Coverage Planning for Robotic Vision Applications in Complex 3D Environment

Submitted in partial fulfillment of the requirements for
the degree of
Doctor of Philosophy
in
Mechanical Engineering

Wei JING

B.Eng., Electrical Engineering, Nanyang Technological University, Singapore
M.S., Mechanical Engineering, Carnegie Mellon University, USA

Carnegie Mellon University
Pittsburgh, PA

July, 2017

Abstract

Using robots to perform the vision-based coverage tasks such as inspection, shape reconstruction and surveillance has attracted increasing attentions from both academia and industry in past few years. These tasks require the robot to carry vision sensors such as RGB camera, laser scanner, thermal camera to take surface measurement of the desired target objects, with a required surface coverage constraint. Due to the high demands and repetitive natures of these tasks, automatically generating efficient robotic motion plans could significantly reduce cost and improve productivity, which is highly desirable.

Several planning approaches have been proposed for the vision-based coverage planning problems with robots in the past. However, these planning methods either only focused on coverage problems in 2D environment, or found less optimal results, or were specific to limited scenarios. In this thesis, we proposed the novel planning algorithms for vision-based coverage planning problems with industrial manipulators and Unmanned Aerial Vehicles (UAVs) in complex 3D environment. Different sampling and optimization methods have been used in the proposed planning algorithms to achieve better planning results.

The very first and important step of these coverage planning tasks is to identify a suitable viewpoint set that satisfies the application requirements. This is considered as a view planning problem. The second step is to plan collision-free paths, as well as the visiting sequence of the viewpoints. This step can be formulated as a sequential path planning problem, or path planning problem for short. In this thesis, we developed view planning methods that generate candidate viewpoints using randomized sampling-based and Medial Object-based methods. The view planning methods led to better results with fewer required viewpoints and higher coverage ratios. Moreover, the proposed view planning methods were also applied to practical application in which the detailed 3D building model needs to be reconstructed when only 2D public map data is available.

In addition to the proposed view planning algorithms, we also combined the view planning and path planning problems as a single coverage planning problem; and solved the combined problem in a single optimization process to achieve better results. The proposed planning method was applied to industrial shape inspection application with robotic manipulators. Additionally, we also extended the planning method to a industrial robotic inspection system with kinematic redundancy to enlarge the workspace and to reduce the required inspection time. Moreover, a learning-based robotic calibration method was also developed in order to accurately position vision sensors to desired viewpoints in these instances with industrial manipulators.

Acknowledgments

First, I would like to thank my thesis advisor Professor Kenji Shimada for his guidance and advices during my Ph.D research. Dr. Wei Lin as my co-advisor also extended great support with his patience and immense knowledge. Their coach has helped me in all times of my Ph.D research and accomplishment of this thesis, and I could not have imagined having a better advisor and mentor.

My sincere thanks also goes to other thesis committee members Professor Koushil Sreenath and Professor David Bourne, who provided me insightful comments and suggestions which triggered me to widen my research from various perspectives.

With a special mention and thanks to Dr. Joseph Polden and Dr. Pey Yuen Tao, for their help on the experiments and the work that we have done jointly.

Also to everyone from Computational Engineering and Robotics Lab, my colleagues in Carnegie Mellon, my soccer teammates from Pittsburgh United, and my friends from Ewaybot Technology, it was wonderful experience studying and working with all of you during last four years.

A very special gratitude goes out to Agency for Science, Technology and Research, for providing the financial support for my Ph.D research.

Most importantly, I would like to express my sincere gratitude to my beloved wife Xiaole Wang, my parents, and other family members, who have provided me thorough moral support. This work will never be accomplished without them.

