compared to the baseline will lead to poor triangulation. Thus we expect both accuracy and repeatability to decline with distance.

We carry out the same experiment as in the previous section with the 2 tags spaced at a constant distance of 240mm apart. This system is placed at different distances (depth) from the camera. Accuracy is estimated as before and the repeatability is estimated by the standard deviation in the translation vector between the tags. Results are given in Table 3.5:

Distance from camera (m)	Tracking system	Euclidean distance (mm)	Error %	Standard deviation (x,y,z) in mm
1.5	Stereo	238.46	-0.64 %	(0.017, 0.006, 0.049)
1.5	Monocular	250.07	4.2 %	(0.111, 0.039, 0.631)
2.5	Stereo	249.74	4.06 %	(0.051, 0.019, 0.174)
2.5	Monocular	244.48	1.87 %	(0.232, 0.254, 2.044)
4	Stereo	275.47	14.78 %	(0.248, 0.052, 0.904)
4	Monocular	241.5	0.63 %	(2.268, 2.193, 25.267)
5.2	Stereo	319.28	33.04 %	(0.681, 0.174, 2.128)
5.2	Monocular	249.08	3.79 %	(0.910, 6.566, 59.330)

Table 3.5: Accuracy and repeatability for given distances

We can observe that the repeatability (especially in the Z direction) is much better for the stereo case than the monocular case, and in both cases it consistently declines over distance, though this is quite steady in the case of stereo and very drastic in the monocular case. For accuracy, the error percentage increases steadily with distance from the camera in the case of stereo vision and in the monocular case it is inconsistent and inexplicably low. More studies can be done as to why the accuracy of monocular vision is abnormally high even as the distance from camera is increased.

Overall our findings from this experimentation is that the stereo camera gives a more consistent result than monocular case. The repeatability is within the limits of the requirements of non-destructive testing but the accuracy is not. Usage of better hardware and doing calibration more carefully would result in better accuracy.

3.4 Motion Tracking Integration

In this section, we will describe the integration of a tracking system into the overall scheme for Non-Destructive Testing. The previous sections have shown us how to determine the pose of the object in camera coordinate frame. This by itself, is often not that useful, unless the tracking camera system is completely fixed and never moves. While viable in some circumstances, in general the tracking camera system will need to be portable and moved around, because of line of sight restrictions and in order to have a portable tracking system. We now show how to get the pose relative to the panel (in world coordinates). This will require us to place a reference tag fixed to the panel (or in the world coordinate system to which the panel is fixed). The reference tag remains constant as the object is moved over the panel. Hence we need to find the relation between the two.

Figure 3.34 shows how the tracking system has a tracking camera (note: tracking camera is either monocular or stereo camera) that is located at some distance away. We define a reference tag and a body tag. The reference tag is fixed to the panel (world coordinate frame) and the body tag is fixed to the object being tracked (an NDT instrument). We need to find the pose of the NDT instrument (body) with reference to the panel (world coordinate frame). If the tracking camera is tracking the reference tag and the body tag at the same time, the pose of body with reference to world coordinates will be independent of the location of the camera.

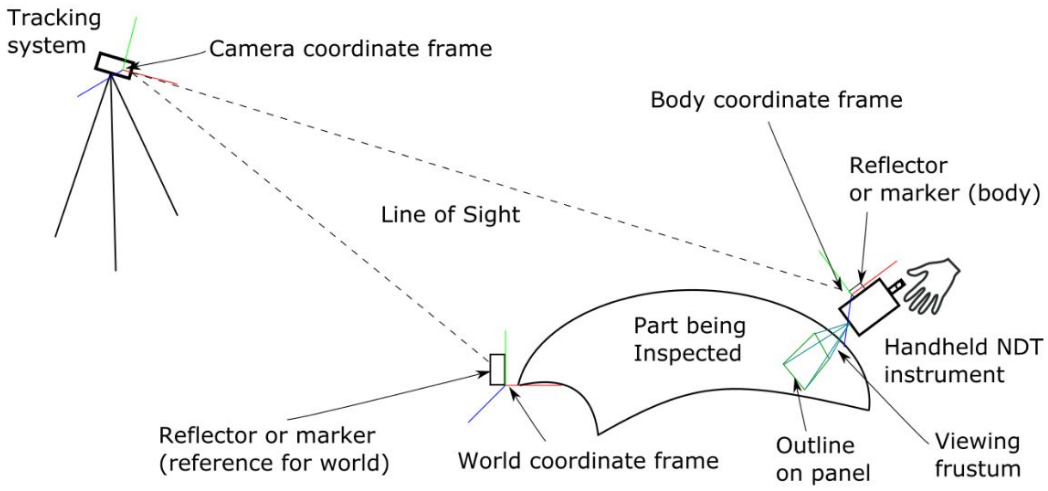


Figure 3.34: Motion tracking complete schematic

Let $\mathbf{R}_{\mathbf{b}}^{\mathbf{c}}$ denote rotation matrix from body to camera reference frame and $\mathbf{R}_{\mathbf{w}}^{\mathbf{c}}$ denote

rotation matrix from world to camera reference frame. $\mathbf{t_b^c}$ and $\mathbf{t_w^c}$ denote the translation vector of body and world (reference tag) in the camera's coordinate frame. These are calculated as per the previous sections. Then we can calculate (Equation 3.5):

$$\begin{aligned} \mathbf{R_w^b} &= (\mathbf{R_b^c})^T \mathbf{R_w^c} \\ \mathbf{t_b^w} &= (\mathbf{R_w^c})^T (\mathbf{t_b^c} - \mathbf{t_w^c}) \end{aligned} \tag{3.5}$$

The outputs of Equation 3.5 are $\mathbf{R_w^b}$: Rotation matrix from world to body frame, and $\mathbf{t_b^w}$: translation vector of body in world coordinate frame.

3.4.1 Tracking for an optical NDT system

The discussion till this point has been broadly applicable for any Non-Destructive Testing system (or in general anything being tracked). We now tackle the specific case of an optical NDT system being used. These optical systems have a camera [1] and they image an object according to what is viewed in the camera frustum. Examples of such NDT systems include thermography, laser shearography and terahertz cameras. These are non-contact, area based, non-hazardous NDT systems that are finding more and more applications in aerospace inspection particularly for composites and foams.

A camera's viewing frustum can be fully defined by three vectors - camera position $\mathbf{p}$, camera target $\mathbf{t}$ and camera's up vector $\mathbf{up}$. The camera position determines where it is located in 3D space, the target determines where it is pointing towards, and the up vector (always perpendicular to the line between camera position and target) is related to the roll of the camera about its optical axis. Thus in order to orient the viewing camera in space, we need to convert the Rotation and Translation parameters obtained into these three vectors.

Figure 3.35 shows the orientation of the viewing camera and the three vectors. The position vectors $\mathbf{p}$, $\mathbf{t}$ and unit direction vector $\mathbf{up}$ can easily be defined in the body tag's coordinate frame. We assume here that the viewing camera is positioned below the tag at distance d_p and oriented towards target at a distance of d_t opposite to the direction of the body's y-axis. The up vector is pointed along the body's z-axis. Then we can define the translation offsets or position vectors of camera and target in the body frame of reference as:

[1] Terminology: The camera in the background is referred to as a tracking camera, while the camera attached to the marker and being used for NDT is called the viewing camera. In case of ambiguity, the camera being referred to should be inferred by context

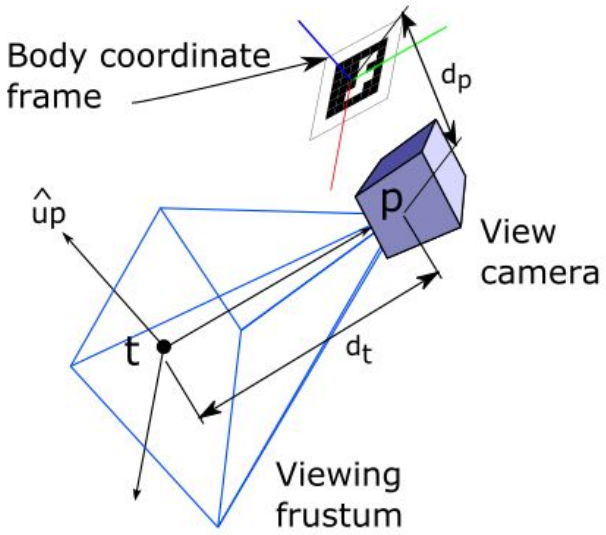


Figure 3.35: Orientation of viewing camera frustum

$$\mathbf{t_p^b} = \begin{bmatrix} 0 \\ 0 \\ -d_p \end{bmatrix}, \mathbf{t_t^b} = \begin{bmatrix} 0 \\ -d_t \\ -d_p \end{bmatrix}, \mathbf{u^b} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.6}$$

where $\mathbf{t_p^b}$ is the camera position vector in body frame, $\mathbf{t_t^b}$ is target position vector in body frame and $\mathbf{u^b}$ is the up direction vector in body frame. Then we can get camera vectors as (Equation 3.7):

$$\begin{aligned} \mathbf{t_p^w} &= \mathbf{t_b^w} + (\mathbf{R_w^b})^\mathbf{T}\mathbf{t_p^b} \\ \mathbf{t_t^w} &= \mathbf{t_b^w} + (\mathbf{R_w^b})^\mathbf{T}\mathbf{t_t^b} \\ \mathbf{u^w} &= (\mathbf{R_w^b})^\mathbf{T}\mathbf{u^b} \end{aligned} \tag{3.7}$$

where $\mathbf{t_p^w}$ is the position vector of camera in world coordinate frame, $\mathbf{t_t^w}$ is the position vector of target in world coordinate frame and $\mathbf{u^w}$ is the direction vector of up vector in world coordinate frame. Once these three vectors are calculated, the viewing frustum is completely determined.

3.4.2 Tracking setup and integration with 3D viewer

We set up an octant shaped panel (part of a propellant tank) and place a reference tag that has a known position with reference to the panel. We place the tracked object which is a marker attached to some legs and a basic camera is used to represent the optical NDT system and is positioned behind the marker and pointed downward. The tracked object is placed on the panel. The imaging is done through a monocular or stereo camera as shown in Figure 3.36

(a) Monocular Setup **(b)** Stereo Setup

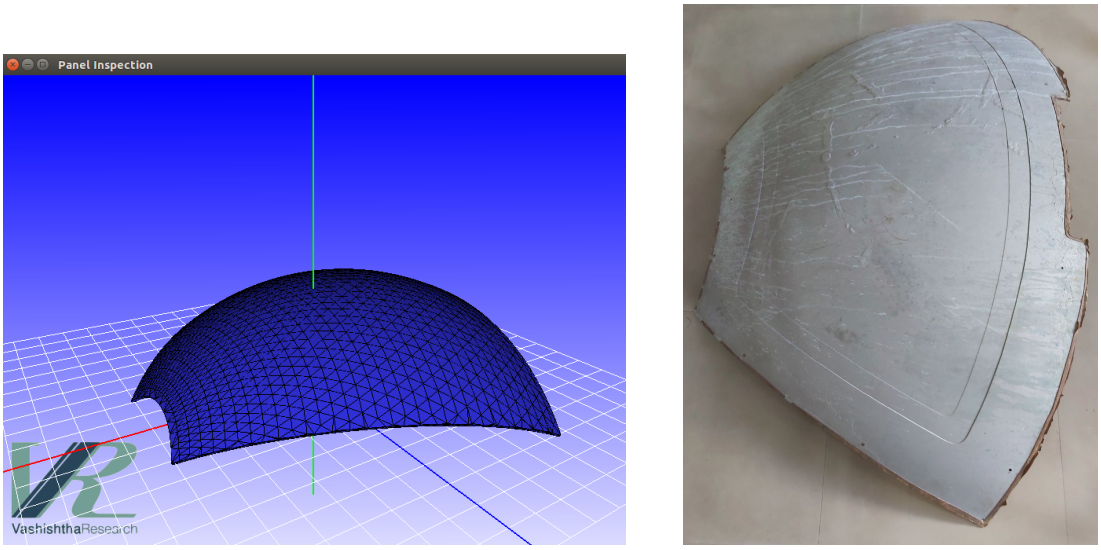
Figure 3.36: Tracking setup for NDT system and panel

A customized 3D viewer was developed using OPENGL and C++, which constitutes a full scale digital model of the tracking system and the panel being scanned. This 3D viewer shows the images being streamed from the tracking cameras and also the 3D representation of the viewing camera frustum as it is moved over the panel.

Figure 3.37a shows the virtual model of our panel as it is loaded into our 3D viewer. Figure 3.37b shows the actual panel being inspected (it is an octant shape which is part of a propellant tank, and the cutout is not included in the virtual model for simplicity). We also have virtual models for a camera showing its viewing frustum and coverage area on the surface.

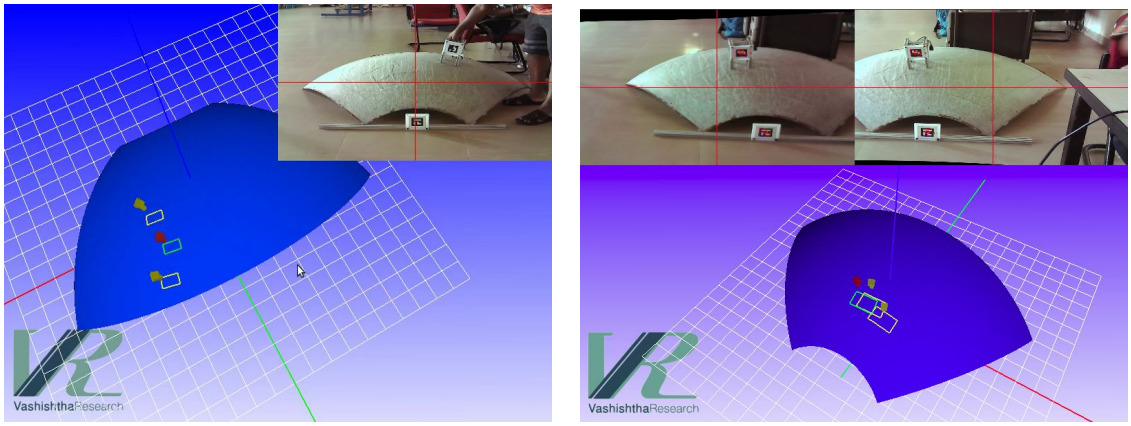
The 3D viewer has the real time feed of the background (tracking) camera on the right top side, with April tags indicated and positioning cross-hairs (Figure 3.38a). In case of the stereo camera, we have the feed from the left and right cameras displayed (Figure 3.38b) after the stereo rectification process.

In Figure 3.38 we can see the current position and orientation of the viewing camera as determined by the pose estimation process of the monocular or stereo tracking process.



(a) 3D viewer with model of panel loaded **(b)** Actual Panel

Figure 3.37: Development of 3D viewers for Panel Inspection



(a) 3D viewer for monocular tracking **(b)** 3D viewer for stereo tracking

Figure 3.38: Integration of 3D viewer with motion tracking.

The outline of the intersection of the camera's viewing frustum is shown in green. This is the area currently being covered by the viewing camera (representing NDT system). The current outline is shown in green and that of previously saved view points are shown in yellow. As we position the NDT system at various view points to cover the whole panel, we will see in the 3D viewer that the entire panel will be covered by these yellow outlines.

The importance of this 3D viewer is that it gives the user a real time knowledge of what area is being covered by the NDT system on the panel in a user friendly way. The pose data is also saved at the viewpoints and can be used for image stitching.

Currently, the algorithm used takes about 700 milliseconds for one set of images to run through the pipeline, all the way from the cameras capturing the images to the pose of the viewing camera frustum being updated in the 3D viewer. This means that when used in real time, it appears very laggy and not smooth. For smooth motion that is undetectable to the human eye, we need 24 frames per second update so the computation loop should be done in 42 milliseconds. The major culprit behind the high processing time is the April tag detection algorithm. Additional work needs to be done on the algorithms and compute hardware for fast motion tracking.

The virtual model software developed comprises various modules, such as loading mesh data (STL files), setting up the NDT System camera (the one held by the robot/ motion tracking system) for viewing and imaging, displaying the camera frustum of the NDT camera and its outline on the surface, setting up the grid system and background, setting up the user interface environment such as rotating, panning and zooming using mouse buttons, projecting rays back onto the surface for selecting points on the surface, and using keyboard keypress actions to initiate algorithms. The algorithms are implemented in separate modules and perform the various activities mentioned in the thesis like integration of motion tracking using April Tags, saving of viewpoint poses and generation of G-code in the path planning process, image stitching using the pointcloud, generation of geodesic paths and coverage path planning. The software development uses C++ OPENGL framework and the external libraries of OpenCV for image processing and April Tag detection. The total number of lines of code involved are around 6000 including all the algorithms (excluding external libraries).

3.5 Summary

We have presented a proof of concept for computer vision based augmented system useful for Non-Destructive testing system that uses monocular or stereo cameras to track April Tags mounted to the NDT system. Additionally assuming that the NDT system is an optical camera based device, we have also linked this pose information from tracking to a 3D virtual model of the panel where the viewing frustum of the NDT system is displayed along with the outline of the panel. The pose information gathered can be used to guide the operator and for image stitching purposes. We also explored what kinds of camera hardware and motion tracking algorithms should be used for the tracking and investigated the accuracy and repeatability of motion tracking.

The major disadvantages of the tracking system developed are the low accuracy and

high processing time. The limitations of the camera hardware used was quantified and evaluated. These can be improved on in future work. Better camera hardware, markers/reflectors, better image processing algorithms and computational hardware will be used in order to create an efficient motion tracking pipeline that is viable by industry standards.

Chapter 4

Viewpoint Planning and 3D Image Stitching Algorithms for Inspection of Panels

This chapter addresses the problem of fully covering a curved panel using a robotic system which holds the Optical based NDT system (such as Thermography or Laser Shearography) up against the panel, and stitching the acquired images together so that the images with defects can be visualized in a single model. A formal way of defining view-points in 3D based on the structure's 3D model and stitching the gathered image data back onto the 3D model has been lacking in the literature. We present algorithms for defining a set of viewpoints for robotic systems using Camera based NDT and for doing full 3D image stitching on the pointcloud generated from the 3D model of the structure. A virtual model of the inspection process taking into account the details of the structural panel, robot and camera model is developed and used for getting inputs from the user for the viewpoint planning process as well as for displaying the output after image stitching. We show excellent results for synthetic image capturing process as well as good results while using the prototype 5-axis cartesian robot with an IP camera to represent the NDT system. This chapter is published as a standalone paper in [31].

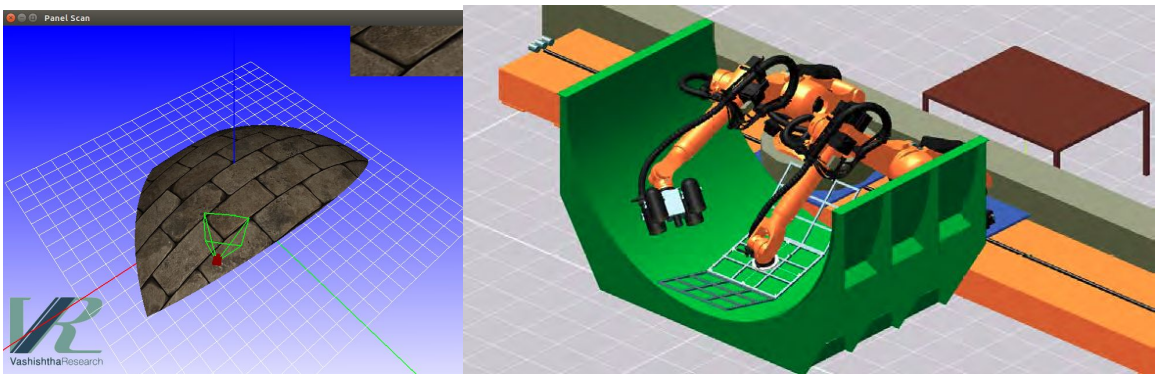
4.1 Introduction

There are many techniques for NDT, including Visual Inspection, Ultrasonic testing, X-Ray, Liquid penetration, Eddy current based, Acoustic sensing, etc. These can be classified into point based or area based methods. For example the ultrasonic sensing is done at a point, while visual inspection uses a camera that covers an area based on its field of view. However, even the camera based methods such as Thermography and Laser Shearography do not cover the entire area that needs to be inspected, based on limitations due to geometry,

resolution and other factors. Thus the movement or scanning motion of the NDT instrument over a surface is an integral part of the inspection process. This can be accomplished manually (using a handheld system), or by using a robotic process such as mounting it on a drone (which can fly over a large area), or a robotic arm system, and many other methods.

In the modern ways of doing non-destructive testing, the the NDT payload is held by a robotic system, which is controlled to move the payload to specific poses, to acquire and record data in the digital form. This integration of the entire system using software is also known by the generic term of Computer Aided Inspection (CAI). We have observed that, even in advanced technical organizations like ISRO, the NDT processes using camera based techniques, even when combined with a robotic system are mostly manually operated, not planned well and it is difficult to reconcile a defect located in the image with the actual location in the panel being scanned. This is because of a lack of 3D virtual model, viewpoint planning and image stitching techniques. The addition of these features adds a great deal value to the NDT system user as he is easily able to plan the viewpoints and view the defects using the virtual model of the scanned structure. This chapter describes the 3D viewer and algorithms for the user to plan the viewpoints for a camera based technique in order to cover a curved structural panel in the virtual model and stitching the data (images) obtained back on to the 3D model of the panel. We also take the kinematics of the manipulator into account during the planning process which is quite important in case a full 6-DOF motion is not available.

Thermography and Laser Shearography are two optical based NDT methods that are camera based and require a certain amount of time for gathering the NDT information. The area that they cover can be expressed as the outline generated by the camera's viewing frustum. Figure 4.1a shows a 3D virtual model where the outline of the camera viewing frustum is shown for a particular viewpoint and the image captured is shown in the corner. The outline will be larger the further the camera is from the surface and will be roughly rectangular if the radii of curvature of the surface panel are large. There is a trade off between the area of surface covered from one viewpoint and the resolution of imaging (and thus defect detection). Generally the area covered for a single viewpoint is small compared to the total surface area of the panel. Thus, the robotic method of scanning a panel using such an NDT technique would be to robotically move the NDT payload to a certain position and orientation (corresponding to the viewpoint) and pause there for a while as the NDT data is gathered, then go to the next viewpoint and so on. This process is considerably different from point based scanning methods such as ultrasonic probes which need to be scanned in lines and the data is gathered simultaneously without stopping the robot.



(a) Generic View point on surface **(b)** Thermography NDT Robotic cell ([32])

Figure 4.1: Camera based Non-Destructive Testing

The problem of computer aided inspection using a robotic system and a camera based imaging method appears to be insufficiently explored. Existing literature in this field mainly deals with point or line based methods such as ultrasonic probes ([33]) , and eddy current probes ([34]). [35] discusses a robotic flash thermography based NDT system for inspecting composite structures. They focus attention on thermographic detection of defects, rather than on planning the coverage of the panel. [33] presents a custom MATLAB toolbox for using ultrasonic through transmission. This is a point based method where the coordination between the two robots holding the transmitter and receiver is very important. [34] presents a computer aided scan path generation for robotic NDT using eddy current probe. Both these approaches are based on scan path generation which is important for point or line based NDT methods. In contrast, camera based NDT techniques require view point planning.

There are some commercial inspection systems as mentioned in [34] such as Genesis Systems for thermography/laser shearography panel inspection using robots. [32] presents a robotic cell using Thermography NDT for use in a production environment (see Figure 4.1b). However the exact techniques for determining the motion plan of the robot is not disclosed in the commercial systems. Offline programming techniques are needed using 3D graphical user interfaces for this viewpoint planning process. We address this problem systematically in this chapter.

After the inspection is complete and the data is gathered, the result would be in the form of several image/video files corresponding to the viewpoints from which the data was gathered. Several hundreds of such files may be generated for even a medium sized panel. In order to view the inspection results for the panel as a whole, an image stitching technique is desired wherein the entire data for the panel can be seen at the same place. Image

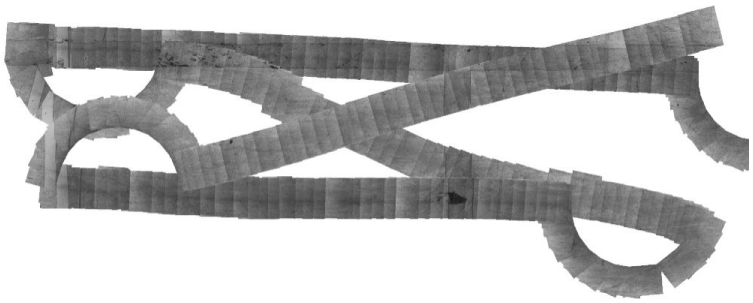


Figure 4.2: Mosiac stitched using images from a mobile robot ([10])

stitching is a well explored field. [36] gives a brief overview of the standard techniques used for image processing and blending. The characteristic features of these standard approaches are that they use matching features in the overlapping region of neighbouring images to calculate the corresponding camera pose, and then the images are warped to fit one of the poses and blended together to form an output image as a mosiac (see Figure 4.2). The NDT data/images gathered from the panel are lacking in features and do not suit such techniques well. [10] implements an image stitching process based on features and visual odometry for a mobile robot, and reports that the process shows difficulty with reflective aluminium surfaces where there are not many features in the captured images.

We depart from these approaches in two fundamental ways - firstly the camera viewpoint pose is used directly from the robot and no matching features in the images are used, and secondly we stitch the images directly onto the 3D model (pointcloud structure) of the panel rather than warping images into a 2D output. Pointcloud stitching/merging and association of pointcloud with images have been previously addressed (eg [11]), but usually in the context of satellite imaging ([37]) , large volume lidar scans ([38]) and autonomous vehicles ([39]). [37] addresses the problem of aligning overhead images (such as those captured by satellites) to 3D pointclouds. [39] performs continous pointcloud stitching for a moving autonomous vehicle. [11] explains the method of registering lidar pointcloud with a set of images corresponding to the same locations (see Figure 4.3). We can see in Figure 4.3 that there is some discontinuity at the image boundary as no blending method is used during the image stitching process of [11]. Blending methods are discussed in [36] and the blending process essentially assigns weights to pixel values in the overlapping region to produce a smooth looking overlap between images. In this work we apply these techniques of image registration to pointcloud, stitching and blending for the application of Non-Destructive testing for panels.

The rest of the chapter is organized as follows. Section 4.2 describes the 3D viewer and

Figure 4.3: Pointcloud registered with stitched images ([11])

virtual model system, and the viewpoint planning interface for generating view points and associated robot motion. Section 4.3 describes the image stitching and blending methods. Section 4.4 shows the results for such a viewpoint planning and image stitching process carried out with a 5-axis cartesian robot system and the simulated system, and also evaluates the effectiveness of this method. Finally section 4.5 summarizes and concludes the chapter.

4.2 Viewpoint Planning

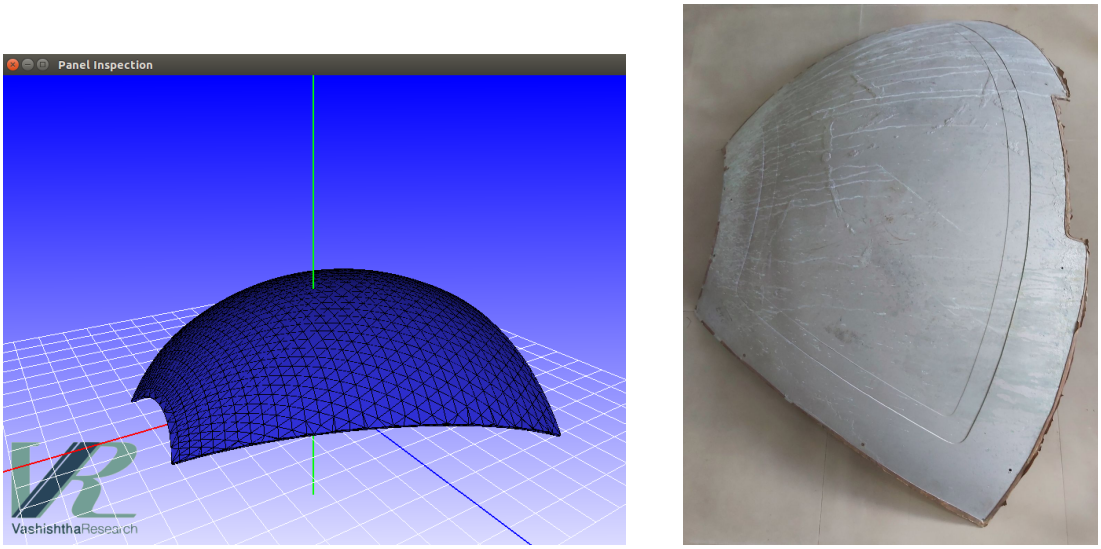
One of the main objectives in Computer Aided Inspection with an camera based NDT method, is to plan a set of viewpoints from which the surface will be inspected. We also want to stitch together the data gathered from the inspection into a single composite image. We take the point of view that the robotic system enables us to place and orient the NDT instrument with very good accuracy.

In our approach, viewpoint planning is done by an operator, using a computer model of the panel to be imaged. When the operator specifies a viewpoint, he is shown the outline of the region which will get imaged from that viewpoint on the panel. In addition, he is also shown the regions corresponding to all the viewpoints he has selected so far, so that he can place "to be imaged" regions with overlap, to cover the whole panel.

In this section we describe the virtual model of the panel, how the "to be imaged" region is obtained, particularly when we use a robot with less than 6 DoF, and then how the robot would be commanded to move to image all the selected regions.

4.2.1 Virtual 3D Model

In this research work, it was felt necessary to develop a customized 3D viewer using OPENGL and C++ which constitutes a full scale digital model of the robotic system and the panel being scanned. The reason for this is to avoid using proprietary software and also to make sure that that we have an appropriate representation of the process without making



(a) 3D viewer with model of panel loaded **(b)** Actual Panel

Figure 4.4: Development of 3D viewers for Panel Inspection

any compromises. This software plays an integral role in all of the algorithms to be discussed in further sections. The algorithms implemented by us require the use of computer geometry to define and compute paths on meshes, the implementation of camera models and robot kinematics, and the selection of view point using mouse inputs. At least one or more of these features are not available in common open source software.

Figure 4.4a shows the virtual model of our panel as it is loaded into our 3D viewer. Figure 4.4b shows the actual panel being inspected (it is an octant shape which is part of a propellant tank, and the cutout is not included in the virtual model for simplicity). We also have virtual models for a camera showing its viewing frustum and coverage area on the surface. The geometry of the robot manipulator and its degrees of freedom are also incorporated into the virtual model for positioning of the camera.

4.2.2 Viewpoint Selection

The method described here is a user based or manual method of selecting the viewpoints using the 3D viewer and software. The selection of the viewpoint is done manually by the user by using the mouse buttons. An automatic path planning approach is described in Chapter 5 (Coverage Path Planning).

Figure 4.5 describes the 3D-viewer interface used for the viewpoint planning process. The triangle mesh (STL file) of the panel is loaded and placed in the 3D viewer. Blue region with white grid lines represent the floor, with the grid representing an area of approximately

2mx2m. The virtual model of the panel is shown placed on the floor. The X,Y and Z axes are represented using the red, green and blue lines. A texture is superimposed onto the surface. This texture has no relation to the real panel and is just there for generation of synthetic images. We did not attempt to capture the actual surface texture of the panel as we had no 3D scanner or other means of doing so.

The camera model is rendered at points of the previously saved viewpoints in yellow color and at the currently selected viewpoint in green. The outlines on the panel corresponding to the viewing area from the saved viewpoints are shown in yellow and that of the current viewpoint in green. The camera's viewing frustum is also displayed in green outlines for the current viewpoint. The view of the panel as seen from the current viewpoint is displayed in the right hand corner (Synthetic image). This textured surface captured by the virtual model of the camera is used to generate synthetic images which are used as inputs for the image stitching algorithms.

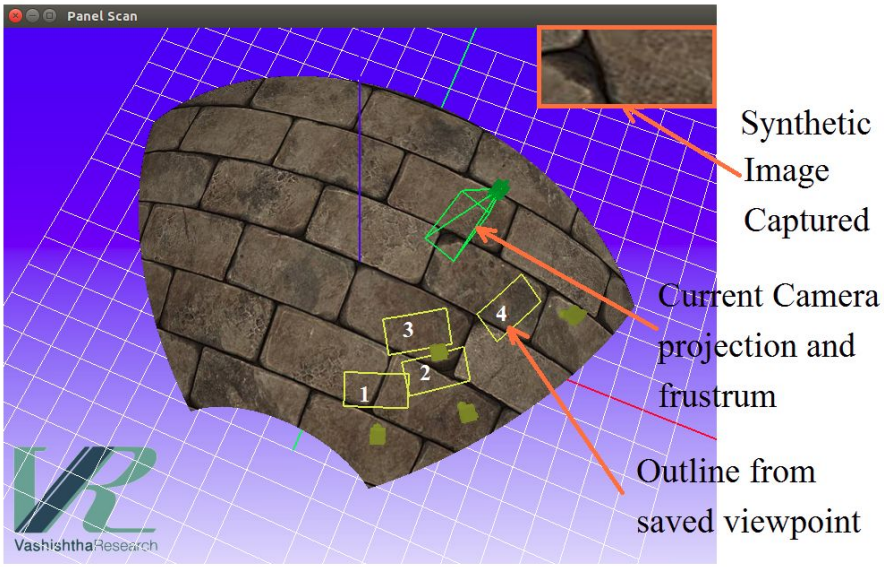


Figure 4.5: 3D viewer for scanning viewpoint selection

When the complete virtual model is loaded up, the user would see the virtual model of the panel, the location of the camera and the synthetic image captured by the camera from that location. He can then move the camera over the panel by holding down the right mouse button and moving it over the panel. The entire 3D view can be rotated, panned and zoomed using the left mouse button and scroll wheel. He would then plan the sequence of viewpoints on this 3D model.

Our viewpoint planning technique using the virtual models of the panel and the robot

is as follows. The surface point is selected by clicking a point on the panel. This point is translated from the 2D output frame (viewport) into 3D-space and a ray is generated using ray casting techniques. The intersection of the ray with the surface is found by checking ray-triangle intersections with all the triangles in the mesh. If more than one intersection is found, we select the closest point. The algorithms for ray-casting and ray-triangle intersection are done using standard techniques common in computer graphics (refer eg. https://www.scratchapixel.com/) and are not repeated here.

The viewpoint is located at a standoff distance d_0 from the surface point along the normal. The value of d_0 is selected by the user as per the needs of the NDT instrument. Figure 4.6 shows the projection of the camera viewing frustum on the surface. The camera is placed at viewpoint $\mathbf{v}$ and pointed at surface point $\mathbf{p}$. The camera's 'up' vector also needs to be specified in order to complete its description. This 'up' vector selected will become the binormal vector and accordingly the tangent vector at the surface point will be calculated. The normal, tangent and binormal $\mathbf{n}, \mathbf{t}, \mathbf{b}$ are unit vectors that form a right handed vector basis. Here, the tangent vector is parallel to the horizontal axis of the camera and binormal vector is parallel to the vertical axis of the camera. Both the tangent and binormal vector will be in the tangent plane of the surface at point $\mathbf{p}$. The outline of the viewing frustum's intersection with the surface will be approximately a rectangle of size p_h x p_v.

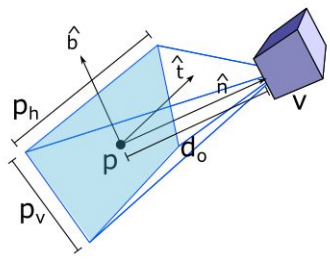
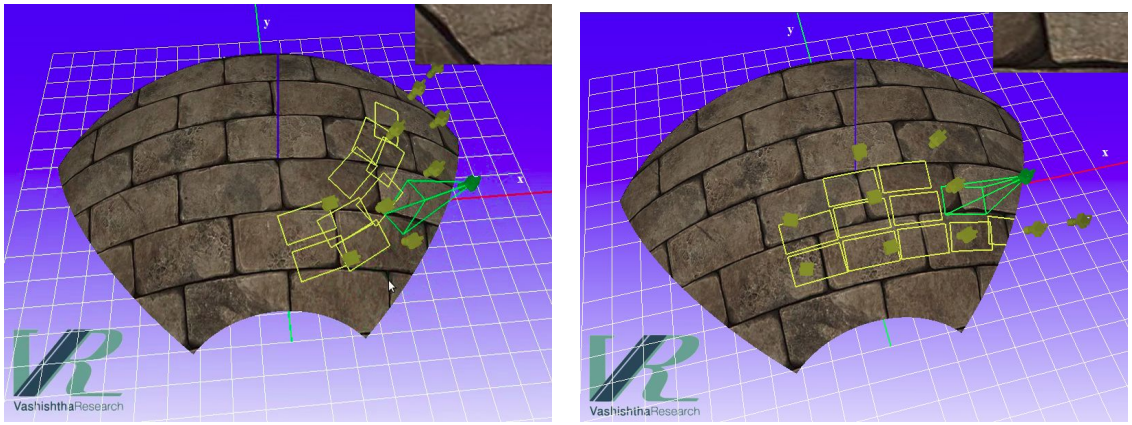


Figure 4.6: Viewpoint and perspective camera projection

The selection of the camera's 'up' vector (binormal) is a matter of some consideration. If we have only a 5-axis robot, such as our prototype cartesian robot, then this is not an independent selection since we cannot roll the camera about the camera optical axis. If we have a 6 axis robot with full 6-DOF, then this up vector can be selected at our discretion. A good selection for this specific panel is in the direction of the y-axis of the world coordinate system as in Figure 4.5. The viewpoint planning and outlines on the panel according to the

5-DOF planning system and 6-DOF planning systems are shown in Figure 4.7a and Figure 4.7b respectively. It can be noted that while the 5-DOF planning system can cover the surface, the 6-DOF planning system is more natural and intuitive. Since we only have a 5-DOF robot, this model is used in the remainder of this chapter as implementing a 6-DOF motion in a 5-DOF robot is impossible.



(a) Viewpoint selection for 5-DOF robot **(b)** Viewpoint selection for 6-DOF robot

Figure 4.7: Viewpoint selection

Once the viewpoint is located, we can render a 3D model of the viewing camera held by the robot, at that position and orientation. The coverage area of this camera on the surface must then be determined and indicated on the surface. The method for this is to create a frustum using 4 triangles and runs triangle-triangle intersections to find the intersection of the viewing frustum with the triangle mesh of the surface. The intersecting lines generated would be shaded in green thus creating the outline. It can be noted that sometimes the outline of the viewing area will not fall entirely on the surface and will be cut off at the edge (eg. Figure 4.7b).

Also note that we have created a virtual model of the viewing camera with the same parameters as that of the real camera. The camera's intrinsic model parameters are given in Table 4.1 which corresponds to an ELP IP camera. This can be modified depending on the optical NDT system used. This virtual model of the viewing camera is used to generate the synthetic image seen in the right hand corner of the 3D viewer scanning application.

We can now save the current viewpoint, thereby automatically saving also the joint variables calculated using inverse kinematics, and G-code corresponding to that particular viewpoint for the robot to be moved later on. It will also save the synthetic image captured by the virtual camera. The synthetic images will be saved in a folder. The previously saved viewpoints and outlines are displayed in yellow color while the current ones are shown in

Parameter	Value	Units
WIDTH	1280	Pixels
HEIGHT	738	Pixels
θ_{VFOV}	18	Degrees

Table 4.1: Camera parameters

green. In case the viewpoint orientation or location is physically outside the workspace of our robot (which may happen at the extremeties), the camera is shown in red and does not allow saving of that viewpoint (Figure 4.9b).

4.2.3 Machine code generation and running

Upon saving the viewpoints, inverse kinematics is carried out to determine the joint variables corresponding to each viewpoint and thus the G-code to be sent to the cartesian robot system. For a 5-DOF robot this calculation is given in Equation 4.1. This corresponds to the manipulator schematics as seen in Figure 4.8.

$$\begin{aligned} \theta &= \tan^{-1}(n_y/n_x) \\ \phi &= \tan^{-1}(n_z/\sqrt{n_x^2 + n_y^2}) \\ \mathbf{w_c} &= \mathbf{v} + j_o(\mathbf{n})/\|\mathbf{n}\| \end{aligned} \tag{4.1}$$

Where $\mathbf{n}$ is the normal corresponding to the surface point, $\mathbf{v}$ is the viewpoint position and j_o is the joint offset or the length from the wrist centre w_c to the focal plane of the viewing camera. θ denotes yaw or A angle and ϕ denotes pitch or B angle. The coordinates of the wrist centre w_c give the x,y,z joint values. The joint variables obtained for the 4 viewpoints showing in Figure 4.5 (The areas corresponding to viewpoint 1,2,3,4 are marked) are calculated and listed in Table 4.2.

X (m)	Y (m)	Z (m)	θ (rad)	ϕ (rad)
0.231949	-0.45877	0.559405	1.93258	1.07531
0.411362	-0.331514	0.573231	2.21026	1.06421
0.291743	-0.234479	0.643132	2.14004	1.17802
0.56556	-0.0148821	0.613765	2.68619	1.14056

Table 4.2: Joint variables for 4 viewpoints

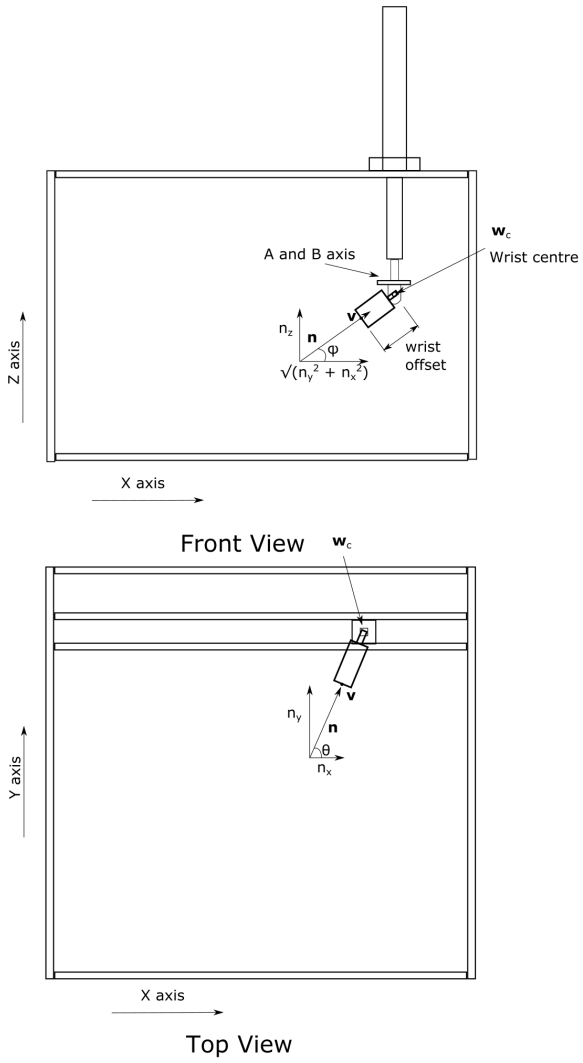
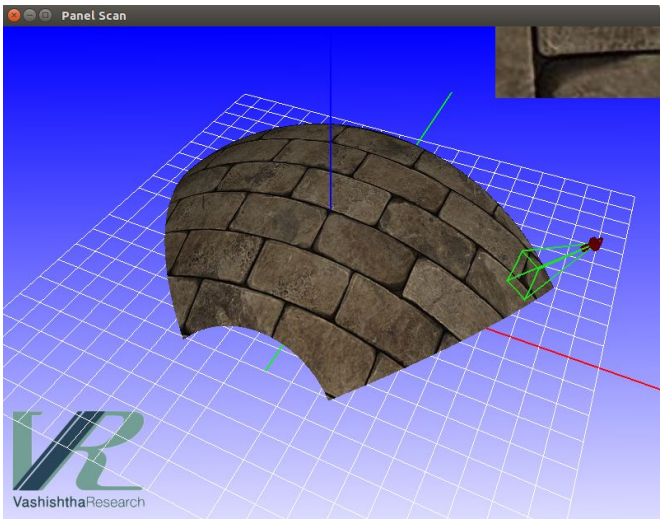


Figure 4.8: 5-DOF Manipulator schematics

The G-code corresponding to these joint variables is generated and sent to the cartesian robot (Figure 4.9a). The Cartesian robot has been described in detail in Chapter 2 After executing one line of the G-code, the robot will stop at the viewpoint and we will capture the image of the panel. This process will be repeated for all viewpoints. The sequence of viewpoints visited will be the same as recorded by the user.

However before the robot can be run in this manner we need to do workpiece registration. In this process we align the workpiece (ie. panel) to the coordinates of the robot workspace and vice versa. For this process live view from the camera with crosshairs superimposed on top of it is streamed to the user (Figure 4.10a). We put the camera pitch

(a) Cartesian 5-DOF robotic cell with panel



(b) Unreachable viewpoint

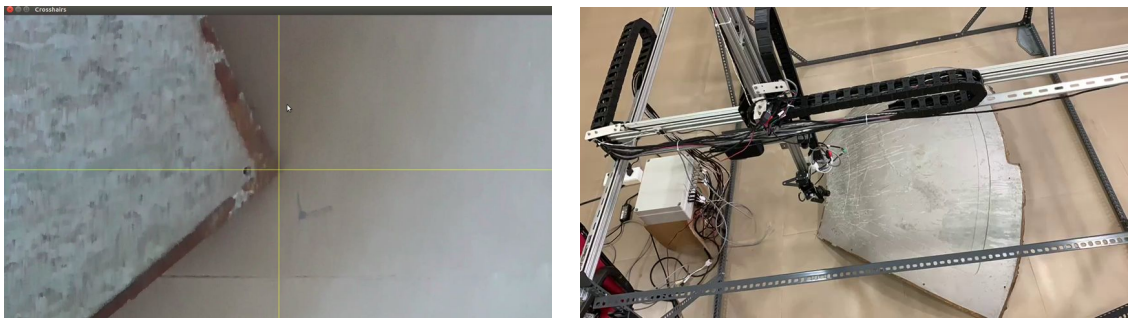
Figure 4.9: Cartesian Robot workspace

axis vertically down and align the yaw axis so that the crosshairs are in the same direction as X and Y axis. This is needed to zero the rotary axes. We then place the workpiece panel into the robotic cell such that the workpiece corner point is exactly positioned at the centre using crosshairs. Manual Pulse Generator (MPG) pendant is used for fine motion for this purpose. We will then obtain the offset coordinates ie the current X, Y, Z positions of the robot. We can then move the robot end effector to the expected location of another corner point and gently rotate the panel (without shifting the first point) until that corner point comes into position. Since this workpiece rests on a plane, 2 points are sufficient for workpiece registration. Similar processes can be done easier if the robot is equipped with a probe or laser displacement sensor.

In our case the panel was placed in an orientation which has stability. Usually the panel is presented to the robot in a manner which enables it to scan the required areas. This could require additional fixtures. The ability of a robot to scan a given panel fully, and the best way of presenting the panel to the robot are relevant issues, which fall outside the scope of the thesis.

Cartesian robot scanning in motion is shown in Figure 4.10b. After every viewpoint is reached, we record the images. This process is an analogue for an NDT system. For the real NDT system we would capture NDT data rather than just a visual image. For example

in the case of Thermography, infrared images/videos would be captured. In case of Laser Shearography, speckle pattern images corresponding to the strain field would be recorded.



(a) Viewing camera with crosshairs **(b)** Cartesian robot in motion (Top View)

Figure 4.10: Cartesian Robot operation

4.3 Image Stitching

Once all the viewpoints are stored and the images from the robot are recorded, we will then use an image stitching algorithm to stitch together all the images. It is necessary to stitch all the images rather than just defect information as it may not be possible to automatically find all the defects using image processing. Thus we need to display the complete data for the user to identify the defects. Also the nature of a fault spanning two or more images would become more clear from such a complete projection.

The novelty of our method compared to other image stitching techniques is that we do not use features of the images or overlapping feature correspondence, since these may not be available on NDT images of a panel, particularly in the defect free portions. This is especially true for Thermographic images which are less clear than standard RGB images. Instead we already have the pose of the viewpoint that we know the robot has been positioned at, with the accuracy provided by the robot controller.

An alternative method which does not use overlapping feature correspondence would be using a sensor for obtaining the pose of the NDT system, such as Depth cameras, stereo vision tracking, laser tracker etc. (See, for example [40]) However there are no depth cameras or other directly built in solutions available for Thermography or other camera based NDT methods. Combining the Thermographic data with another sensor (such as RGBD depth camera) would lead to a very complex solution which is unnecessary in this case as the pose information is available from the robot itself. The method of obtaining

pose information from depth cameras or other motion tracking devices can be very useful in cases where the NDT device is being moved by hand.

Our approach is as follows. First we generate a pointcloud out of the triangle mesh of the panel. Each point in this pointcloud is then independently examined as falling in one or more images. The end result is a grey-scale or coloured pointcloud whose properties are related to the images to which it belongs. This automatically accomplishes stitching of images too.

For the purpose of image stitching on pointcloud, we have made use of Cloudcompare, an open source pointcloud viewing and editing software.

4.3.1 Pointcloud generation

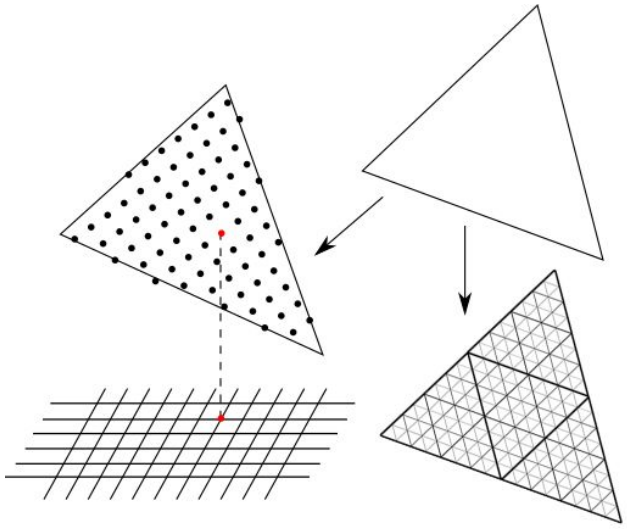
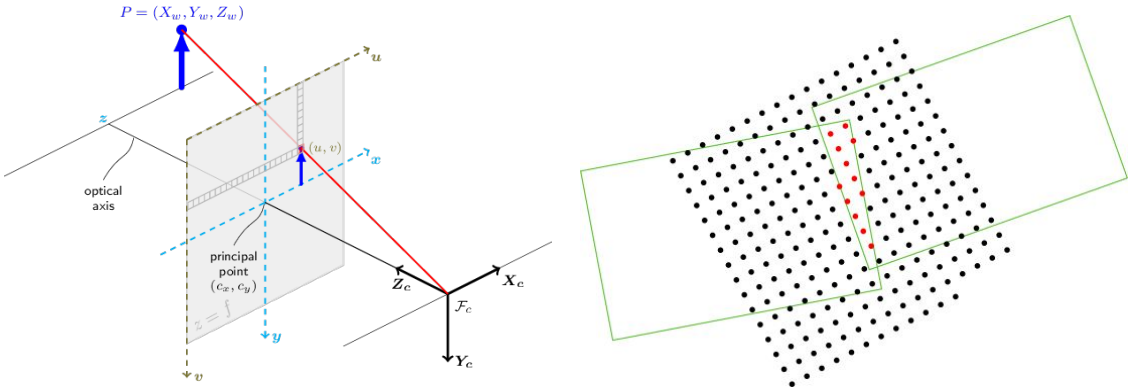


Figure 4.11: Generation of the pointcloud

The generation of the pointcloud from each triangle of the surface mesh can be done in many ways. One method uses triangle subdivision to create sub-triangles, and those sub-triangles can then be subdivided as well. Catmull-Clark ([41]) or Loop subdivision methods could be used and subdivision could be halted once a triangle is sufficiently small. After some iterations, the vertices of all the triangles thus created can be taken as the pointcloud. Another method could involve projecting a ray from the 2D grid pattern and taking the point of intersection between the ray and the triangle. The points generated would thus form the pointcloud. This process is shown in Figure 4.11. We are using triangle subdivision for pointcloud generation in this work.

Once the pointcloud is generated, we use the camera model to find where that point would be located on the image taken from a particular viewpoint. The pixel location is found by rounding off this point to the nearest integer. As we used a camera instead of an NDT instrument in this work, Pinhole Camera Model (Figure 4.12a) is considered and the parameters are that which are corresponding to the real camera (focal length, pixel dimensions). The virtual camera model uses the same intrinsic parameters (Table 4.1) used for viewpoint planning and the camera location and orientation is known from the viewpoint. If the point projected to the camera sensor plane is outside the field of view, then this viewpoint is ignored. If the point is within the field of view, then the corresponding pixel values (R,G,B) are taken into consideration to assign the value to the point. A point can be imaged in more than one viewpoint. This will be observed in the overlapping regions (Figure 4.12b). In this case blending would need to be done.



(a) Pinhole Camera model **(b)** Overlapping regions in Pointcloud

Figure 4.12: Pointcloud mapping to viewpoints

Standard pinhole camera model (refer eg. https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html) is used and not elaborated here. After the pointcloud is generated, we end up with 5.15 million points representing the surface of the panel. Our 3D viewer software is currently not equipped to handle such a large number of points. Hence CloudCompare (https://www.cloudcompare.org) open source pointcloud viewer software is used for viewing the results.

4.3.2 Blending

The development of the "stitching onto a pointcloud" method was thus done in order to get a high resolution and more useful output. After locating all the viewpoints for a particular point (in which that point has been imaged), we need to blend the data of the different

images of the point. If we simply assign the value from the last viewpoint, then we will get a situation where the overlapping region looks as though the images have been cut and pasted on it (Figure 4.13). (This result is obtained from the process described in Section 4.4.2 .)

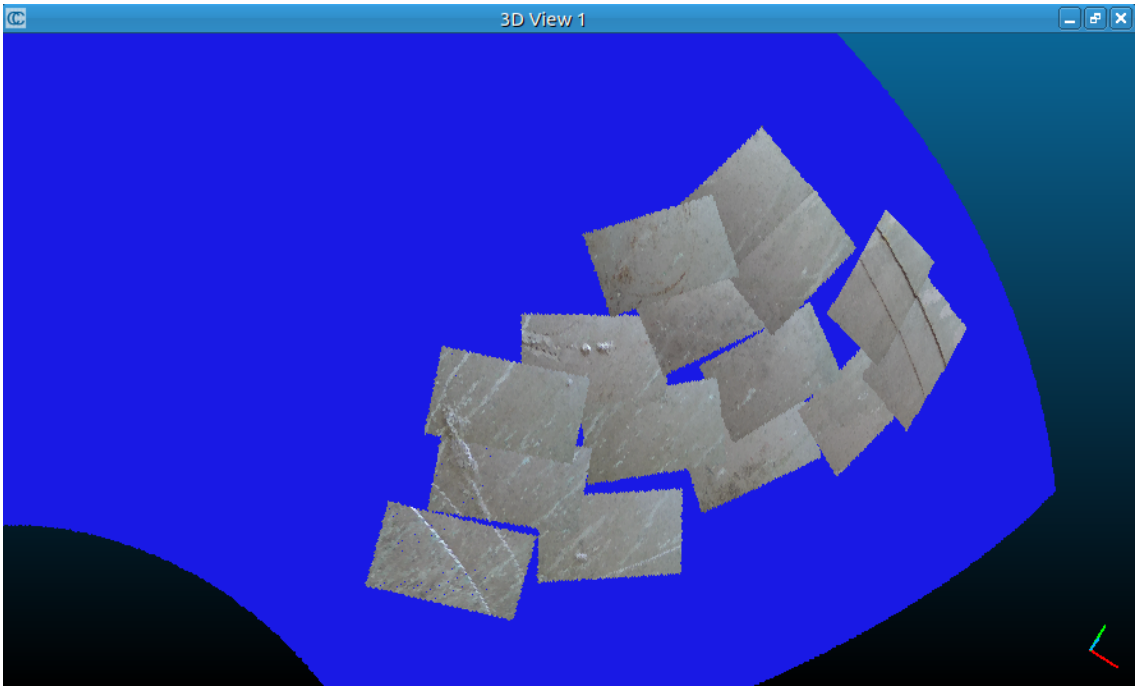


Figure 4.13: Overlapping images with no blending

The method for blending that we apply is a centre weighting technique using grassfire transform. This simply means that the weight assigned to a pixel in an image is more if it is near the centre and less if it is near the edge of the image. The distance to the edge can be computed by using grassfire transform ([36]). Mathematically,

$$d(x, y) = min(x_{max} - x, x - x_{min}, y_{max} - y, y - y_{min}) \tag{4.2}$$

This distance can then be used to compute the weighted average pixel value as:

$$p_{avg} = \frac{\sum_{i=1}^{n} d_i p_i}{\sum_{i=1}^{n} d_i} \tag{4.3}$$

Where n denotes the total number of viewpoints this point is imaged in. After blending, we can see the output in Figure 4.14 for 15 viewpoints.

More detailed results of the image stitching process is given in the next section.

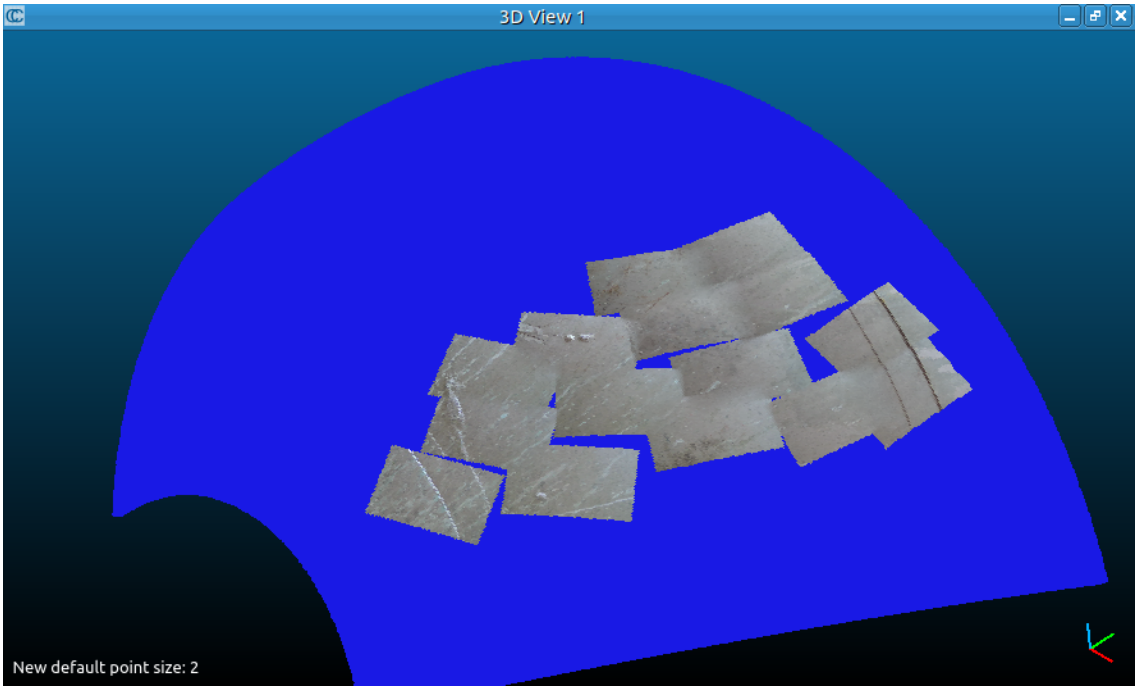


Figure 4.14: Pointcloud with stitched and blended images

4.4 Results

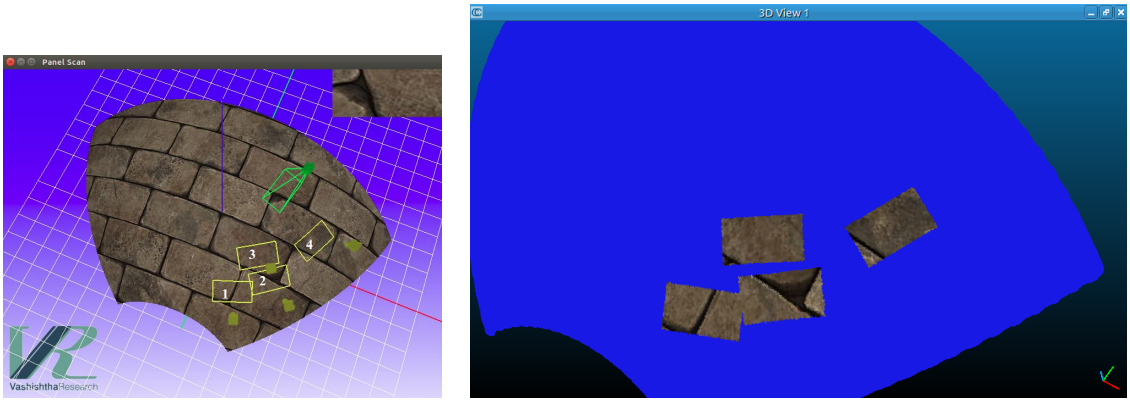
The previous two sections have detailed how the process of viewpoint planning and image stitching are done. As the surface of the panel does not have any regular pattern, we generated synthetic images with a regular pattern (texture) during the viewpoint planning and imaging process. The next step is to run it on the robotic system to obtain the real images as well.

The results are presented here for three cases. For the first two cases (Section 4.4.1), only the synthetic images generated are used, and in the third case (Section 4.4.2) the synthetic images and real images are both used. Evaluation is carried out in the third case as well (Section 4.4.3).

4.4.1 Synthetic image results

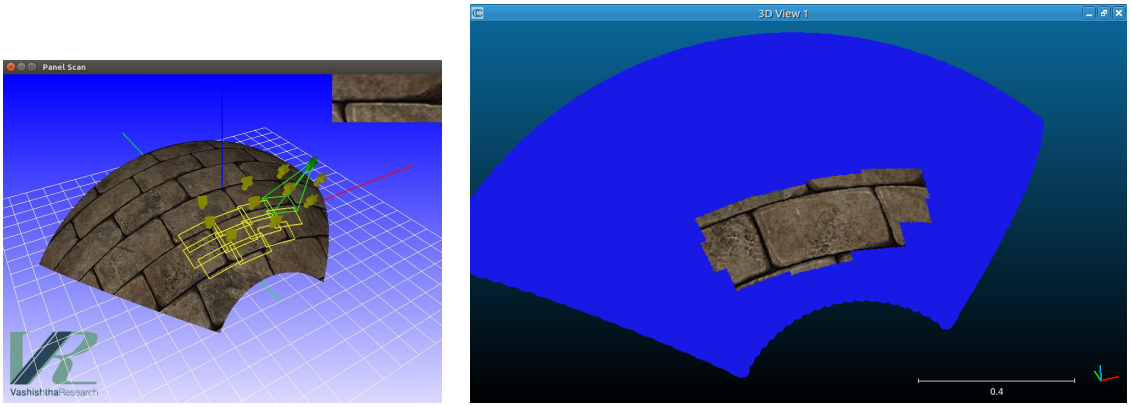
The purpose of generating synthetic images and running the stitching algorithm on them is to verify the theoretical correctness of our method. For 5-DOF planning case, we can compare the image stitching and viewpoint planning side by side for the 4 viewpoints as given in Figure 4.5 and Table 4.2. This is shown in Figure 4.15b and Figure 4.15a. In this case, the viewpoints selected are widely dispersed across the panel, and any point not

imaged in any of the viewpoints remains blue in color.



(a) Four viewpoint selection **(b)** Pointcloud for 4 viewpoints synthetic images

Figure 4.15: Synthetic images for 4 viewpoints



(a) Selection plan for 6-DOF motion **(b)** Pointcloud for 6-DOF synthetic images

Figure 4.16: Synthetic images for 6-DOF motion coverage without gaps

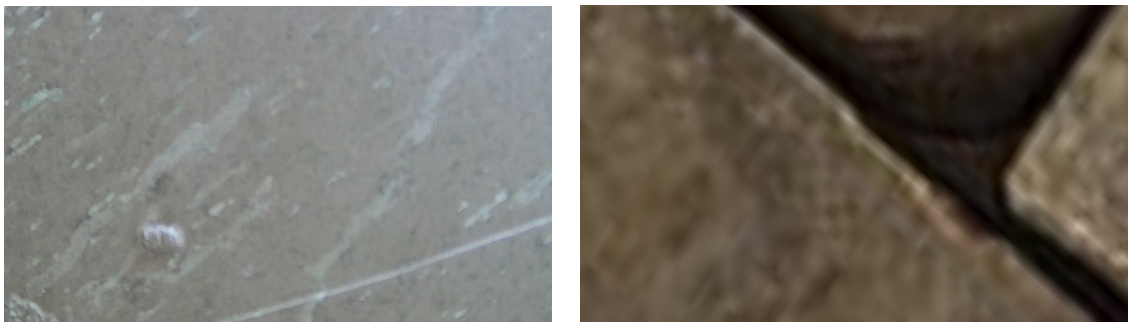
For the case of 6-DOF planning, we can see the plan in Figure 4.16a and the stitching results in Figure 4.16b. In this case the viewpoints chosen are carefully selected to give a complete coverage of a region of the panel with no gaps in between the viewpoints. We could not however execute this plan on the real system on account of not having a 6-DOF robot.

4.4.2 Real image results

In this case, we plan a 5-DOF path with 15 viewpoints and some gaps are intentionally left between the viewpoints (For the case of a coverage without gaps, see Figure 4.16). We

deliberately left these gaps in order to illustrate that the image stitching process depends only on the viewpoint from which the image was captured and not on the image itself.

After the robot is moved through the viewpoints and images are recorded, we have one set of synthetic images and one set of captured images. An example from each set is shown in Figure 4.17a and Figure 4.17b. Note that both are RGB images of same resolution (1280x768).



(a) Real image captured **(b)** Synthetic image generated

Figure 4.17: Simulated and captured images

These images and the viewpoint pose are the input for image stitching. We perform viewpoint planning and image stitching for 15 viewpoints in 5-DOF planning mode. This results in a 3D pointcloud as shown in Figure 4.14. A zoomed in view at high resolution is shown in Figure 4.18.

The image stitching results from the synthetic images from the same 15 viewpoints are shown in Figure 4.19.

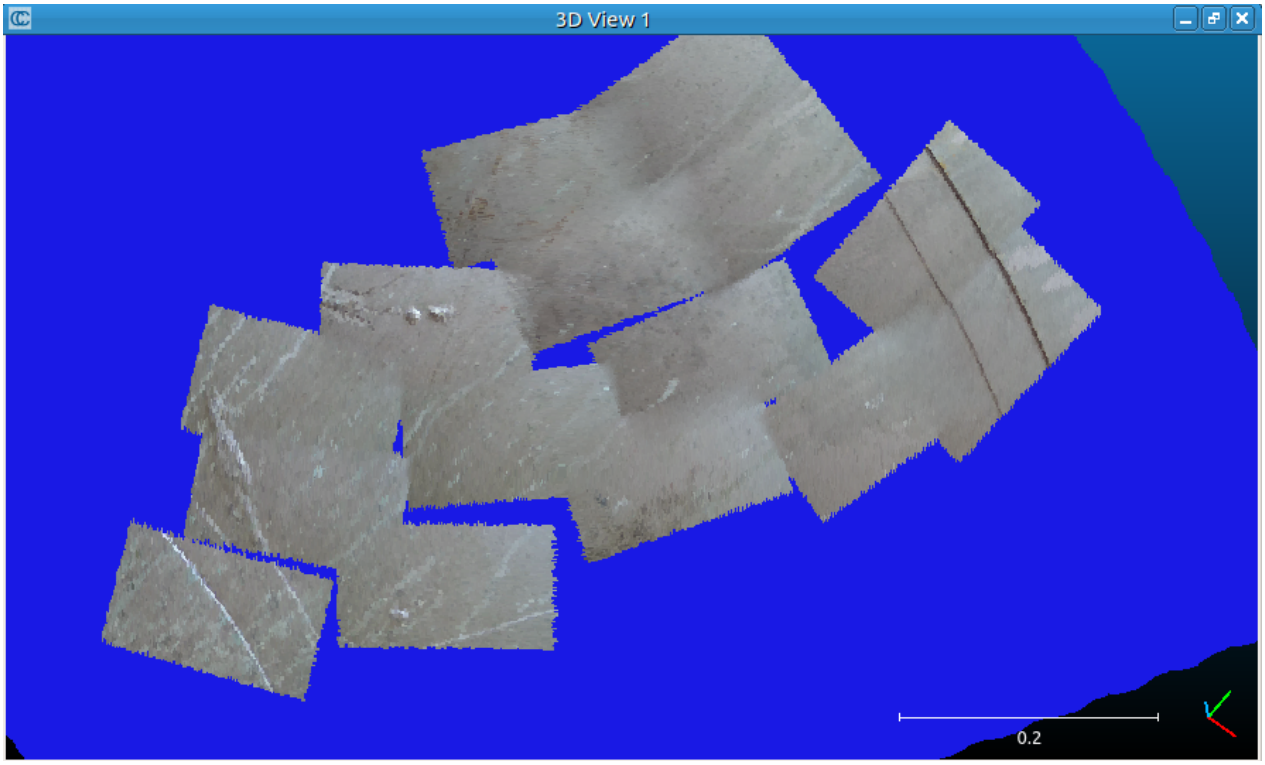


Figure 4.18: Pointcloud from real images (zoomed in)

4.4.3 Evaluation

As we can see in Figure 4.19 (and also Figure 4.15b, Figure 4.16b), the stitched pointclouds from the synthetic images are perfect as expected, and require no further comments. This shows the theoretical correctness of our method. For the pointcloud resulting from the real images, we see that the major details of the panel surface have been captured. Notably there is a change in illumination for 4 images in the top right corner possibly due to shadows. Changes in illumination are common while recording real world data if the environment is not maintained exactly same.

The pointcloud obtained in Figure 4.18 from the real images shows discontinuity as a result of poor accuracy of the robot and non-uniform lighting conditions. In comparison with Figure 4.3 ([11]), which is based on a similar technique of projecting camera data onto pointclouds, we can see similar discontinuity as a result of dissimilar lighting conditions. In actual thermographic NDT imaging conditions, we can assume a uniform lighting would be provided by the halogen or flash lamps used and a shroud would be provided to block out stray light, thus resulting in less discontinuity. Other imaging techniques also assume uniform illumination through laser or LED lighting.

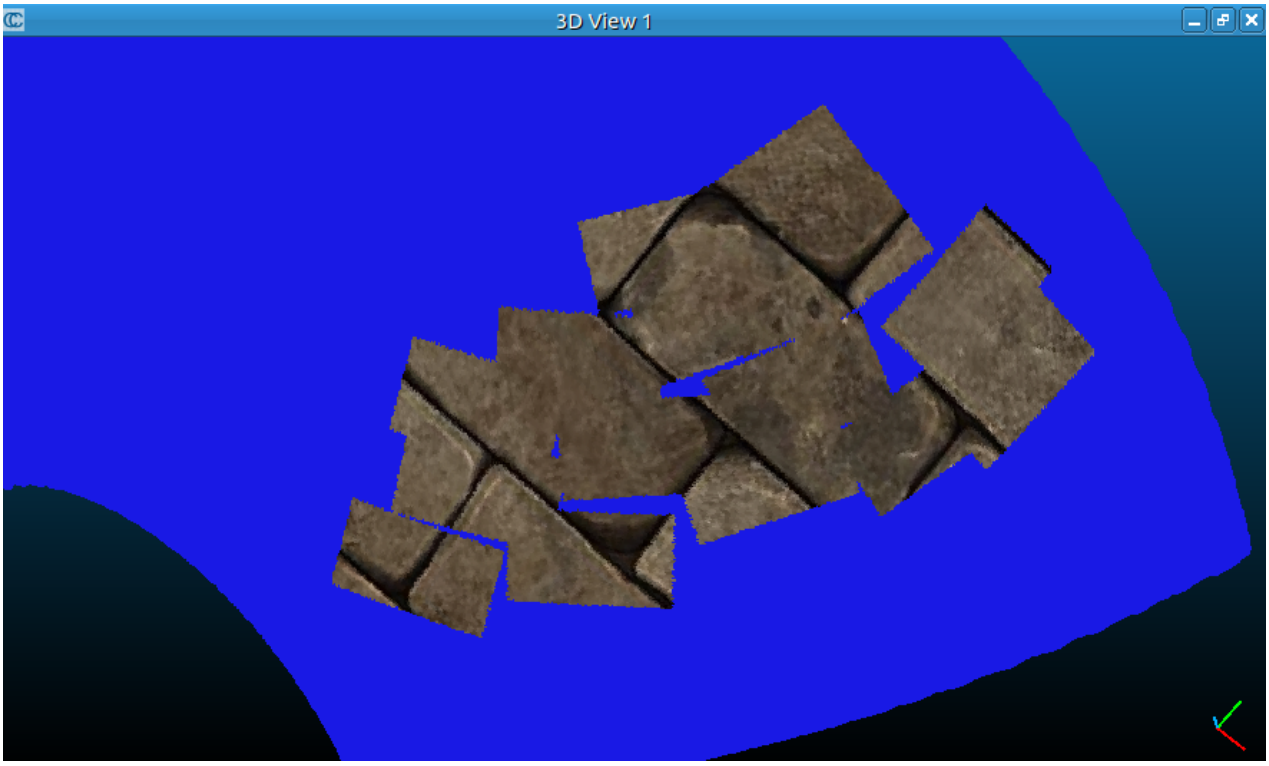
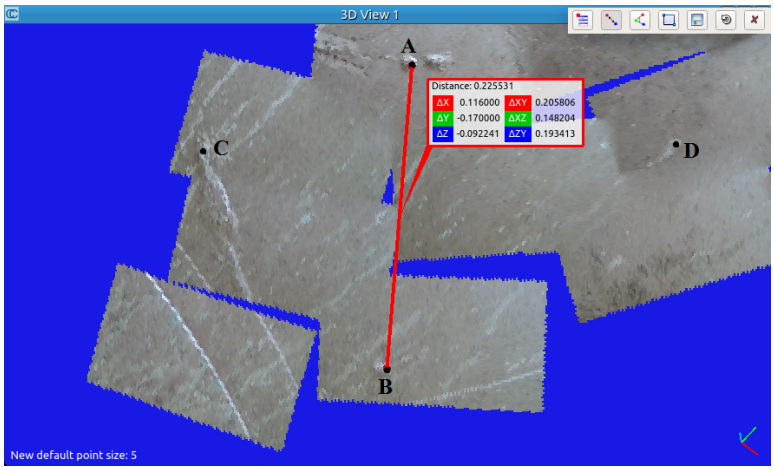


Figure 4.19: Pointcloud from synthetic images (zoomed in)

Direct comparison of our technique with conventional image stitching methods is difficult, because these methods are based on feature correspondences which are very less or non-existent in these images. As an example, we tried using OpenCV's cv2.Stitcher program with 2 of our images captured from the experiment, but it failed to provide any result. We can expect the same to be true for NDT images as well.

In order to evaluate the effectiveness of our technique of viewpoint planning and image stitching algorithms, we should use some metric. The metrics usually used in image stitching are based on feature correspondences ([36]) but we cannot use that since we are not using features in the images taken by the viewing camera for image registration. One possible metric is gauging the straight line distance between two artefact points on the panel, on the real model of the panel (Figure 4.20b) and on the 3D stitched pointcloud (Figure 4.20a). This pointcloud was derived from real images of the panel taken by the robot as per the technique discussed section 2 and section 3. We measure the distances between 4 points A,B,C,D as showin in the figure. Only the distance AB is marked in (Figure 4.20a), the others are listed in Table 4.3. The measurements on the panel are made using a vernier calipers, since the vernier calipers measures the straight line distance between the two points on the panel and the pointcloud measurement also gives the straight line distance.



(a) Distance Measurement on Pointcloud

(b) Measurement on Panel

Figure 4.20: Distance measurement

Point	Measured distance on pointcloud (cm)	Real distance on panel (cm)	Difference (cm)
AB	22.55	24.90	2.35
BC	20.82	23.74	2.92
CD	34.91	39.10	4.20
AC	16.63	18.30	1.67
AD	20.15	23.30	3.15
BD	27.02	29.41	2.39

Table 4.3: Pointcloud Distance Measurement Accuracy

We observe a deviation of about 1-4 cm between the measurements on the pointcloud and the panel. The difference can be explained by the following factors:

- Inaccuracy of the robot
- Inaccurate representation of the camera model
- Inaccurate registration of the panel to the robot

We can get better results by improving on all these factors. In particular, when we estimated the accuracy of our cartesian robot in Chapter 2, it was found to have inaccuracy of the order of 1.75 cm per metre of travel in X and Y axes. The poor performance of the robot can be attributed to the fact that it is just a prototype robot with inaccuracies in fabrication and assembly, and also has some play. This is the main factor why the distance measured on the pointcloud is lower than that of the real panel.

A possible way to deal with the inaccuracy of the robot would be to ensure that there is sufficient overlap between images by decreasing the planned projection area in the view-

point planning process (Figure 4.21). If the estimated x and y translation errors are δx and δy respectively, and if the errors in the first and second revolute axes (first revolute axis is in x direction and the second is perpendicular to it) are $\delta\theta$ and $\delta\phi$ respectively, then the original projected area $p_x \times p_y$ has to be reduced to $(p_x - 2\delta x - 2d_0\delta\phi) \times (p_y - 2\delta y - 2d_0\delta\theta)$ during the planning process, in order to make sure all areas are covered.

This does not take curvature of panel into account, and models the compensation needed for error in revolute axes only approximately. More accurate compensation for the inaccuracies of the robot and other models are currently left to the future work.

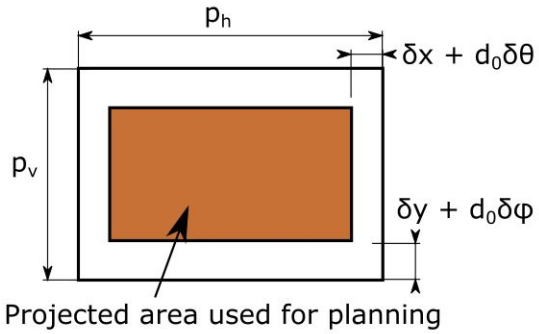


Figure 4.21: Reduction in projection area

The computational complexity of the image stitching algorithm is $O(nv)$ where n is the number of points in the pointcloud and v is the number of viewpoints. For our experiment with 5.15 million points and 15 viewpoints the process takes a few minutes on an ordinary laptop. The process could be speeded up by representing the pointcloud as an octree data structure.

4.5 Summary

We have demonstrated a simple and effective technique for planning viewpoints for a robot based NDT system, and stitching the images obtained onto a 3D model (pointcloud). The techniques worked perfectly for synthetic images. Some discrepancies were noted for the real images captured by the robotic system, due to inaccuracies in the prototype robot, inaccuracy in camera model, workpiece registration and variations in lighting. View point planning with the 5-axis robot is difficult as there is no control on the camera roll orientation, and hence the facilitation we provide for the user for covering the entire panel, is particularly useful.

The future work for this would include use of a 6 axis cartesian robot (which we are developing) carrying the actual NDT system, which will allow for more intuitive view point planning. This robot would be more rigid and will have better workpiece registration processes and thus accuracy will be improved. The NDT system camera will be calibrated and modeled more accurately and lighting condition will be made uniform. The NDT data will replace the RGB images used in this work .

The thermal images (NDT data) are temperature values which can be converted to RGB false color images. Functionally for image processing purposes these false color RGB images are similar to normal RGB images as captured through a regular camera. Hence the proposed algorithm will be applicable to thermographic images. Our thermographic NDT system is still under development at this time and thus the application of this technique to thermography will be addressed in a future work.

Some computational improvements are also possible by using octree data structure for pointcloud. This will lead to a fully developed integrated system for optical based inspection.

Chapter 5

Semi Automated Coverage Path Planning for Inspection of Panels using Camera Viewpoints

In this chapter, we improve upon the manual selection of viewpoints for the coverage of the surface, as presented in Chapter 4 , by introducing an automated system for path planning. The coverage of a surface using multiple viewpoints is a topic of great interest for robotic path planning in inspection applications. Two approaches for coverage path planning are broadly addressed in the literature —geometric methods and optimization methods. While the optimization methods may be the most flexible, they are frequently difficult to implement in practical applications due to their computational complexity.

This chapter describes a geometric algorithm for the coverage path planning of panels used for aerospace applications using a generic camera model that can represent area inspection techniques like thermography and laser shearography. This algorithm relies on drawing a 2D grid on the 3D surface of the panel using geodesic lines on the surface. The coverage of the surface is done by propagating geodesic lines from a starting point until the patch thus covered diverges too much from a flat surface, and after that, the coverage is continued from another point. The propagation of the geodesic lines is stopped when they begin to converge or diverge, and we define two criteria for the stoppage. We show that the proposed algorithm has good results for 3D virtual models and emphasize its speed, simplicity, and reliability for such applications. This chapter is published as a stand-alone paper in [42].

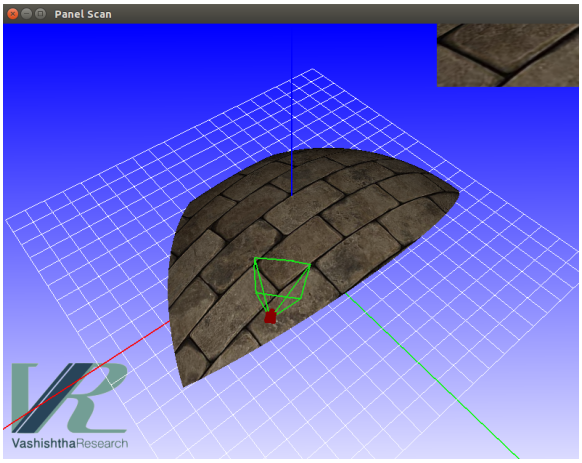
5.1 Introduction

In certain applications in robotics and autonomous vehicles, we desire to cover an area, and we would like to obtain a path by which the entire area is covered in the best way using some metric such as the shortest distance, lowest travel time or energy expenditure. Examples of such applications include vacuum cleaner robots covering the floor, lawnmower robots cutting grass on a lawn, ploughing robot covering a field, inspection of underwater oil rigs and pipelines using autonomous submarines and so on. This generic problem is called coverage path planning. [43] give an overview of the generic applications of coverage path planning.

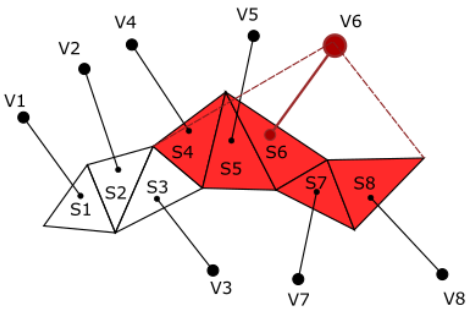
In the case of inspection applications where the sensor is held at a distance, what we desire to know is a set of viewpoints from which the entire surface can be inspected using the sensor at those viewing points. This problem is generically known as View Planning Problem (VPP) and in case we need to plan a route from one viewpoint to another (such that all viewpoints are covered), then that problem is a subset of Traveling Salesman Problem (TSP). Both VPP and TSP are well known to be NP hard problems (NP Hard problems are those which cannot be solved in polynomial time, for more information see [44]), and their combination TVPP (Traveling view planning problem) is thus also NP hard ([45]). There are also other ways of framing the problem such as art gallery problem, watchman route problem and so on (see [43]) but the common solution to these is based on numerical optimization. [46] is another survey on Coverage Path Planning algorithms, but this too covers mostly optimization based techniques.

The problem addressed here is the imaging of a curved panel for the purpose of Non-Destructive Testing (NDT), so as to cover the entire surface. We assume that the imaging will be done by a device like a camera. The device (camera based NDT system) will be moved to a set of view points, oriented at the appropriate angle and then the images would be taken. Since the view point can be placed anywhere in space and oriented in any direction, in general there may be infinite number of view points from which the surface may be viewed. Chapter 4 (Viewpoint planning and image stitching) has described a method for planning the coverage by manually selecting each viewpoint. Figure 5.1a shows a generic view point at which a camera is placed and oriented with its viewing frustum capturing the image of the panel from that viewpoint (according to the parameters of the view camera). Considering that a surface mesh is a discrete representation of the panel surface consisting of surface elements like polygons, we can reduce the solution space by just assuming a single viewpoint for each surface element which is normal to the surface at that point.

This is shown in Figure 5.1b, where we have generated a view point for each surface patch (the surface patch here being one triangle of a triangle mesh) using the normal to the surface. Each view point can capture more than one surface patch (in this example, viewpoint V6 covers surface patches S4 to S8). Then the optimization problem becomes a matter of choosing a minimum number of viewpoints such that all the surface patches are viewed.



(a) Generic View point on surface



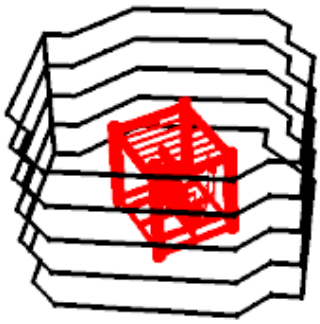
(b) View points corresponding to surface patch

Figure 5.1: View points on surfaces

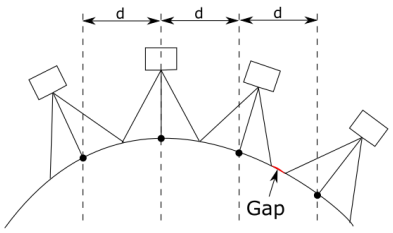
If we use the above approach with the surface modeled as a triangle mesh we will be having thousands of surface elements and view points even for a simple surface and these will all be variables for optimization and thus would need a commercial solver to solve it in any reasonable amount of time. [45] shows a typical approach to solving TVPP with numerical optimization using IBM Cplex solver, which is not only expensive but difficult to integrate into custom applications. Another example [47] explores the topic of coverage path planning of outdoor structures with UAVs using optimization based methods, the results of which can be seen in Figure 5.3a. In [47], the church model has been simplified to just 200 triangles in order to keep the optimization algorithm from taking massive amount of time. [48] explores coverage path planning for structural inspections using a drone, and various heuristic methods to solve the optimization problem (eg. genetic algorithms) are implemented here.

An alternative approach to these optimization methods and solving NP-hard problems is to use geometric algorithms, however care must be taken to use the appropriate algorithm for the problem at hand. The main contribution of the present work is to demonstrate the usage of a geometric method instead of complex numerical optimization methods and show that we get good results for the 3D virtual models of real world examples. [49]

describes a geometric algorithm using slicing planes for 'circling' around an underwater structure using an underwater robot for inspection (see Figure 5.2a), and [43] also describes a similar circling strategy for UAVs in urban environment. [50] describes a basic geometric algorithm for coverage path planning for spraying using drones.



(a) Circling path for underwater inspections

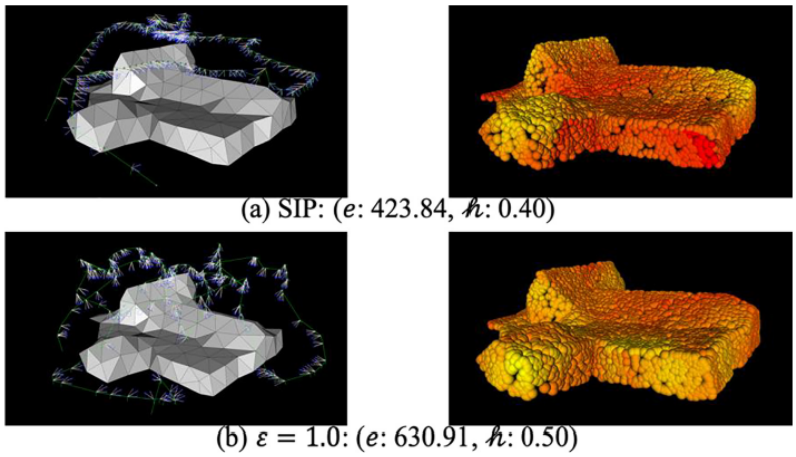


(b) Failure of a basic slicing algorithm

Figure 5.2: View points on surfaces

Geometric algorithms based on slicing the surface based on planes are commonly used in 3D printing applications, however if the distance between planes is kept constant then we will immediately run into problems for even the simplest doubly curved surfaces (see Figure 5.2b for how a gap in between viewpoints has occurred), and in general such a technique cannot be used efficiently without making some assumptions about the surface mesh. [49] have implemented the circling algorithm for their underwater inspection and note that it is 'inflexible' and 'not optimal'. Slicing algorithms have also been implemented in [51] for the purpose of inspecting large structures like Wind turbines using UAVs (Figure 5.3b). Although geometric algorithms do not guarantee optimality or complete coverage, it is possible to use the geometric methods for applications such as panel inspection for aerospace systems, and obtain efficient algorithms that give near optimal solutions by selecting the appropriate input parameters carefully. In addition, geometric algorithms are well known for their speed, simplicity and ease of use.

This chapter describes a simple geometric approach based on using the geodesics of the surface. The near rectangular camera image sensor projections are placed, side by side, with overlaps to ensure that all points on the surface are covered. What is done is to place one rectangle, and then construct adjacent rectangles by moving in the length and width directions along the geodesics of the surface in those directions. The advantages of



(a) UAV inspection of church using optimization methods [47]

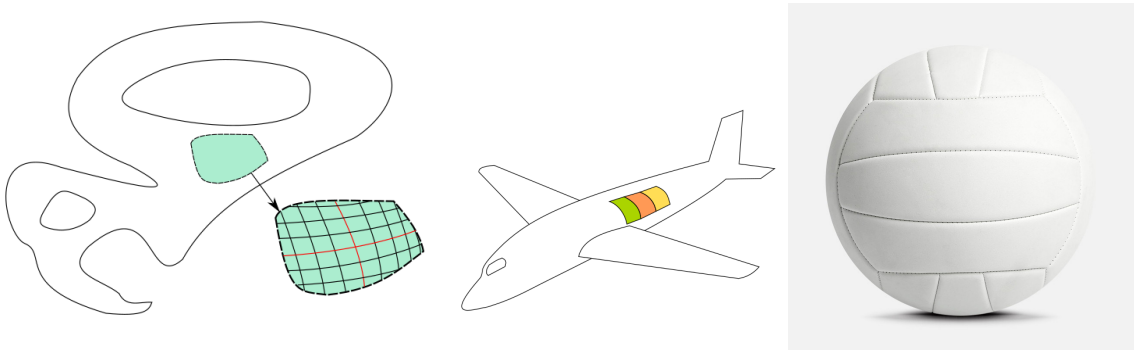
(b) UAV inspection of wind turbine using slicing based geometric method [51]

Figure 5.3: Coverage Path Planning of structures using UAVs

using geodesics are that (a) on near planar surfaces geodesics can form a near rectangular grid, and (b) geodesics are well defined curves which can be constructed using simple algorithms.

The entire surface cannot reasonably be mapped by a single grid, or we will run into problems because of the non-developable nature of the shape in general, even though a small section of the surface will always be near-developable. Figure 5.4a shows a generic shape that can be decomposed into several sections, each of which are near developable ie. a 2D grid can be mapped on the surface without much distortion. Going from the generic to the specific, this model of using sections as parts of surfaces is exactly what is used in the Aerospace industry (see Figure 5.4b), and the surface of the aircraft or launch vehicle is made of large number of such sections which are also called panels. These panels are usually manufactured (and inspected) separately and then fitted together into the final vehicle. We show our results on panels taken from the real world aerospace industry - an octant which is part of a propellant tank and a cylindrical surface with cutouts that is part of a launch vehicle. The notion of constructing a non-developable shape out of sections is far from being restricted to the aerospace Industry, see for example the construction of a volleyball in Figure 5.4c Thus our technique can be used for a number of other applications as well.

One issue with the use of geodesics is that on complex curved surfaces they could converge or diverge, leading to neighbouring view-rectangles overlapping too much or not



(a) Generic shape and section **(b)** Aircraft structure panels **(c)** Volleyball

Figure 5.4: Surfaces and sections of a shape

overlapping and creating gaps. This requires some manual intervention and appropriate partitioning of the total surface into segments. We automate this process to some extent by stopping node propagation when too much convergence or divergence is detected. We use two criteria for this, the 'indirect' criterion based on Gauss map of the surface normals, and the 'direct' criterion based on the distances between neighboring nodes. The results are explored for a wide range of panel shapes relevant to Aerospace applications. The coverage algorithm here is "Semi-automated", in which the user is expected to select the starting criteria for node propagation and then save the viewpoints. This provides far more versatility than 'Fully automated' mode which can have several failure conditions.

For finding geodesics, we use the approaches proposed by [52] and [53]. [52] gives an algorithm to compute a *discrete geodesic path* on a surface mesh, and [53] uses this algorithm to fit woven cloth (resembling a 2D grid) onto the surface. We proceed in a similar fashion as [53] but dispense with the notions of warp, weft and other terminologies related to fabric draping and replace the node mapping algorithms in [53] with our own node insertion algorithm. We show that the algorithm achieves good results for real world scenarios.

The method presented in the present work has the advantage over optimization based techniques in being fast and simple to implement. Its advantage over other geometry based techniques is that it is much more versatile than slicing based methods. Finally it has an advantage over just manually planning the path (such as that in Chapter 4) in that it is semi-automatic and requires much less human intervention than planning the pose of each individual viewpoint.

It is to be noted that the current work focuses primarily on planning the viewpoints given start criteria provided by the human user, by decomposing the surface into regularly

arranged neighbouring camera viewpoints. The user input is very important to achieving proper coverage using this method, thus making it a semi automated technique. Once the set of these viewpoints has been generated, they can be linked together into a path. However, the travel planning process can be done trivially, or by using known methods to solve Travelling Salesman problem (see Section 5.6). We also note that in the given application of camera based Non-Destructive Testing, the position and orientation of the viewpoints and the coverage of the surface that they provide is of paramount importance, while optimally planning the travel plan or sequence of viewpoints is not that important because the robot can move very fast between viewpoints, but needs to hold at a viewpoint for much longer time to record the data. Hence the algorithms described in this chapter are basically about creating the set of viewpoints for covering the surface.

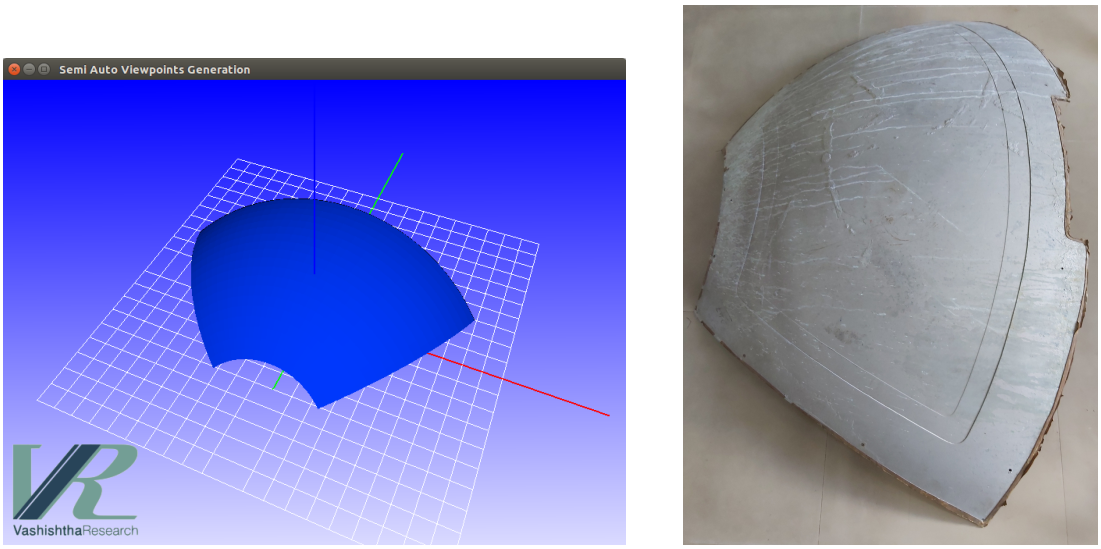
The rest of the chapter is organized as follows. Section 5.2 gives an overview of the user workflow and inputs needed for the entire process. Section 5.3 details the discrete geodesic path algorithm that is used to generate curved paths on the surface. Section 5.4 explains our node insertion algorithm that maps nodes all over the surface to generate viewpoints and also incorporates the criteria for which the node propagation needs to be stopped. Section 5.5 shows the results of the coverage algorithms on a variety of panel shapes. Section 5.6 discusses how to link together all the viewpoints into a motion path, and finally section 5.7 concludes and summarizes the chapter.

5.2 Overview of process workflow

In this section we detail the overall workflow of the process of generating the coverage path on the panel. First, we explain how the surface model is loaded into the 3D viewer that has been developed in C++ and OPENGL framework. Since this is a Semi-automatic technique, the user will be prompted to enter the initial criteria for starting the node propagation process. The node propagation will continue till the entire surface is covered, as long as the stoppage criteria is not met. After the node propagation process is stopped, the user can save these viewpoints and start again at another point. He can continue this process until the coverage is completed.

5.2.1 Loading Model into 3D viewer

The customized 3D viewer developed in Chapter 4 is utilized for this application. Figure 5.5a shows the virtual model of our panel as it is loaded into our 3D viewer. Figure 5.5b



(a) 3D viewer with model of panel loaded **(b)** Actual Panel

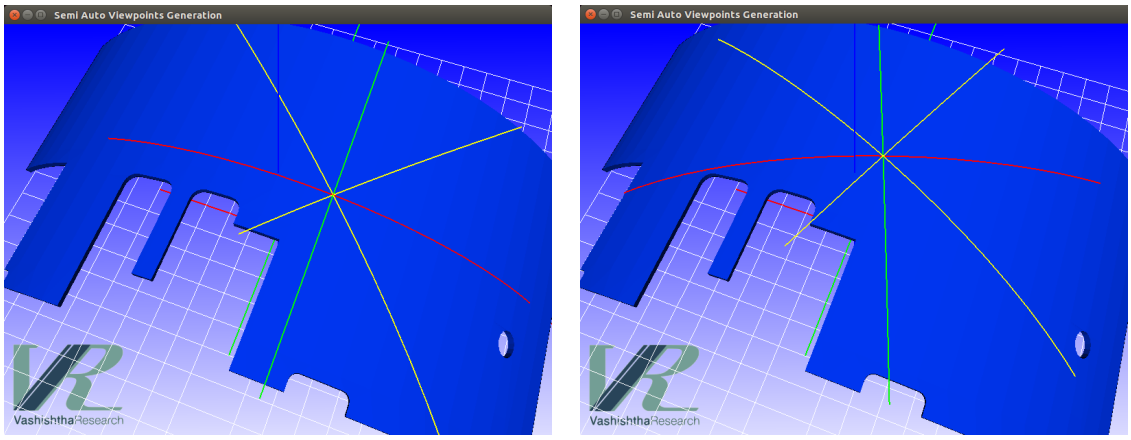
Figure 5.5: Development of 3D viewers for Panel Inspection

shows the actual panel being inspected (it is an octant shape which is part of a propellant tank, and the cutout is not included in the virtual model for simplicity). The triangle mesh (STL file) of the panel is loaded and placed in the 3D viewer. Blue region with white grid lines represent the floor, with the grid representing an area of approximately 2mx2m. The virtual model of the panel is shown placed on the floor. The X,Y and Z axes are represented using the red, green and blue lines respectively.

5.2.2 Selecting the starting point on the surface

The starting point is selected by right clicking on the surface. This point is translated from the 2D output frame (viewport) into 3D-space and a ray is generated using ray casting techniques. The intersection of the ray with the surface is found by checking ray-triangle intersections with all the triangles in the mesh. If more than one intersection is found, we select the closest point. The algorithms for ray-casting and ray-triangle intersection are done using standard techniques common in computer graphics (refer eg. https://www.scratchapixel.com/) and are not repeated here.

The primary axes for the coverage algorithm (ie the X and Y directions along which the geodesic needs to be propagated) then need to be defined. By default, the primary axis of the 3D viewer (ie. X axis) is projected onto the tangent plane located at this point, and this defines the X or tangent direction. The Y direction or bi-normal vector can be found using the normal and tangent vector. This process is shown in Figure 5.6a. The X and Y (or



(a) Selecting the starting point on the surface **(b)** Rotating the starting direction

Figure 5.6: Selection of the starting parameters

tangent and bi-normal directions) are used as the starting direction to project geodesics on the surface upto a certain distance. These geodesics are colored red (X direction) and green (Y direction). Another two sets of geodesics (coloured yellow) are shown for diagonal directions. The algorithm for generating the geodesic paths is explained in more detail in Section 5.3.

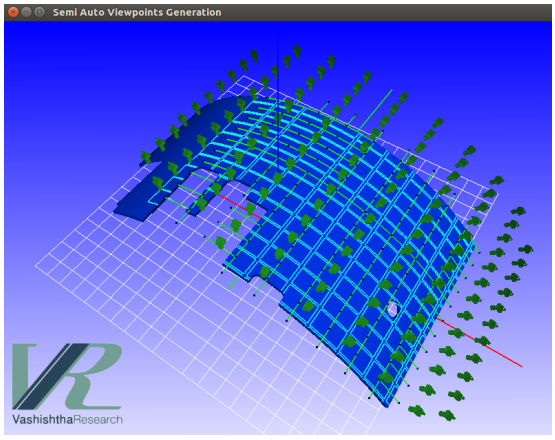
5.2.3 Rotating the starting direction on the surface

The user can rotate the start direction (ie. X-axis or tangent vector) on the tangent plane of the starting point using scroll button while holding down the right mouse button. An example of rotation of the start direction is shown in Figure 5.6b.

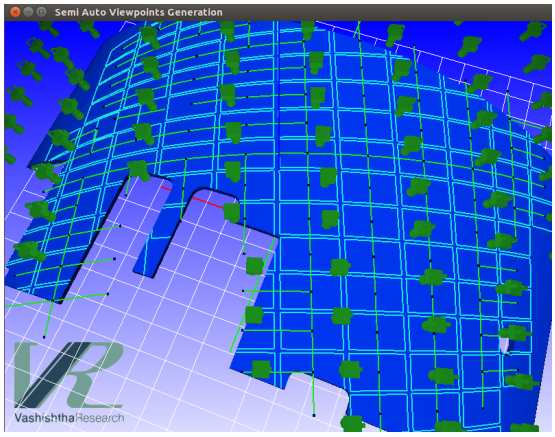
The start direction can have a significant impact on the node propagation on the surface. This can be seen in Figure 5.7b. We can select this in order to cover some types of surface more effectively.

5.2.4 Initiating Node propagation

The user can send the command to initiate the node propagation process (the algorithm for this is described in more detail in Section 5.4). This creates a preview set of viewpoints using the nodes generated by this process. The outlines corresponding to these viewpoints are displayed on the panel. If the user is unhappy with the preview, he can select another starting point and starting direction and start again. Figure 5.7a shows the output after preview is done.

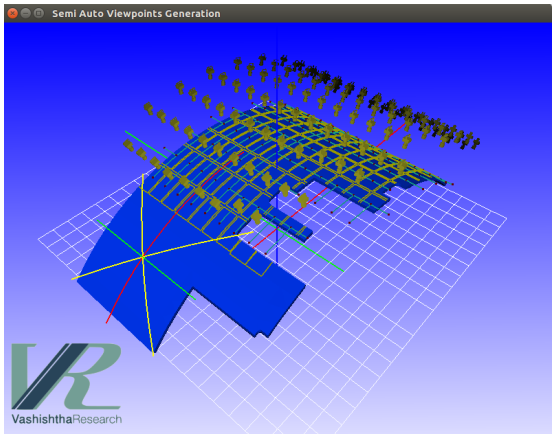


(a) Preview using the primary axes

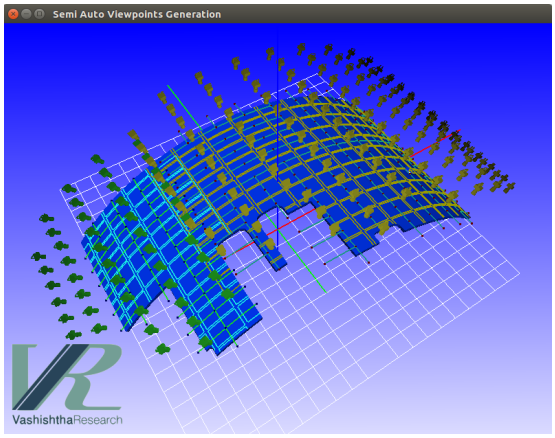


(b) Preview using the rotated axes

Figure 5.7: Previews generated by rotating the start direction



(a) Selecting the second starting point



(b) Generating preview from the second point

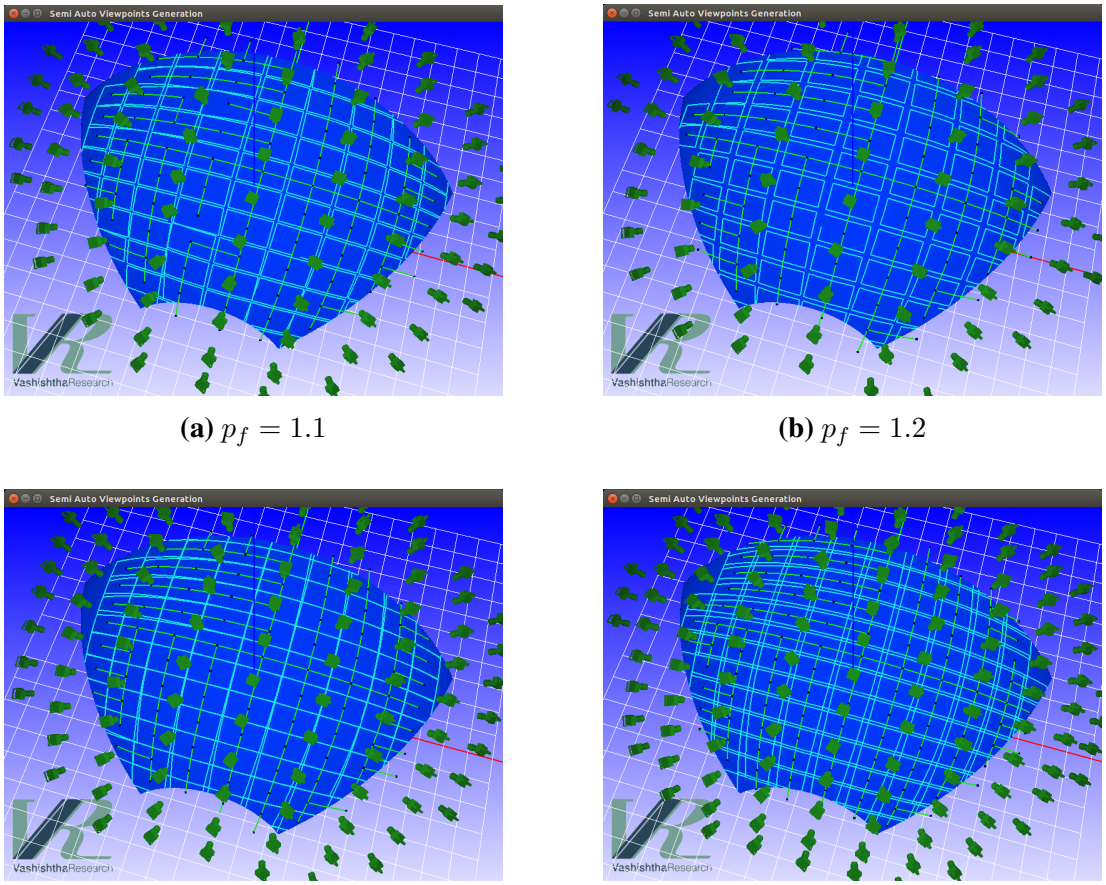
Figure 5.8: Saving the current set of viewpoints and generating preview from another point

5.2.5 Saving viewpoints

If the user is satisfied with the preview (Figure 5.7a), he can save the current set of viewpoints by sending the appropriate command. Upon doing so, these viewpoints are saved and they appear in a different color (ie. yellow) on the panel (the outlines and camera locations). After this, the user can select another point (Figure 5.8a and start the preview from there as well (see Figure 5.8b). He can save those viewpoints as well. This process can be continued until the entire panel surface is covered to the user's satisfaction.

5.2.6 Adjusting the Projection Factor

The projection factor p_f controls the distance between the two adjacent nodes (see section 5.4, Figure 5.13) and is used to control the amount of gap or overlap between the viewpoints. Generally speaking, there should be no gaps between the viewpoints for complete coverage over the panel. However for ease of understanding/preview, it is more helpful if there are some gaps between the outlines of the neighbouring viewpoints. We set the projection factor to a default of $p_f = 1.1$ for preview purpose and the user can adjust it by sending appropriate commands to change it in increments of 0.05.



(a) $p_f = 1.1$ **(b)** $p_f = 1.2$

(c) $p_f = 1.0$ **(d)** $p_f = 0.9$

Figure 5.9: Adjusting the Projection Factor

Figure 5.9 shows the impact of adjusting projection factor. The default value of $p_f = 1.1$ is shown in Figure 5.9a. On increasing it to $p_f = 1.2$ in Figure 5.9b, we can see that the gaps between the neighbouring viewpoint outlines have increased. Decreasing it to $p_f = 1.0$ (Figure 5.9c) reduces the gaps to almost non existent (in fact there will

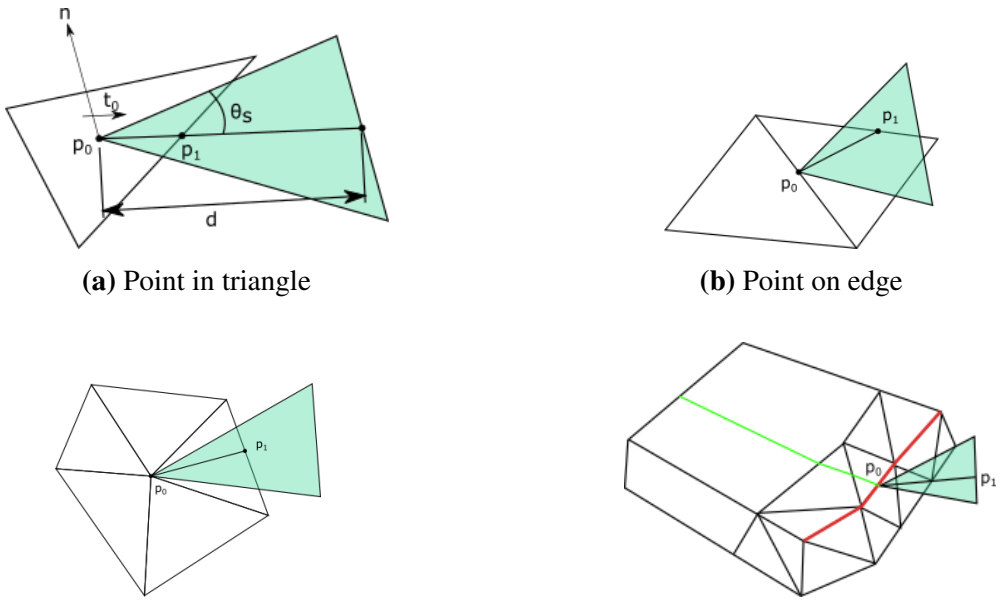
exactly be no gap for a developable surface) and decreasing it further to $p_f = 0.9$ (Figure 5.9d) creates overlap between the neighbouring viewpoint outlines, which is favourable for complete coverage. Unfortunately Figure 5.9d is difficult to interpret at first glance. The user is recommended to use the default value of $p_f = 1.1$ and then decrease it based on the requirement to generate the preview.

After the coverage plan (viewpoints) has been generated to the user's satisfaction, the travel path planning (Section 5.6) can be done and accordingly the G-code or Robot code for the manipulator to reach these viewpoints can be generated. Then the manipulator can be moved according to the code for doing the actual NDT inspection on the panel. These processes are covered in detail in Chapters 2 and 4 and are not repeated here.

5.3 Geodesic Paths on surface mesh

We now describe in detail the algorithm for finding the geodesic path of a given length L over the surface mesh given a starting point $\mathbf{p_0}$ and starting direction $\hat{t}_0$. This section is already described in [54] but we repeat it here for more clarity. The length L corresponds to the movement of the camera from node to node as detailed in the next section and it is typically the distance to the next near-rectangle, in the length or width direction. This algorithm is taken from [52] with some modifications but it is worth repeating here and going into the details of how it is implemented. Briefly, it is an iterative algorithm that takes an initial point and direction, determines the next point and direction using triangle intersection, and then repeats the process until the desired length is reached, or the boundary is reached, in which case it will extend the tangent over the edge up to the final length.

A pictorial explanation of the algorithm is shown in Figure 5.10. First it is determined whether $\mathbf{p_0}$ lies inside a triangle, on an edge between two triangles or on a vertex between several triangles. Accordingly the tessellation normal $\hat{n}$ is computed. After that the direction vector $\hat{t}_0$ is modified so as to make it perpendicular to $\hat{n}$ and then an intersection triangle is formed along the directions of $\hat{t}_0$ and $\hat{n}$, with a length of d which is the geodesic path distance remaining to be covered (see Figure 5.10a). The size of the intersection triangle is also determined by θ_s which is the search angle. The search angle is used as a check to decide if the boundary is reached (rather than a continuation of the curvature) and is normally set to 30 degrees. Then we check for the intersection of this triangle with the triangles that $\mathbf{p_0}$ is a part of. There will only be one intersection line in all three cases (as we can see in Figure 5.10a,5.10b.5.10c), or no intersection at all in the case where there is no next triangle or the next triangle is folded at an angle greater than θ_s as we can see



(a) Point in triangle

(b) Point on edge

(c) Point on vertex

(d) Point on boundary

Figure 5.10: Determining geodesic path

in Figure 5.10d. In both cases it means that the boundary has been reached, so we simply compute the final point with the distance remaining along the tangent vector at that point. The definition of 'boundary' here is a sharp discontinuity in the triangle mesh as we can see in 5.10d, and it is determined by selection of θ_s.

This algorithm also has two sub-functions - determining if p is a part of a triangle (including being on edges and vertices), and computing the tessellation normal n at that point. These two functions are described in Algorithm 5.1 and Algorithm 5.2. Triangle triangle intersection (computing the intersection line of two triangles) is described in [55] and not repeated here. The main algorithm is described in Algorithm 5.3.

Algorithm 5.1 is based on checking first whether the point is in the plane of the triangle, and then computing barycentric coordinates u,v to check whether it is inside the triangle or not. In the boundary cases where p lies on the edge or vertex, we would have atleast one of $u = 0, u = 1, v = 0, v = 1$ (for vertices two of them would be true). Algorithm 5.2 computes the tessellation normal as per [52]. If the point is inside the triangle, the normal is the normal vector of that triangle. If the point is on an edge or vertex, the normal is the average of the normal vectors of all the triangles that this point p is a part of.

The inequalities mentioned in these algorithms are usually evaluated in code with some tolerance built in, in order to account for numerical errors with floating point numbers (eg. $u \leq 1 + \epsilon$ rather than $u \leq 1$). Also, for a closed surface not having a well defined

Algorithm 5.1: Checking if point p is in triangle T

Input: Point $\mathbf{p}$, Triangle T
Output: Boolean, true if p is inside or on edges of T

Normal $\mathbf{N}$ of T is known
T has vertices as point A, B, C
$\theta_{PN} \leftarrow cos^{-1}((\mathbf{P}-\mathbf{A}) \cdot \mathbf{N}/\|\mathbf{P}-\mathbf{A}\|)$
/* If p is in the plane of the triangle, then $\mathbf{P}-\mathbf{A}$ is normal to $\mathbf{N}$ */
if $|\theta_{PN}| > 0$ **then**
    return false
end
/* Calculating barycentric coordinates u and v */
$u \leftarrow ((\mathbf{P}-\mathbf{A}) \times (\mathbf{C}-\mathbf{A}) \cdot \mathbf{N})/\|\mathbf{N}\|^2$
$v \leftarrow ((\mathbf{B}-\mathbf{A}) \times (\mathbf{P}-\mathbf{A}) \cdot \mathbf{N})/\|\mathbf{N}\|^2$
/* For p to be inside the triangle $0 \leq u, v \leq 1$ and $u+v \leq 1$ */
if $0 \leq u \leq 1$ *and* $0 \leq v \leq 1$ *and* $u+v \leq 1$ **then**
    return true
else
    return false
end

boundary such as a sphere or a torus, the geodesic path will simply wrap around the surface and continue on. The starting point in Algorithm 5.3 can be any point on the surface. In practice, we start from the origin and project a ray upward along the z axis. The intersection of this ray with the top of the mesh gives us our starting point.

The reason for using a geodesic path algorithm to generate a path of the surface, is that geodesics are very well defined curves that are easy to understand and model for parametric curved surfaces. However other techniques such as using slicing planes to generate a path could also be used and this is a topic which could be explored in future. Also the algorithm given in [52] is essentially an approximate way of computing a geodesic and this can also be improved. In the next section, we will see how the geodesic path algorithm will be used to generate the child nodes from parent nodes during node insertion.

Algorithm 5.2: Calculating tessellation normal of point p

Input: Triangle set S for point p where p is in T for all T $\in$ S
Output: unit normal vector $\mathbf{n}$

if *size(S) = 1* **then**
    /* The point lies inside triangle T */
    return $\mathbf{n}_T$
else
    /* The point lies on an edge or vertex */
    return avg($\mathbf{n}_T$) for all T $\in$ S
end

Algorithm 5.3: Generating Geodesic path sequence given input mesh, starting point, direction and desired path length

Input: Point $\mathbf{p}_0$, starting Direction $\mathbf{t}_0$, geodesic path length L, surface mesh M
Output: Geodesic path sequence SEQ (sequence of line segments), endpoint $\mathbf{p}_e$, endpoint normal vector $\mathbf{n}_e$, endpoint tangent vector $\mathbf{t}_e$

$\mathbf{p} \leftarrow \mathbf{p}_0$
$\mathbf{t} \leftarrow \mathbf{t}_0$
$l \leftarrow L$
// initialization of p, t vectors and the path length remaining
$SEQ \leftarrow \emptyset$ // Geodesic Path Sequence
while $l > 0$ **do**
    // Find set S of all triangles that contain p
    S $\leftarrow \emptyset$
    for *all triangles T* $\in$ *M* **do**
        if *isPointInTriangle(*$\mathbf{p}$*, T)* **then**
            $S \leftarrow S + T$ // adding T to the set S
    // Calculation of normal vector and also change tangent vector accordingly
    $\mathbf{n} \leftarrow calculateNormal(S)$
    $\mathbf{b} \leftarrow (\mathbf{n} \times \mathbf{t})/\|\mathbf{n} \times \mathbf{t}\|$
    $\mathbf{t} \leftarrow (\mathbf{b} \times \mathbf{n})/\|\mathbf{b} \times \mathbf{n}\|$

    // Intersection triangle
    Create a triangle I with vertices
    $\mathbf{p}, \mathbf{p} + l * \mathbf{t} + l * \tan\theta_s * \mathbf{n}, \mathbf{p} + l * \mathbf{t} - l * \tan\theta_s * \mathbf{n}$
    $ls \leftarrow$ intersection of triangle I with all triangles in S (there will only be one intersection line, or none)

if *no Intersection found* **then**

// Boundary has been reached, compute last segment

$\mathbf{p}_1 = \mathbf{p} + l * \mathbf{t}$

$\mathbf{t}_1 = \mathbf{t}$

$ls \leftarrow$ Line segment $\mathbf{pp_1}$

$SEQ \leftarrow SEQ + ls$ // Adding line segment ls to SEQ

$l = 0$

else

// Place line segment ls into the sequence

$\mathbf{p}_1 \leftarrow$ other point of ls (p is one point)

$\mathbf{t}_1 = (\mathbf{p}_1 - \mathbf{p})/\|\mathbf{p}_1 - \mathbf{p}\|$

$SEQ \leftarrow SEQ + ls$ // Adding line segment ls to SEQ

$l = l - \|\mathbf{pp_1}\|$

// Update p and t vectors for next loop

$\mathbf{p} \leftarrow \mathbf{p}_1$

$\mathbf{t} \leftarrow \mathbf{t}_1$

// After the loop ends, assign the outputs

$\mathbf{p}_e \leftarrow \mathbf{p}$

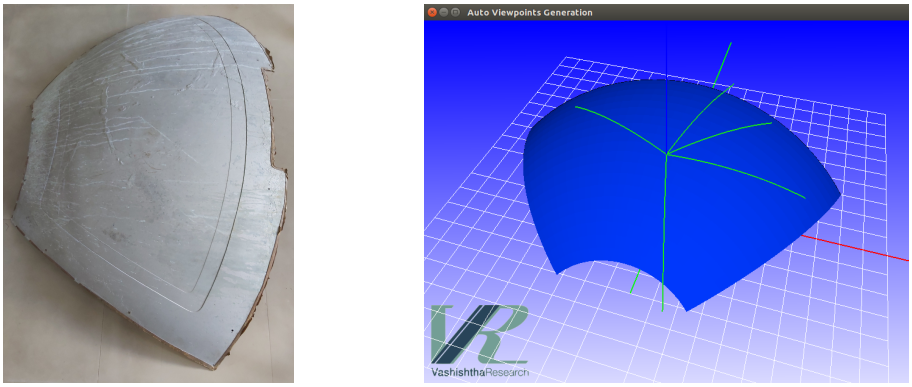
$\mathbf{t}_e \leftarrow \mathbf{t}$

$\mathbf{n}_e \leftarrow \mathbf{n}$

// SEQ has already been updated with all the line segments

return $SEQ, \mathbf{p}_e, \mathbf{n}_e, \mathbf{t}_e$

An example output is shown in Figure 5.11b. A panel that is part of a launch vehicle propellant tank (it is an octant of a sphere with its top cut off) is shown in Figure 5.11a. We compute a set of geodesic paths on the 3D mesh model of this panel, starting from the origin point and length of 0.65m, with directions of 5°, 45°, 75°, 155°, -75°.



(a) Panel for inspection **(b)** Geodesic paths on 3D model of panel

Figure 5.11: Geodesic path on surfaces

We note that the curve follows the surface, and it either completes within the mesh or upon reaching the boundary a last straight line is drawn off from it in the direction of the last direction vector up to the required length. In these cases, the end point of the geodesic path is outside the mesh.

5.4 Node insertion algorithm for determining viewpoints

We now detail our algorithm for inserting nodes ie. points on the surface from which viewpoints will be generated. These nodes are added in a tree-like fashion - first the initial node is expanded in four directions then its child nodes are expanded and so on until the node propagation stoppage criteria is met for all the nodes.

5.4.1 Nodes and Viewpoints

A new data structure is created for the node and its details are explained in Table 5.1.

Data	Data type	Explanation
$\mathbf{p}$	vector(3)	Position vector of the node in R3 (on Mesh)
$\mathbf{n}$	vector(3)	outward normal vector of the node
$\mathbf{t}$	vector(3)	tangent vector of the node
$\mathbf{b}$	vector(3)	binormal vector of the node
i	integer	X-coordinate of the node in the 2D space
j	integer	Y-coordinate of the node in the 2D space
active	boolean	Whether the node is active or not

Table 5.1: Node Data structure

We note that the position of the node is recorded in both 3D and 2D space. In 3D space, the node's position $\mathbf{p}$ is a point (x,y,z) on the mesh that is calculated by the geodesic path algorithm. In 2D space, the node's position is not in units of distance but as integer (i,j) which denotes the projection from the initial point in the horizontal (tangent) and vertical (binormal) directions. The initial point is at (0,0) and expanding to the right creates node at (1,0) and so on.

The normal, tangent and binormal $\mathbf{n}, \mathbf{t}, \mathbf{b}$ are unit vectors that form a right handed vector basis with its origin at the node (also called Frenet Serret reference frame) and the normal is oriented outwards from the mesh so that the view point $\mathbf{v}$ can be located at some offset distance d_0 looking down on at the node. The view camera's up direction vector is oriented along the binormal direction. Figure 5.12 describes the node and view camera.

Figure 5.12 also shows us the projection of the viewing frustum on the surface. As we have assumed a quasi-flat approximation, we can say that the projection on the surface is approximately that of the viewing frustum. The horizontal projection p_h and vertical projection p_v can be thus calculated by the following formulas (Equation 5.1).

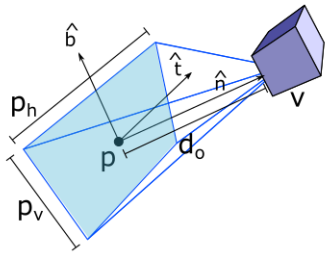
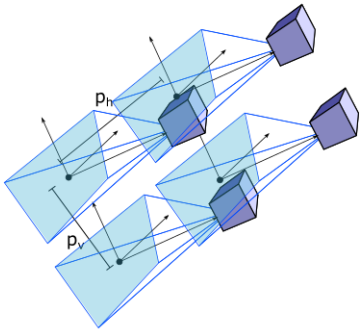
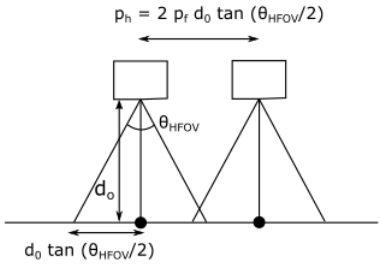


Figure 5.12: Node point and view camera

$$p_h = 2p_f d_0 tan(\theta_{HFOV}/2)$$
$$p_v = 2p_f d_0 tan(\theta_{VFOV}/2) \tag{5.1}$$



(a) Multiple view points corresponding to neighbouring nodes



(b) Section view of horizontal projection

Figure 5.13: Viewpoint projection on surface

This can be understood by referring to Figure 5.13. We see in Figure 5.13b that on an ideal flat surface, the viewpoint can see a frustum of length $2d_0tan(\theta_{HFOV}/2)$ in the horizontal direction (and $2d_0tan(\theta_{VFOV}/2)$ in vertical direction. Since we are using a quasi-flat approximation, we will assume that we need to move the same distance away to create another viewpoint with no overlap or gap. We add a projection factor p_f, which will decide whether the viewing areas have gaps in between them or overlaps. If $p_f > 1$, there will be gaps (because we will be inserting nodes further away than their actual projection on the surface), and if $p_f < 1$, there will be overlaps (Figure 5.13a shows gaps and Figure

5.13b shows overlaps). In practice we want some overlaps between images but this makes the visualization of the coverage algorithm very confusing, hence in the present work we set $p_f = 1.1$ in order to visualize each viewing area (corresponding to the node and view point) separately on the surface, as in Figure 5.16a and so on.

The lengths p_h and p_v are important because these are our length inputs (L) to the geodesic path algorithm. We will find the child nodes from the parent node by moving by p_h in horizontal (tangent) direction and p_v in vertical (binormal) direction along the surface.

We also have two variables θ_{HFOV} and θ_{VFOV}, these denote the horizontal and vertical field of view of the camera viewing frustum which we can see in Figure 5.12. These variables are not independent but related through the aspect ratio using Equation 5.2

$$\theta_{HFOV} = \theta_{VFOV} * WIDTH/HEIGHT \tag{5.2}$$

Where WIDTH and HEIGHT correspond to the sensor's image size and are measured in pixels. For the present work, we use the parameters for an ELP IP camera with the camera's parameters as per Table 5.2. In this way, our virtual camera can be modelled for use in our algorithm, corresponding to a real camera. Note that focal length is not an independent parameter (it depends on sensor width, height and field of view) and is not modelled here. The offset distance d_0 is a constant for all the viewpoints as we want all the images taken to be consistent and roughly the same size, and this distance is fixed at 30 cm for our algorithm. This gives us viewing area of about 16.5 cm x 9.5 cm on the surface.

Parameter	Value	Units
WIDTH	1280	Pixels
HEIGHT	738	Pixels
θ_{VFOV}	18	Degrees

Table 5.2: Camera parameters

The nodes are inserted adjacent to one another until the node propagation stoppage criteria is reached for every node. The details of the algorithm are discussed in section 5.4.3 and the stoppage criteria are described in the next section.

5.4.2 Node propagation stoppage criteria

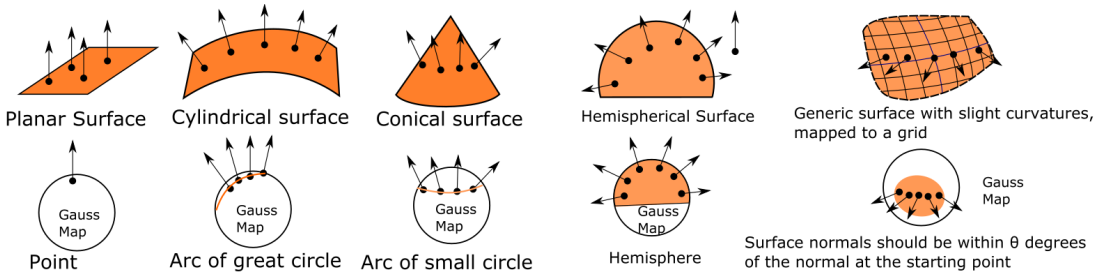
In this section we will discuss the criteria of stopping the node propagation process. We can observe that since the geodesic lines could draw close together or move apart on a highly

curved surface like a hemisphere, too much overlap or separation of adjacent projections (bad behaviour) could occur. Thus, we introduce criteria for node stoppage to limit the view propagation so that the neighbouring nodes do not converge or diverge from each other too much. We propose and explore two approaches here - the 'indirect' approach which relies on normal vectors not diverging too much from the starting point, and the 'direct' approach (which relies on managing the distance between neighbouring viewpoints). Both the approaches are explained in the next section and the Results section shows the output from both methods so that we can compare them for many different use-cases.

As we are not partitioning the surface, when we use a stopping criterion, only a portion of the surface may be covered before stopping. So we need to repeat the process from a yet to be covered portion. Hence in addition to the primary stopping criteria which we discussed in the previous paragraph, we need to also ensure that the current node propagation process does not move into an already existing set of nodes.

5.4.2.1 Indirect method (using Gauss Maps)

The idea behind this method comes from the concept of Gauss Maps in Differential Geometry (Refer, eg. [56]). The Gauss map is a map that maps any surface to a unit sphere, which transforms the unit normal at every point on the surface to the corresponding point on the unit sphere. If we draw the Gauss map for developable surfaces ie. Plane, Cylinder and Cone, we observe that the normals lie at the same point, along a great circle, or along a small circle respectively (Figure 5.14a). On the other hand, for a highly non-developable surface such as a hemisphere, the Gauss map occupies a large area on the unit sphere. In fact, the Gauss map of a hemisphere is also a hemisphere (Figure 5.14b).



(a) Gauss map for developable surface **(b)** Gauss map for non-developable surface

Figure 5.14: Gauss maps of surfaces

From the above observations, we define the concept of a near-developable patch on the surface as being an area where the normals diverge from each other by a maximum of some

angle (say 60 degrees). In this region, we can still fit a planar grid onto the surface with no significant deviation. Thus the stoppage criteria for the direct approach is arrived at. If the normal vector of the current node being inserted is more than a certain angle θ_N with respect to that of the starting node, we will stop the node propagation at this node.

The process is detailed in Algorithm 5.4.

Algorithm 5.4: Indirect method - Stoppage criteria

Input: Unit normal vector at current point $\mathbf{n}$, Unit normal vector at the start point $\mathbf{n}_0$
Output: Boolean, true if node propagation should be continued
$\theta \leftarrow cos^{-1}(\mathbf{n} \cdot \mathbf{n}_0)$
if $\theta > \theta_N$ **then**
    return false
else
    return true
end

This method has a drawback that developable surfaces such as a cylinder will still stop the node propagation at that value of normal angle divergence θ_N, even though the node propagation can still continue without any distortion (as the surface is developable). We will then need to take viewpoints from other starting points as well after saving the current set of viewpoints. This can be fixed to some extent by considering angular deviations in two directions, but as that also has some drawbacks, we do not explore it here. Instead we use an alternative technique named 'Direct method' explained in the next section.

5.4.2.2 Direct method

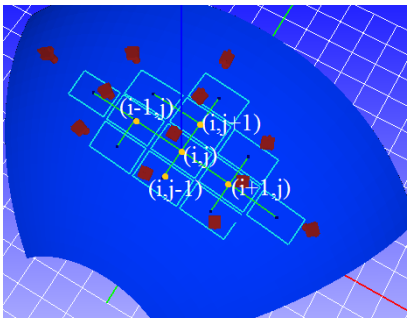


Figure 5.15: Node stoppage criteria (Direct)

The idea behind this method is to check for convergence and divergence of the current node being inserted with respect to that of its neighbouring nodes (if they already

exist). In particular, if the node is located at (i,j) coordinates with respect to the start point (See section 5.4, Table 5.1), we check the distance to its neighbouring nodes at (i+1,j),(i-1,j),(i,j+1),(i,j-1) as seen in Figure 5.15. We compare the distance between current node (i,j) and nodes at (i+1,j),(i-1,j) with the horizontal projection on panel p_H (section 5.4, Figure 5.13) and the distance between current node (i,j) and nodes at (i,j+1),(i,j-1) to the vertical projection on panel p_V. If either of the two distances is less than or more than its prespecified limit, the propagation will be stopped.

The complete algorithm for implementing this criteria is given as follows (Algorithm 5.5).

Algorithm 5.5: Direct method - Stoppage criteria

Input: current point position $\mathbf{p}$, Coordinates of current position (i,j), current set of nodes S

Output: Boolean, true if node propagation should be continued

$min_H \leftarrow 0.9p_H$ /* Minimum Horizontal Distance between nodes */

$max_H \leftarrow 1.2p_H$ /* Maximum Horizontal Distance between nodes */

$min_V \leftarrow 0.9p_V$ /* Minimum Vertical Distance between nodes */

$min_V \leftarrow 1.2p_V$ /* Maximum Vertical Distance between nodes */

for *all nodes* $N \in S$ **do**

  $dist \leftarrow Distance(N.\mathbf{p}, \mathbf{p})$ /* Euclidean Distance between node N's position and current point position */

  if *N.i = i and (N.j = j+1 or N.j = j-1) and* ($dist < min_V$ *or* $dist > max_V$) **then**

    return false

  end

  if *N.j = j and (N.i = i+1 or N.i = i-1) and* ($dist < min_H$ *or* $dist > max_H$) **then**

    return false

  end

end

return true

5.4.2.3 Checking if Near Existing Saved Nodes

The final check for node propagation stoppage is to see whether the current node being inserted is close to any saved nodes or not. This is necessary to ensure that the node propagation is stopped if it 'runs into' already existing nodes that were saved from a previous coverage planning process initiated from another start location.

This criteria is detailed in Algorithm 5.6.

Algorithm 5.6: Checking near saved nodes - Stoppage criteria

```
Input: current point position p, set of saved nodes S
Output: Boolean, true if node propagation should be continued
min_S ← p_V /* Minimum Separation allowed between nodes          */
for all nodes N ∈ S do
    dist ← Distance(N.p, p) /* Euclidean Distance between node N's
        position and current point position                      */
    if dist < min_S then
        return false
    end
end
return true
```

The value of p_V is chosen conservatively as the minimum separation allowed between nodes, as there will always be an overlap if the node separation is less than p_V (no matter which direction these viewpoints are angled, assuming $p_V < p_H$). Note that the Euclidean distance is used here with the assumption that the radius of curvature is large compared to the projection of viewing frustum on the surface.

5.4.3 Node Insertion Algorithm

We now detail our node insertion algorithm. A starting point is chosen on the mesh, its normal is determined and a direction is given as its starting tangent vector (This is given as an input by the user taking into consideration the panel shape). A set of nodes is created which is initially empty. We now insert the starting node into the set, with (i,j) as (0,0) and binormal vector can be calculated from the normal and tangent vectors. Whenever we insert a node, it is initially set to active (an active node is one whose neighbours have not been fully expanded). We now continue adding nodes until there are no more active nodes. If an active node is found, we create new nodes along right, up, left and down directions using the geodesic path algorithm, taking care to ensure that there isn't already a node

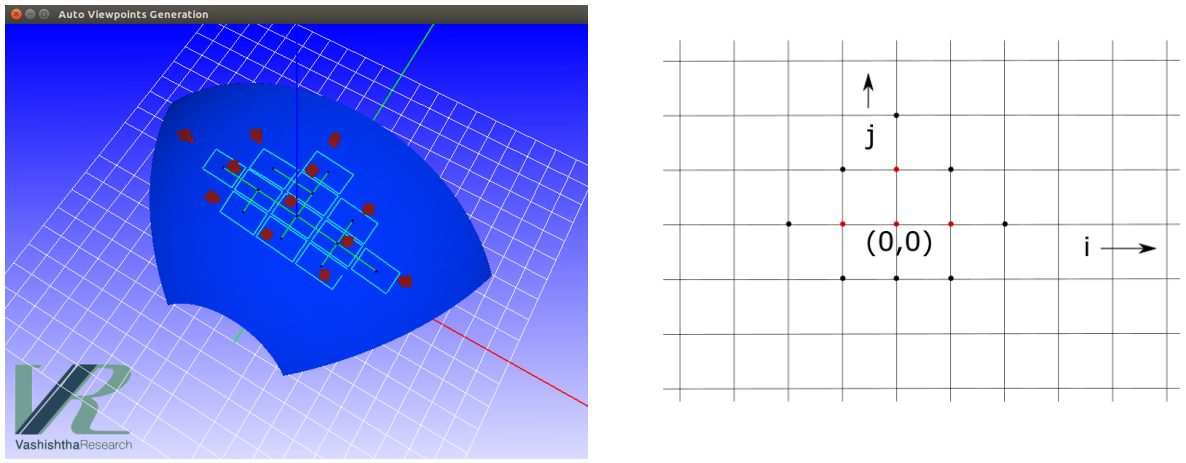
present at that position. The direction vector input depends on tangent or binormal vectors (for left/right or up/down) and the length input is p_h or p_v as described in Figure 5.13a.

We insert new node at that point with the appropriate position, normal, tangent, binormal vector and i,j set as active except in the case where the Node stoppage criteria is satisfied. in which case we make it inactive. After all the expansion possible is completed we set this node to inactive. We repeat this until all the nodes have become inactive which leads to a complete coverage of the area.

Node propagation stoppage criteria is explained in detail in Section 5.4.2 and is briefly recapped here. The criteria is that any one of the following is true:

- The newly inserted node lies outside the mesh
- The newly inserted node is too close to an existing saved node
- Either the 'Indirect method' or 'Direct method' is used to check the divergence of the nodes on the curved surface

Node insertion in progress is shown in Figure 5.16. Currently 12 nodes have been inserted so far. Figure 5.16a shows the position of the nodes on the surface of the mesh, the geodesic paths taken to get there and also the corresponding view points and intersection of viewing frustum with mesh surface. Figure 5.16b shows the same node insertion progress in the 2D space of (i,j). The active nodes are shown in black and inactive nodes (which have been fully expanded) are shown in red.



(a) Node insertion on 3D surface **(b)** Node insertion 2D space

Figure 5.16: Node insertion algorithm

The complete algorithm for node insertion is given in Algorithm 5.7.

Algorithm 5.7: Node Insertion Algorithm

Input: Starting Point $\mathbf{p}$, Starting Tangent Direction $\mathbf{t}$, Surface Mesh M

Output: Set S of all nodes inserted

S $\leftarrow \emptyset$

`/* Creating first node. Normal can be computed for point p on M and binormal vector b can be computed from t and n */`

Create node N with position $\mathbf{p}$, tangent $\mathbf{t}$, normal $\mathbf{n}$, (i,j) as (0,0) set Active

S $\leftarrow$ S + N

while *atleast one active node in S* **do**

- Find N, an active node in S
- `// Placing neighbouring nodes`
- **if** *No node at (N.i+1, N.j)* **then**
  - $SEQ, \mathbf{p}_e, \mathbf{n}_e, \mathbf{t}_e$ = geodesicPathSequence(N.$\mathbf{p}$, N.$\mathbf{t}$, p_h, M)
  - Create node R with position $\mathbf{p}_e$, tangent $\mathbf{t}_e$, normal $\mathbf{n}_e$, (i,j) as (N.i+1,N.j), set Active
  - Set R inactive if Node Propagation Stoppage Criteria is satisfied
  - S $\leftarrow$ S + R
- **if** *No node at (N.i, N.j+1)* **then**
  - $SEQ, \mathbf{p}_e, \mathbf{n}_e, \mathbf{t}_e$ = geodesicPathSequence(N.$\mathbf{p}$, N.$\mathbf{b}$, p_v, M)
  - Create node U with position $\mathbf{p}_e$, tangent $\mathbf{t}_e \times \mathbf{n}_e$, normal $\mathbf{n}_e$, (i,j) as (N.i,N.j+1), set Active
  - Set U inactive if Node Propagation Stoppage Criteria is satisfied
  - S $\leftarrow$ S + U
- **if** *No node at (N.i-1, N.j)* **then**
  - $SEQ, \mathbf{p}_e, \mathbf{n}_e, \mathbf{t}_e$ = geodesicPathSequence(N.$\mathbf{p}$, -N.$\mathbf{t}$, p_h, M)
  - Create node L with position $\mathbf{p}_e$, tangent $-\mathbf{t}_e$, normal $\mathbf{n}_e$, (i,j) as (N.i-1,N.j), set Active
  - Set L inactive if Node Propagation Stoppage Criteria is satisfied
  - S $\leftarrow$ S + L
- **if** *No node at (N.i, N.j-1)* **then**
  - $SEQ, \mathbf{p}_e, \mathbf{n}_e, \mathbf{t}_e$ = geodesicPathSequence(N.$\mathbf{p}$, -N.$\mathbf{b}$, p_v, M)
  - Create node D with position $\mathbf{p}_e$, tangent $-\mathbf{t}_e \times \mathbf{n}_e$, normal $\mathbf{n}_e$, (i,j) as (N.i,N.j-1), set Active
  - Set D inactive if Node Propagation Stoppage Criteria is satisfied
  - S $\leftarrow$ S + D
- Set N as inactive

return S

5.5 Results

We run the coverage path planning (CPP) node insertion algorithm for a variety of surfaces. Two of them are panels whose hardware is available to us from the Aerospace Industry (ISRO). One is of an octant-shape (part of a propellant tank) and a cylindrical shape with cutouts, part of a launch vehicle structure. These two are listed as Doubly Curved Convex and Singly Curved Convex shape respectively. By rotating them upside down, we also obtain a Doubly Curved Concave and Singly Curved Concave shape respectively. This is also a realistic use case in the industry as we need to do this kind of NDT inspection (Thermography) for both sides of the panel. Three more panel shapes are considered here in mesh form (although we have no physical panels of that shape available to us) - A highly Doubly Curved Convex surface (hemisphere), Highly single curved in one direction and slightly curved in the other (nose cone shape), and a Concave-Convex structure (saddle shape). These examples are used to illustrate extensively the shapes of panels used in Aerospace application (see, eg. Figure 5.17). Flat or near flat shapes (eg. airfoils, wings, empennage) and full cylinders (eg. fuselage, engines) are not listed as they are basic variations of listed shapes. Note that all results here are produced with default value of $p_f = 1.1$. See Section 5.2.6 to see how the results change based on the projection factor.

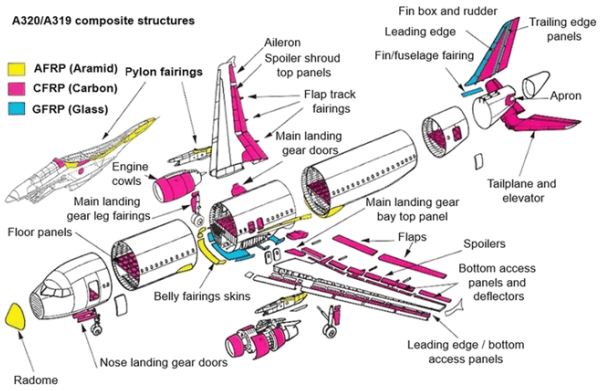


Figure 5.17: Aircraft panels

The complete list of panel shapes is thus:

1. Singly Curved Convex: Partial cylinder panel with cutouts, part of a launch vehicle.
2. Singly Curved Concave: The same panel as Singly Curved Convex but rotated 180 degrees about Y-axis
3. Doubly Curved Convex: Octant shape, part of a propellant tank.

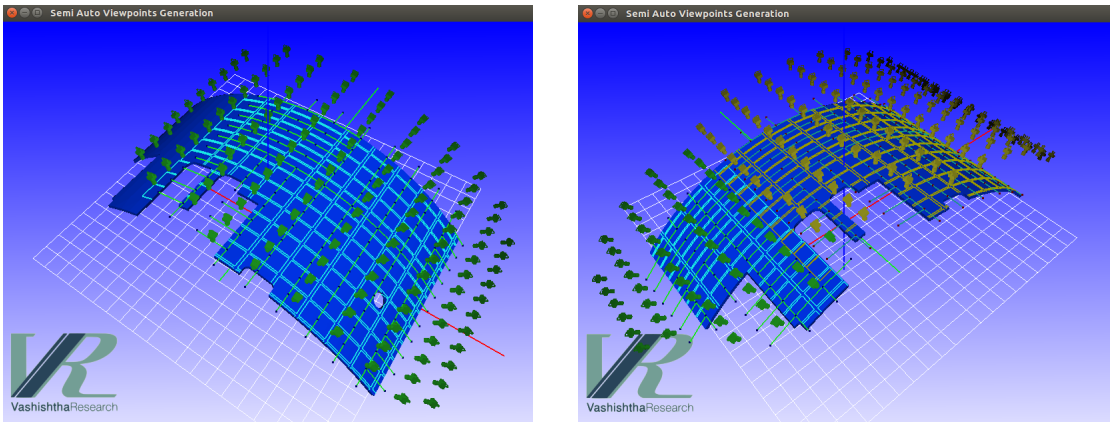
4. Doubly Curved Concave: The same panel as Doubly Curved Convex but rotated 180 degrees about Y axis
5. Doubly Curved Convex highly curved: Hemispherical Shape
6. Nose Cone: Highly curved in one direction and slightly curved in the other, with a singularity at the tip
7. Concave Convex: Saddle shape

5.5.1 Singly Curved Convex Shape

This panel is part of a launch vehicle body. The cylindrical surface has 10194 triangles and an upper surface area of 22529 cm^2.

5.5.1.1 Indirect method

We select two starting points using the Indirect method to cover the entire surface.



(a) Singly Curved Convex (first start point) **(b)** Singly Curved Convex (second start point)

Figure 5.18: Node insertion Results - Singly Curved Convex surface, Indirect method

5.5.1.2 Direct method

The Direct method can cover the surface with just one starting point.

We observe that both the Indirect method (Figure 5.18) and Direct method (Figure 5.19) give excellent results, although the Indirect method requires two starting points owing to the limitation described in Section 5.4.2.1. The number of nodes generated in the Indirect method is 172 compared to 161 using the Direct method. This is because of overlap in the area covered by the first and second starting points as seen in Figure 5.18b.

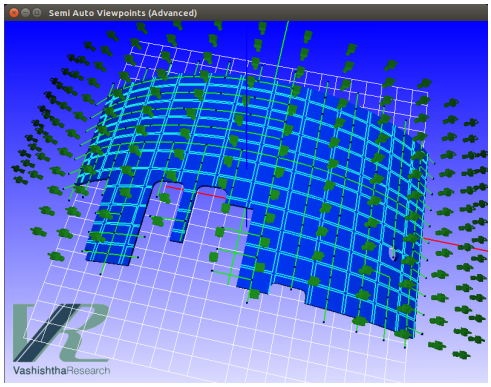
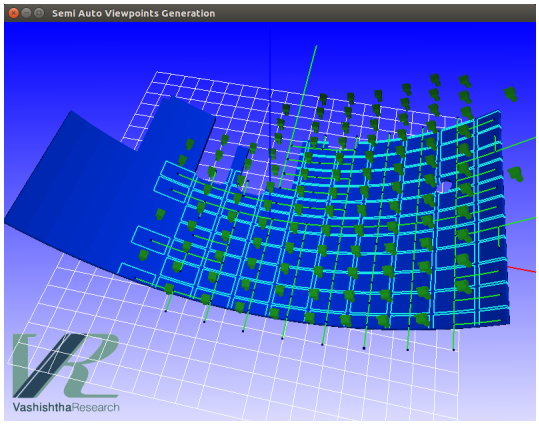


Figure 5.19: Node insertion Results - Singly Curved Convex surface, Direct method

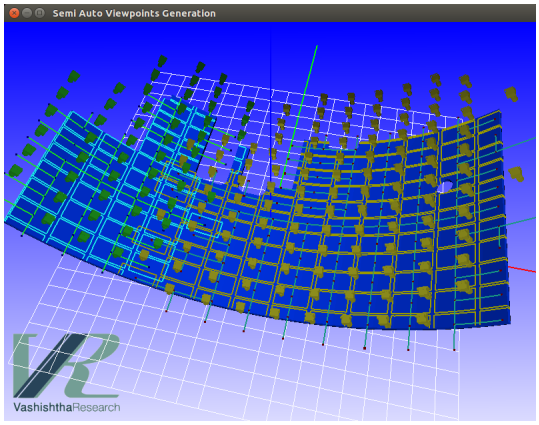
5.5.2 Singly Curved Concave

5.5.2.1 Indirect method

Two starting points were used to cover the surface, as seen in Figure 5.20.



(a) Singly Curved Concave (first start point)



(b) Singly Curved Concave (second start point)

Figure 5.20: Node insertion Results - Singly Curved Concave surface, Indirect method

5.5.2.2 Direct method

As in the previous case, we observe that both the Indirect method (Figure 5.20) and Direct method (Figure 5.21) give excellent results, as this is the same panel but rotated to make it upside down. The Indirect method requires two starting points just as in the previous case. The number of nodes generated in the Indirect method here is 164 - owing to better selection of starting points, we have less overlap.

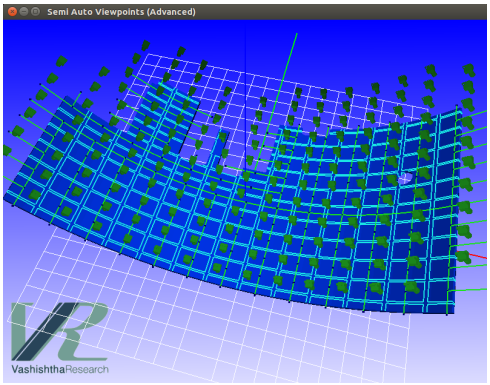


Figure 5.21: Node insertion Results - Singly Curved Concave surface, Direct method

Given the viewing frustum area of 156.75 cm^2, we would expect 143 nodes for the cylindrical panel (Singly Curved Concave/Singly Curved Convex) surface coverage if the area was to be covered exactly. We see that we get more nodes than expected because many of these are near the boundary and cover very little of the surface. This also applies to the previous case of Singly Curved Convex surface as it is the same panel.

Both the Singly Curved Convex and Concave surfaces are equivalent to a curved plane, hence there is no divergence of geodesic lines observed.

5.5.3 Doubly Curved Convex

This surface is an octant of a sphere (part of a propellant tank). It has 3046 triangles and an upper surface area of 13644 cm^2.

5.5.3.1 Indirect method

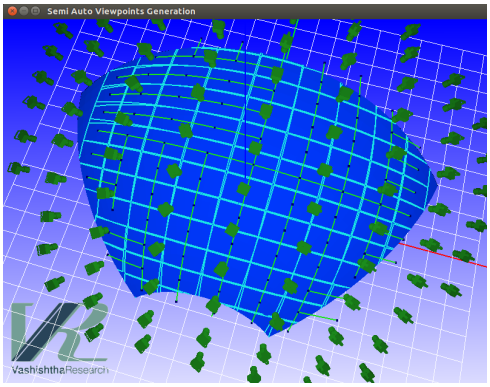


Figure 5.22: Node insertion Results - Doubly Curved Convex surface, Indirect method

The Indirect method does not cover the entire surface, leaving uncovered areas towards two of the corners. This is because of the limitation as described in Section 5.4.2.1. We omit multiple start points for full coverage here for simplicity.

5.5.3.2 Direct method

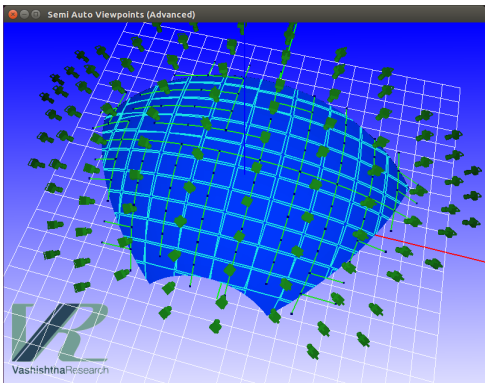


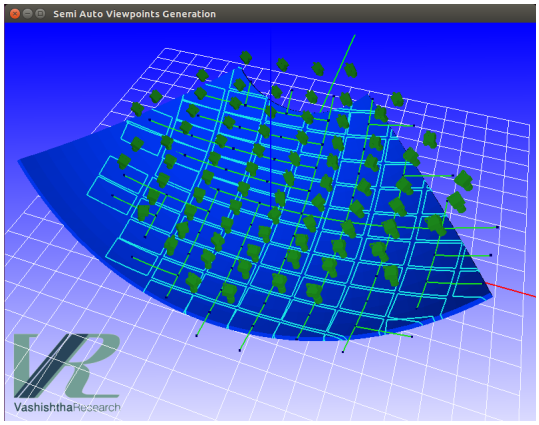
Figure 5.23: Node insertion Results - Doubly Curved Convex surface, Direct method

We observe that both the Indirect method (Figure 5.22) and Direct method (Figure 5.23) give excellent results. Note that the geodesic lines start drawing close together (converging) towards the edges of the panel. This is due to the inherent Doubly Curved Convex nature of the surface.

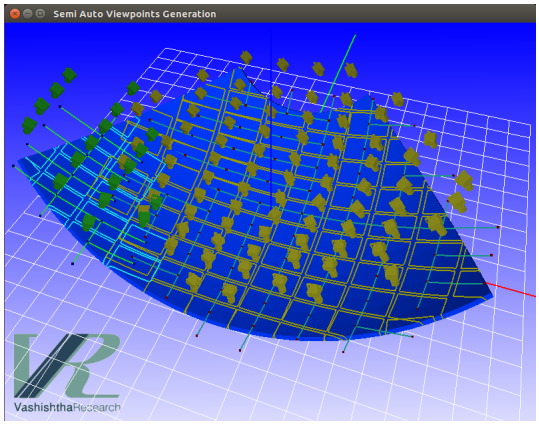
5.5.4 Doubly Curved Concave

5.5.4.1 Indirect method

The Indirect method requires two starting points for full coverage (Figure 5.24).



(a) Doubly Curved Concave (first start point)



(b) Doubly Curved Concave (second start point)

Figure 5.24: Node insertion Results - Doubly Curved Concave surface, Indirect method

5.5.4.2 Direct method

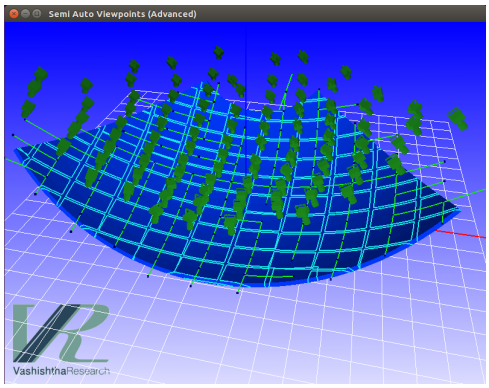


Figure 5.25: Node insertion Results - Doubly Curved Concave surface, Direct method

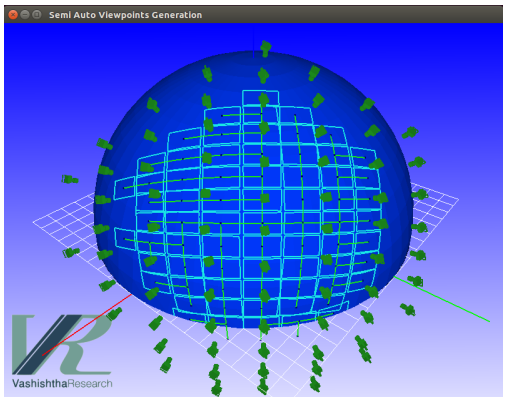
As in the previous case, we observe that both the Indirect method (Figure 5.24) and Direct method (Figure 5.25) give excellent results, as this is the same panel but rotated to make it upside down. The number of nodes generated in the Indirect method here is 97, and in the Direct method is 95, as compared to 87 nodes that would be expected for perfect coverage using the surface area of the octant shaped panel. This is because of overlaps and wasted space at the boundaries.

We also note that in this case, the viewpoints diverge (or grow farther from each other) as we go from the centre to the boundary. This is to be expected from the nature of geodesics on a Doubly Curved Concave surface. It is in contrast to the converging nature observed in the case of the Doubly Curved Convex surface.

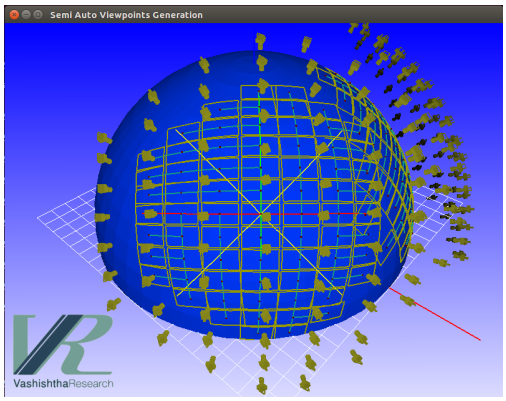
5.5.5 Highly Double Curved - Hemispherical Surface

5.5.5.1 Indirect method

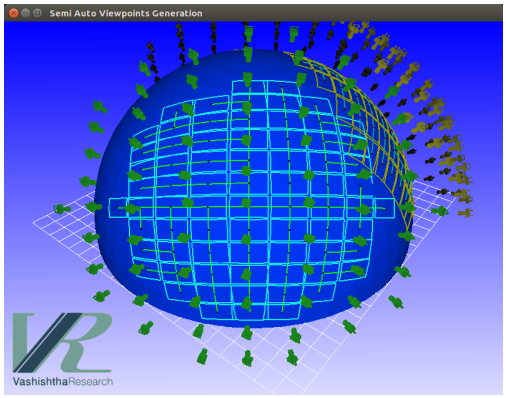
For the coverage of the hemisphere using the Indirect method, we need quite a few starting points. We have four main starting points (Start point 1 to 4) corresponding to the four quadrants, then a 5th one for the top, then a 6th one to cover some gaps in between and so on (See Figure 5.26). The complete coverage needed a total of 404 nodes using this method.



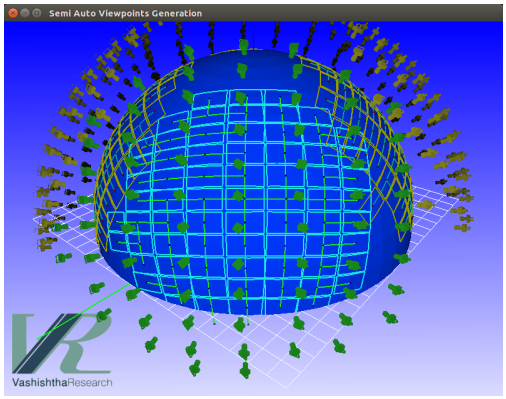
(a) Start point 1



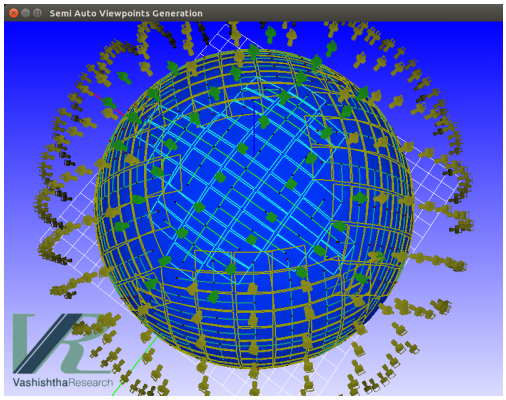
(b) Start point 2



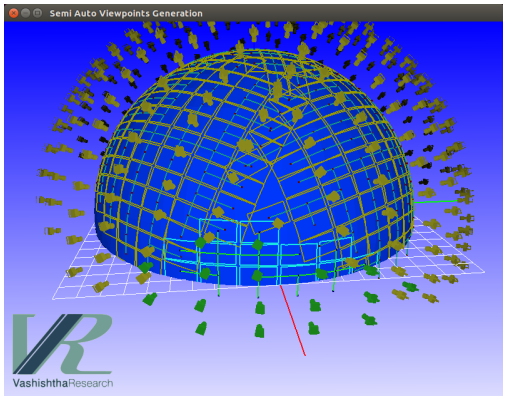
(c) Start point 3



(d) Start point 4



(e) Start point 5

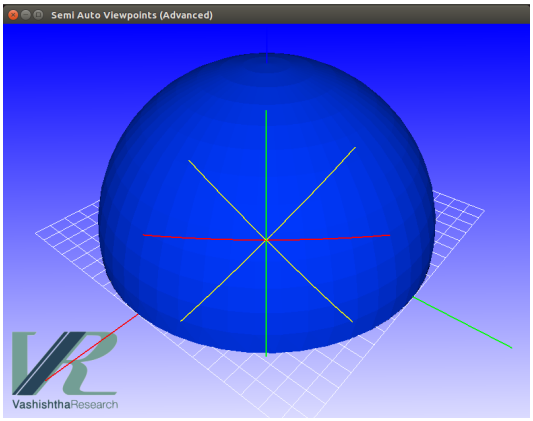


(f) Start point 6

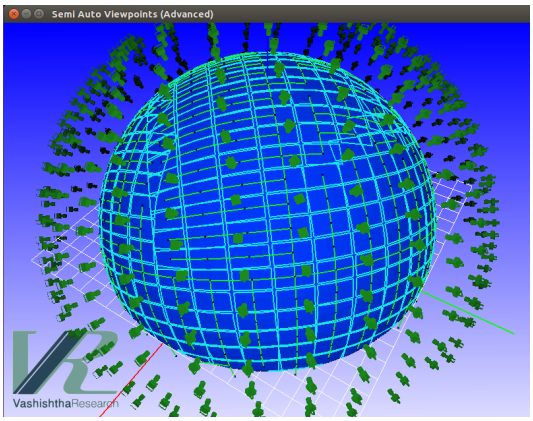
Figure 5.26: Hemisphere Coverage - Indirect method

5.5.5.2 Direct method

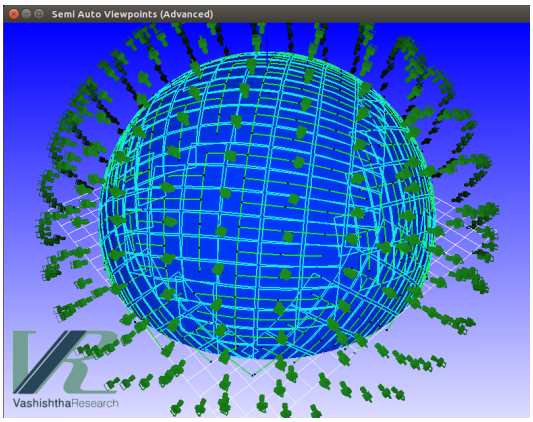
The advantage of the Direct method in this case is that we can generate the entire coverage just in one click given a starting pont (Figure 5.27a). In this case, what we observe is that the node propagation gets stopped at a certain region, but then it gets covered anyway by nodes coming from a different direction.



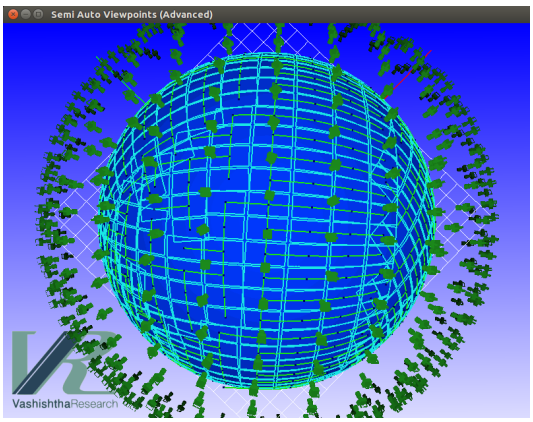
(a) Starting point for the Direct method



(b) Coverage of Hemisphere



(c) Coverage alternate viewpoint



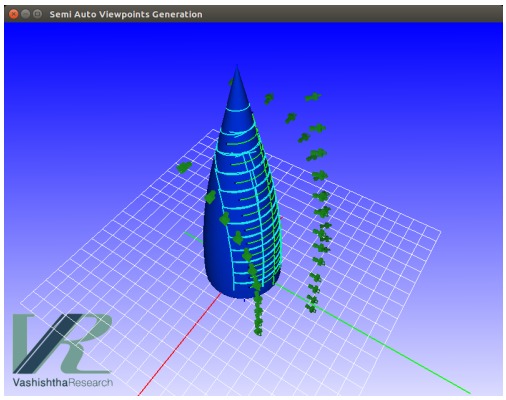
(d) Coverage alternate viewpoint

Figure 5.27: Hemisphere Coverage - Direct method

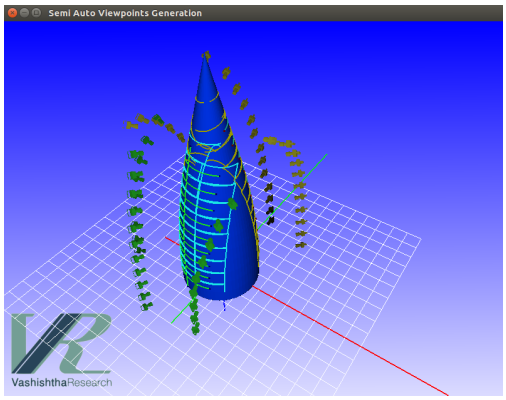
We also observe lots of overlaps (Figure 5.27) looking at the coverage from different angles by rotating the 3D view. These overlaps are caused when nodes from different directions of propagation bump into each other. Also, the notion of 'fitting a grid to the surface' is essentially lost in this case because some distance after the start point, the nodes get propagated in ways that are not very intuitive. In the Indirect method, by comparison, we can easily see how a grid is fitted to small patches of the sphere without much distortion. A total number of 421 nodes are generated using the Direct method to cover the surface.

5.5.6 Nose Cone Surface

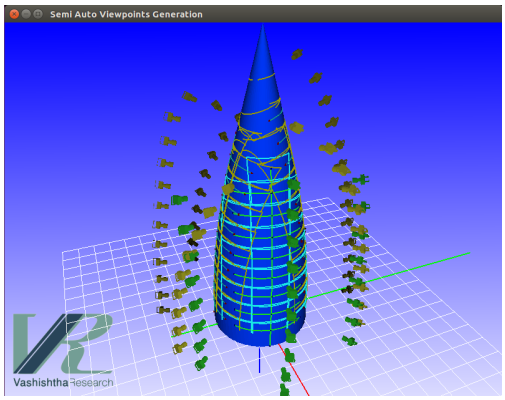
5.5.6.1 Indirect method



(a) Start point 1



(b) Start point 2



(c) Start point 3

Figure 5.28: Nose Cone Coverage - Indirect method

Three starting points were used for coverage of the Nose-cone shape (Figure 5.28).

5.5.6.2 Direct method

We observe that both the 'Indirect method' and 'Direct method' perform poorly on the Nose-Cone surface. It can be seen that geodesic lines 'wrap around' the top point of the nose cone causing weird and non-intuitive node propagation. Moreover the coverage area produced by the viewing frustum on the surface is no longer near-rectangular near the tip of the cone. In this case, effective coverage is hard to evaluate. The Indirect method works better and more intuitively at least near the base of the cone. A total of 97 nodes were

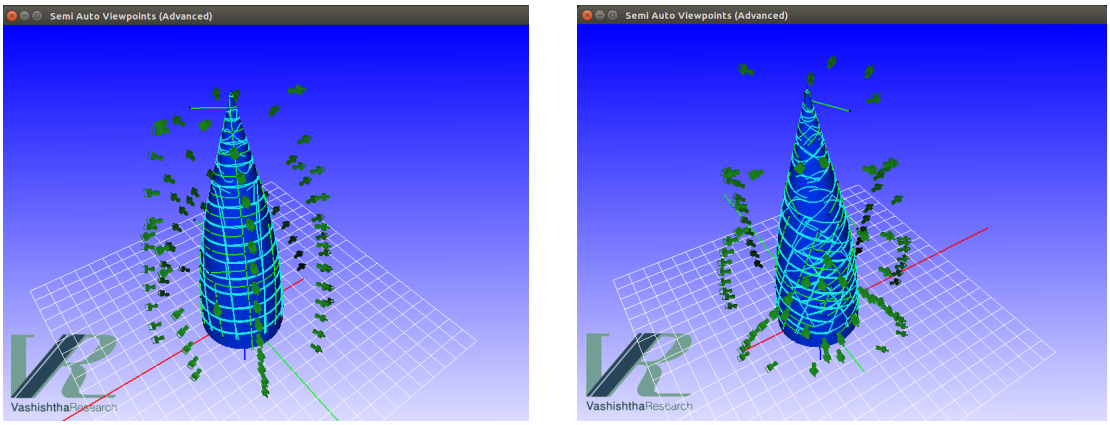


Figure 5.29: Coverage of Nose Cone surface, Direct method (Viewed from two angles)

generated by the Direct method as compared to 90 nodes by the Indirect method for this surface.

5.5.7 Concave Convex surface - Saddle Shape

5.5.7.1 Indirect method

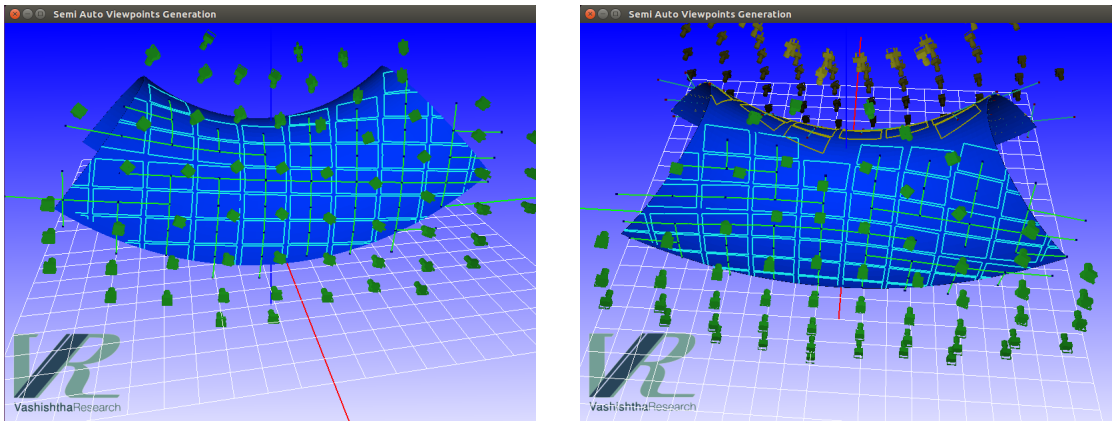


Figure 5.30: Concave Convex surface, Indirect method (Two start points)

Two start points can be used to cover most of the Saddle Shaped surface (Figure 5.30), however some area near the top is left uncovered, which would need a few more start points to be fully covered (not shown here).

5.5.7.2 Direct method

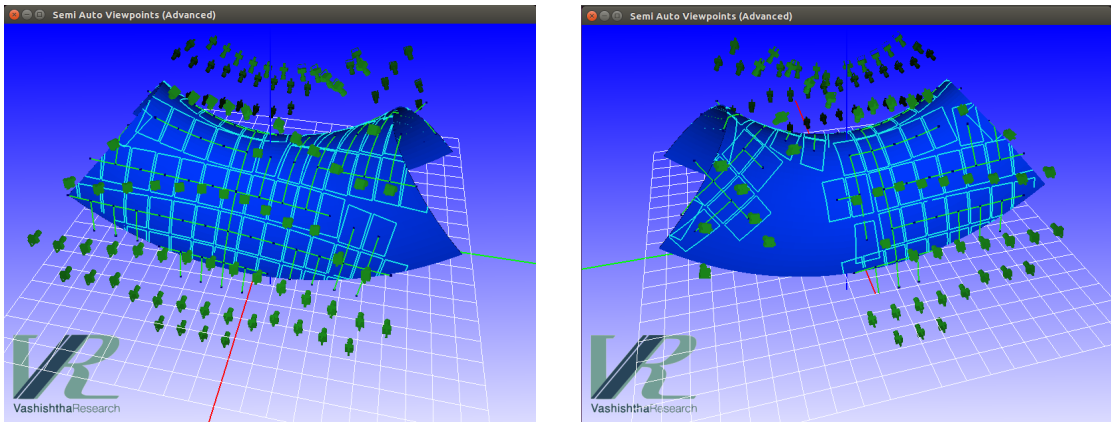


Figure 5.31: Concave Convex surface, Direct method (Viewed from two angles)

The Direct method works oddly and non-intuitively for the saddle shape, as the convergence and divergence of geodesics along different directions causes the node propagation to leave large patches of the surface uncovered (Figure 5.31). A few more start points (not

shown here) would be needed to cover these patches. In this case also, we can say that the Indirect method produces a more intuitive result.

5.5.8 Limitations of the method

As we can see, the proposed method works reasonably well for all the panel shapes we are likely to encounter and the indirect method is more likely to give better results in the cases where the direct method is not working well. In general, for more complex shapes, the further we get from the central assumption that near rectangular patches are being fitted to cover the surface, the more likely that this technique will not work well, however this would be true of any other algorithm as well. The other major limitation of this technique is that it is not fully automated but requires user input. The user has to select the starting criteria properly, which can take some experience.

5.5.9 Computational analysis

It was observed that this algorithm is extremely fast and takes only a few seconds on a normal laptop. This means that the preview corresponding to a start point and direction chosen by the user can be generated nearly in real time without causing frustration to the user. We can see from Algorithm 5.7 that the computational complexity of the algorithm is $O(n^2M)$ where n is the number of nodes and M is the number of triangle faces in the mesh. This is because Algorithm 5.7 and Algorithm 5.6 both iterate over the number of nodes (n) and Algorithm 5.3 iterates over the number of triangles (M), thus leading to the basic formula of $O(n^2M)$. Thus the algorithm runs in polynomial time. It can be noted that the number of nodes n, can be computed approximately in advance by dividing the panel surface area by the surface area covered by each node. Hence n depends on the camera parameters (Field of view, distance from the surface, etc.) and also on the size of the panel.

While a direct comparison with Optimization based methods is outside the scope of the present work (due to absence of literature addressing this specific topic of panel inspections), we can gain some insight about the speed efficiency of this method by looking at performance of the drone based coverage algorithm in [47] that is doing the coverage path planning over a church represented as a triangle mesh. In [47] , the coverage planning algorithm takes 2.48 minutes to complete for the church model represented by just 200 triangles. In comparison the method presented in the present work takes only a few seconds to cover the panels represented by over 10,000 triangles.

Further computational improvements can be made by using connectivity information

between triangles (instead of searching the whole mesh every time in Algorithm 5.3) and also between nodes in the 2D parametric space. This would necessitate using half edge data structure for the mesh and additional changes. More computational improvements are possible by modifying Algorithm 5.6 and Algorithm 5.7 to search only an appropriate subset of nodes instead of iterating over all nodes. However, because the current ('inefficient') algorithm still runs in just a few seconds on a normal laptop, the run speed was not considered a significant problem and the focus was more on simplicity rather than the efficiency of the algorithms. This can be explored in future work.

Since these results are obtained for models of actual panels used in the Aerospace industry, we believe that our algorithm can directly be applied in this area for non-destructive testing of actual panels. The Digital models of the camera and the surface are faithfully followed in order to get representative results relevant to the industry.

5.6 Travel Planning

The node insertion algorithm presented above generates the set of view points, but it doesn't tell us how to move from one view point to another in order to cover the surface. In general the problem of determining a minimum length motion path for a set of points such that each point is reached at least once is part of the Travelling Salesman Problem (TSP) which is NP-hard. We should note here that in the type of application planned here (Thermographic NDT), the inspection time is much greater than the travel time, and thus there is no need to emphasize an optimality in the path planning process.

The points generated from a single start point (using Indirect method as the stoppage criteria) are set out corresponding to a 2D grid. We propose a simple algorithm which would give a satisfactory solution to the problem. Note that this is only applicable to the case where these points are mapped roughly to a grid on the surface.

Figure 5.32 shows us the 2D space in which nodes have been mapped, each node representing a viewpoint. We proceed to develop a boustrophedon-like coverage path, starting from the right-lowermost node and moving up until all the nodes in that column have been covered. Then we move to the previous column uppermost node and change the direction of motion. This path planning is quite intuitive and so we do not describe the algorithm in more detail. [57] and [43] describe the boustrophedon cell decomposition methods which is what we follow but with a small difference. The 'cutout regions' where there are no points in the 2D space do not correspond to 'obstacles' in the conventional coverage path planning sense, since we can just pass over them (there is no physical obstacle). Hence

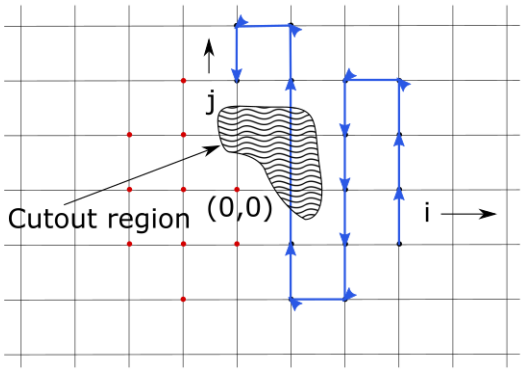


Figure 5.32: Boustrophedon Path for linking viewpoints

we do not need to break down the region into cells and so on. We can see how the 'cutout region' is handled in Figure 5.32. Although this travel path is not necessarily optimal, it is good enough for practical purposes.

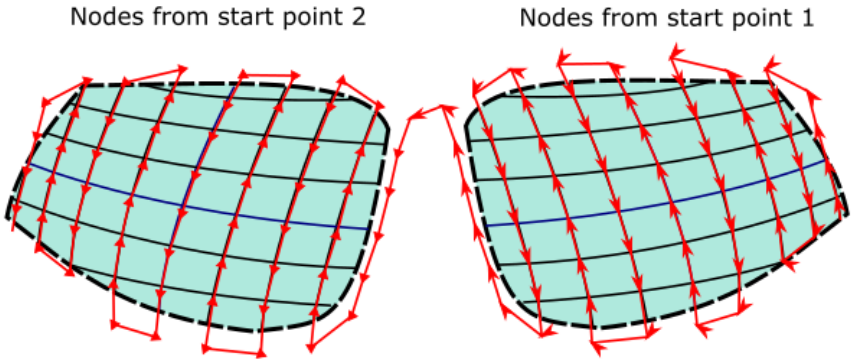


Figure 5.33: Linking paths from multiple start points

In the case where we we have multiple start points, we can link the end point of the first travel path to the start point of the second travel path thus generated and so on. Figure 5.33 illustrates this process.

For the case of path planning using the Direct method for stoppage criteria, or to ensure optimality in general, we can implement the solution using Travelling Salesman Approach. Since solutions to the Travelling Salesman Problem are well known in the literature, they are not explored here.

Since each node corresponds to a viewpoint, and each viewpoint can be represented by

the position and orientation of the sensor, we can thus derive the motion of the machine or robot for moving to these viewpoints in sequence. We could thus generate CNC G-code or Robot code for this purpose to be used in the actual application.

5.7 Summary

We have demonstrated a simple, easy-to-use and fast algorithm for coverage path planning for non-destructive testing of panels used in aerospace applications using camera viewpoints. This uses a geometric method relying on generating nodes on the surface using geodesic path finding, and it gives good results for representative sample panels used in the aerospace industry. The semi-automated approach relying on the user to set the start criteria, and two alternative approaches for setting node propagation stoppage criteria provides versatility and efficiency for covering the entire surface. This algorithm is much simpler than any optimization based method and more suitable to be implemented in real world applications.

The future work for this would include implementing the geodesic path finding algorithm using half-edge data structure for the mesh in order to make it more robust and computationally efficient. We could also calculate and demonstrate the actual coverage area on the panel for greater benefit to the user.

A further future scope of work would be to rigorously analyze the spatio-temporal complexity of the algorithm. An alternative method using an optimization and heuristic based approach could also be implemented to solve the same problem of panel scanning. Metric comparatives could then be generated to qualitatively assess the performance of geometric methods and optimization methods.

Chapter 6

Concluding Remarks

This chapter summarizes the results and lessons learned from the present study and points to areas of improvement which could be taken up as a future work. The concluding remarks are arranged into four sections, which deal with the development and limitations of prototype hardware, motion tracking integration, viewpoint planning and image stitching and automated coverage path planning respectively.

6.1 Development and Limitations of the Prototype hardware for Robotics and Motion Tracking

Due to the constraints of budget, the demonstration of algorithms and techniques have been done in this project using prototype hardware rather than procuring industrial grade motion tracking systems, NDT systems and Robotic systems. While this process was good enough to demonstrate the proof of concept for all these systems, the limitations of these prototype devices were quite significant. Efforts were made to locate and characterize these limitations. In the case of the 5-Axis Cartesian Robot, the accuracy and reliability were estimated using metrological techniques. In the case of motion tracking systems, optical estimation techniques such as measurement of Modulation transfer function and Noise estimation of the camera and lens system were carried out.

In future robotic systems and motion tracking systems can be designed and developed, with better characteristics, based on the experience obtained here.

6.2 Integration of Motion Tracking Systems with NDT

The implementation of a motion tracking system using an image processing pipeline was done for both a monocular and a stereo camera setup. The images are converted into pose data of rotation and translation using this data, and this pose is used for locating the projection of the NDT camera frustum on the 3D model of the panel.

A good image tracking hardware can be integrated with the NDT camera in future, based on the work done here. The location and orientation of the camera as the image is being captured can be recorded, used to correlate defects observed in images with their actual location on the panel. It can also be used for image stitching using methods as detailed in Chapter 4.

6.3 Viewpoint Planning and Image Stitching

A viewpoint planning system for planning the path of a robotic manipulator to scan a panel for defects by using a digital model of the system was detailed. The code from this path planning process was run on the prototype 5-axis robot and the images were recorded. Synthetic images were also generated from the path planning process in the virtual model. The images obtained were stitched onto the 3D model of the panel by using a 3D stitching algorithm as presented, which is based on mapping points in the pointcloud to the images taken. Blending is done at the overlapping areas covered by more than one viewpoint, using the Grassfire algorithm. Evaluation of the stitching result is done by comparing it with data from the actual panel. It was found that the method produced perfect results for the synthetic images, thereby verifying the technical correctness of the method. The results using the real world data were good, although there were some inconsistencies due to non-uniform lighting and the inaccuracy of the prototype robot.

The future work would involve repeating the same process using a good quality robot and an NDT system with uniform illumination to get better results. Also an octree data structure for pointcloud calculations could be used to improve computational speed in the image stitching algorithm.

6.4 Automated Coverage Path Planning

To improve on manual viewpoint planning techniques, an algorithm for automated coverage path planning of the surface to be inspected was sought. Although most literature

discusses optimization based approaches and slicing based geometric approaches, an alternative geometric method has been explored here which relies on geodesic path generation and mapping a grid of viewpoints onto the surface. Nodes corresponding to viewpoints are generated automatically and the node generation is stopped based on certain criteria. This approach is semi automated and relies on the user to give good starting conditions for the node propagation. A large number of examples are presented to show the strengths and weaknesses of the method. It was shown that this method produces good results for the example scenarios and is fast, simple and versatile compared to previous techniques.

The future work for this would include implementing the geodesic path finding algorithm using half-edge data structure for the mesh in order to make it more robust and computationally efficient. Rigorous proofs of Spatio-temporal complexity and generation of Metrics for comparisons to other techniques could also be done to prove the theoretical advantages of this method of coverage path planning.

Bibliography

[1] K. E. Cramer, D. F. Perey, and J. L. Brown, "The application of line scan thermography using multiple collaborative robots," *Review of Progress in Quantitative Nondestructive Evaluation*, 2019.

[2] K. E. Cramer, "Current and future needs and research for composite materials nde," in *Behavior and mechanics of multifunctional materials and composites XII*, vol. 10596. SPIE, 2018, p. 1059603.

[3] FastSuite, "Ultrasonic ndt with kuka for aerospace," 2019. [Online]. Available: https://www.youtube.com/watch?v=kfBkazbcoXc&ab_channel=FASTSUITE

[4] Dantec-Dynamics, "Dantec dynamics q 800 automated shearography robosystem," 2016. [Online]. Available: https://www.youtube.com/watch?v=t6IGP0skQb8&ab_channel=DantecDynamics

[5] J. Peeters, B. Bogaerts, S. Sels, B. Ribbens, J. Dirckx, and G. Steenackers, "Optimized robotic setup for automated active thermography using advanced path planning and visibility study," *Applied optics*, vol. 57, no. 18, pp. D123–D129, 2018.

[6] NASA and U. Robots, "How universal robots and robodk automate nasa's fuselage inspection," 2018. [Online]. Available: https://www.youtube.com/watch?v=PuejIauuP7o&ab_channel=UniversalRobots

[7] Tecnatom, "Tecnatom - wiipa cell," 2019. [Online]. Available: https://www.youtube.com/watch?v=LvBYJIZgqgw&ab_channel=Tecnatom

[8] Tecnatom, "Wiipa lite portable," 2021. [Online]. Available: https://www.youtube.com/watch?v=yoEaxY6Lahg&ab_channel=Tecnatom

[9] Innvotek, "Firefly inspect – ai-empowered drone system for inspecting wind turbine blades and airplane wings," 2022. [Online]. Available: https://www.youtube.com/watch?v=0JuXEShsYeg&ab_channel=Innvotek

[10] G. Dobie, R. Summan, C. MacLeod, and S. G. Pierce, "Visual odometry and image mosaicing for nde," *NDT International*, vol. 57, pp. 17–25, 2013.

[11] J. Shao, W. Zhang, Y. Zhu, and A. Shen, "Fast registration of terrestrial lidar point cloud and sequence images," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, no. 2/W7, 2017.

[12] C. Q. Wu, W. P. Wang, Q. G. Yuan, Y. J. Li, W. Zhang, and X. D. Zhang, "Infrared thermography non-destructive testing of composite materials," *Advanced Materials Research*, vol. 291, pp. 1307–1310, 2011.

[13] J. W. Newman, "Aerospace ndt with advanced laser shearography," in *17th world conference on nondestructive testing*, 2008, pp. 1–6.

[14] NASA and RoboDK, "Dual robot inspection at nasa - robodk," 2019. [Online]. Available: https://www.youtube.com/watch?v=rSf2_1NMg0Y&ab_channel=RoboDK

[15] J. N. Zalameda, W. P. Winfree, K. E. Cramer, P. D. Juarez, E. D. Gregory, and J. Brown, "Composite thermal nondestructive evaluation research at nasa langley," 2016.

[16] J. Peeters, "Robotic thermography on composite bicycle frame," 2017. [Online]. Available: https://www.youtube.com/watch?v=CvTcDwBE1IQ&ab_channel=JeroenPeeters

[17] C. L. Dotson, *Fundamentals of dimensional metrology*. Cengage Learning, 2015.

[18] B. J. Marsh, K. D. VanScotter, L. S. Bodziony, and G. E. Georgeson, "System and method for automated inspection of large-scale part," 2007, uS Patent US20080307886A1.

[19] G. D. Boreman, *Modulation transfer function in optical and electro-optical systems*. SPIE press Bellingham, Washington, 2001, vol. 4.

[20] M. Mapper, "Mtf mapper," 2021. [Online]. Available: https://sourceforge.net/projects/mtfmapper/

[21] C. Liu, W. T. Freeman, R. Szeliski, and S. B. Kang, "Noise estimation from a single image," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 1. IEEE, 2006, pp. 901–908.

[22] C. Liu, R. Szeliski, S. B. Kang, C. L. Zitnick, and W. T. Freeman, "Automatic estimation and removal of noise from a single image," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 2, pp. 299–314, 2007.

[23] A. López-Cerón and J. M. Cañas, "Accuracy analysis of marker-based 3d visual localization," in *XXXVII Jornadas de Automática Jornadas de Automática*. Comité Español de Automática, 2016, pp. 1124–1131.

[24] P. Irmisch, "Camera-based distance estimation for autonomous vehicles," Ph.D. dissertation, Technische Universität Berlin, 2017.

[25] M. Windolf, N. Götzen, and M. Morlock, "Systematic accuracy and precision analysis of video motion capturing systems—exemplified on the vicon-460 system," *Journal of biomechanics*, vol. 41, no. 12, pp. 2776–2780, 2008.

[26] P. Azad, T. Asfour, and R. Dillmann, "Stereo-based vs. monocular 6-dof pose estimation using point features: A quantitative comparison," in *Autonome Mobile Systeme 2009*. Springer, 2009, pp. 41–48.

[27] E. Olson, "Apriltag: A robust and flexible visual fiducial system," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 3400–3407.

[28] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[29] R. Horaud, G. Csurka, and D. Demirdijian, "Stereo calibration from rigid motions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1446–1452, 2000.

[30] B. K. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Josa a*, vol. 4, no. 4, pp. 629–642, 1987.

[31] S. Chatterjee and K. K. Issac, "Viewpoint planning and 3d image stitching algorithms for inspection of panels," *NDT & E International*, vol. 137, p. 102837, 2023.

[32] T. Schmidt and S. Dutta, "Production integrated ndt by means of automated thermography," *11th International Conference on Quantitative InfraRed Thermography*, 2012.

[33] C. Mineo, S. G. Pierce, P. I. Nicholson, and I. Cooper, "Robotic path planning for non-destructive testing–a custom matlab toolbox approach," *Robotics and Computer-Integrated Manufacturing*, vol. 37, pp. 1–12, 2016.

[34] M. Morozov, S. Pierce, C. N. MacLeod, C. Mineo, and R. Summan, "Off-line scan path planning for robotic ndt," *Measurement*, vol. 122, pp. 284–290, 2018.

[35] M. Antonello, S. Ghidoni, and E. Menegatti, "Autonomous robotic system for thermographic detection of defects in upper layers of carbon fiber reinforced polymers," in *2015 IEEE International Conference on Automation Science and Engineering (CASE)*. IEEE, 2015, pp. 634–639.

[36] R. Szeliski, "Image alignment and stitching," in *Handbook of mathematical models in computer vision*. Springer, 2006, pp. 273–292.

[37] R. S. Kaminsky, N. Snavely, S. M. Seitz, and R. Szeliski, "Alignment of 3d point clouds to overhead images," in *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 2009, pp. 63–70.

[38] H. Xu, L. Yu, J. Hou, and S. Fei, "Automatic reconstruction method for large scene based on multi-site point cloud stitching," *Measurement*, vol. 131, pp. 590–596, 2019.

[39] F. Hu, Y. Li, and M. Feng, "Continuous point cloud stitch based on image feature matching constraint and score," *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 3, pp. 363–374, 2019.

[40] H. Li, H. Liu, N. Cao, Y. Peng, S. Xie, J. Luo, and Y. Sun, "Real-time rgb-d image stitching using multiple kinects for improved field of view," *International Journal of Advanced Robotic Systems*, vol. 14, no. 2, p. 1729881417695560, 2017.

[41] E. Catmull and J. Clark, "Recursively generated b-spline surfaces on arbitrary topological meshes," *Computer-aided design*, vol. 10, no. 6, pp. 350–355, 1978.

[42] S. Chatterjee and K. K. Issac, "Viewpoint generation using geodesics and associated semi-automated coverage path planning of panels for inspection," *Applied Sciences*, vol. 14, no. 2, p. 906, 2024.

[43] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous systems*, vol. 61, no. 12, pp. 1258–1276, 2013.

[44] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2022.

[45] P. Wang, R. Krishnamurti, and K. Gupta, "View planning problem with combined view and traveling cost," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 711–716.

[46] C. S. Tan, R. Mohd-Mokhtar, and M. R. Arshad, "A comprehensive review of coverage path planning in robotics using classical and heuristic algorithms," *IEEE Access*, vol. 9, pp. 119 310–119 342, 2021.

[47] Z. Shang, J. Bradley, and Z. Shen, "A co-optimal coverage path planning method for aerial scanning of complex structures," *Expert Systems with Applications*, vol. 158, p. 113535, 2020.

[48] I. Z. Biundini, M. F. Pinto, A. G. Melo, A. L. Marcato, L. M. Honório, and M. J. Aguiar, "A framework for coverage path planning optimization based on point cloud for structural inspection," *Sensors*, vol. 21, no. 2, p. 570, 2021.

[49] K. O. Ellefsen, H. A. Lepikson, and J. C. Albiez, "Multiobjective coverage path planning: Enabling automated inspection of complex, real-world structures," *Applied Soft Computing*, vol. 61, pp. 264–282, 2017.

[50] E. V. Vazquez-Carmona, J. I. Vasquez-Gomez, J. C. Herrera-Lozada, and M. Antonio-Cruz, "Coverage path planning for spraying drones," *Computers & Industrial Engineering*, vol. 168, p. 108125, 2022.

[51] S. S. Mansouri, C. Kanellakis, E. Fresk, D. Kominiak, and G. Nikolakopoulos, "Cooperative coverage path planning for visual inspection," *Control Engineering Practice*, vol. 74, pp. 118–131, 2018.

[52] G. R. Kumar, P. Srinivasan, V. D. Holla, K. Shastry, and B. Prakash, "Geodesic curve computations on surfaces," *Computer Aided Geometric Design*, vol. 20, no. 2, pp. 119–133, 2003.

[53] C. C. Wang, K. Tang, and B. M. Yeung, "Freeform surface flattening based on fitting a woven mesh model," *Computer-Aided Design*, vol. 37, no. 8, pp. 799–814, 2005.

[54] S. Chatterjee and K. Issac, "Automated coverage path planning for inspection of panels using camera viewpoints," in *NDE 2021 - Virtual Conference and Exhibition, 09-11 Dec 2021*. Indian Society for NDT (ISNDT). e-Journal of Nondestructive Testing, 2022. [Online]. Available: https://www.ndt.net/?id=26731

[55] T. Möller, "A fast triangle-triangle intersection test," *Journal of graphics tools*, vol. 2, no. 2, pp. 25–30, 1997.

[56] A. N. Pressley, *Elementary differential geometry*. Springer Science & Business Media, 2010.

[57] H. Choset and P. Pignon, "Cover path planning: The cellular boustrophedon decomposition," in *Proceedings International Conference on Field and Service Robotics, Canberra, Australia*, 1996.

List of Publications

Refereed Journals

1. Chatterjee, Saurabh, and K. Kurien Issac. "Viewpoint planning and 3D image stitching algorithms for inspection of panels." NDT and E International 137 (2023): 102837.

2. Chatterjee, Saurabh, and Kaadaapuram Kurien Issac. "Viewpoint Generation Using Geodesics and Associated Semi-Automated Coverage Path Planning of Panels for Inspection." Applied Sciences 14.2 (2024): 906.

Refereed Conferences

1. Chatterjee, S.; Issac, K. Automated Coverage Path Planning For Inspection Of Panels Using Camera Viewpoints. In Proceedings of the NDE 2021—Virtual Conference and Exhibition, Virtual Event, 9–11 December 2021.