Thank you all wholeheartedly, for those help and supports among the journey!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

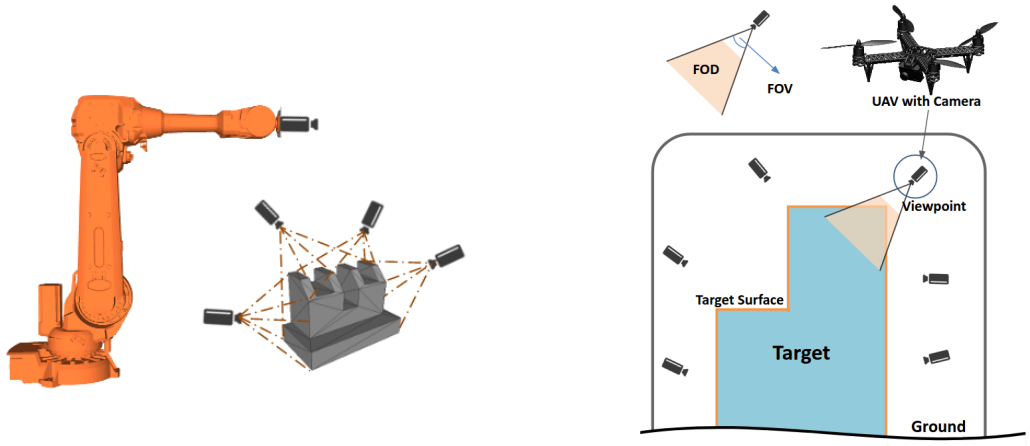
1.1 Background

Using autonomous robot for repetitive and labour-intensive tasks saves cost and improves efficiency, which is one of the most essential motivations for robotic researchers. These automation tasks with robots have been used extensively in many industrial applications over the years. Among all these robotic automation tasks, one category is to use robots with vision sensors to take a series of visual measurements, in order to achieve certain coverage requirements. These vision-based coverage tasks include inspection, surveillance and shape reconstruction.

There are many examples of the coverage tasks of the autonomous robotic system with vision sensors. For instance, in thermal inspection of a building [1], an Unmanned Aerial Vehicle (UAV) with a thermal camera flies around the building to check the heat loss from the building surfaces. Another application is the search and rescue operation in an urban area damaged by a natural disaster [2], where the UAVs are sent to take a set of images to search for survivors. Another example application is the shape inspection for production defects [3][4][5], in which the differences between the scanned model of the final product and the 3D CAD model of its intended design are recorded to quantify the variances from the manufacturing processes.

These vision-based coverage applications such as inspections, surveillances and shape reconstructions with robots require gathering surface information of the target objects as the very first step. The step of gathering surface information could take significant amount of time and human intervention. Thus planning effective robot motions automatically for these tasks to reduce cost and to improve efficiency is highly desirable. In these tasks, the target object could be of any arbitrary geometries and sizes, and placed in a complex 3D environment. Vision sensors, such as RGB cameras, depth sensor, laser scanner, thermal cameras, are carried by the autonomous robotic system to take surface measurement of the target object. Sensor measurements are subject to certain constraints such as occlusion, viewing angle and other vision-related constraints. These applications may have different surface coverage requirements according to different practical considerations. Typically, the whole surface area, or a certain portion of the

surface is required to be measured.



(a) **(b)**

Figure 1.1: Example applications: (a) the industrial shape inspection problem with robotic manipulators; (b) the UAV with vision sensor for inspection, surveillance and scene reconstruction

The robotic planning problem with coverage constraints is usually termed as Coverage Planning Problem (CPP) in literature [6][7][8], in which a series of sequential robotic actions need to be planned. In this thesis, we focus on the coverage planning problems of autonomous robot with vision sensors. In order to differentiate it from the coverage planning problems with contact operation in 2D space [6] such as the planning problems for robotic vacuum, we term this type of Coverage Planning Problems [8] as *Visual Coverage Planning Problem (VCPP)*. Note that CPP is also used to refer to the VCPP problems in literature [8].

1.2 Problem Statement

The Visual Coverage Planning Problem (VCPP) in complex 3D environment is challenging, because in addition to finding the collision-free path between the robot poses, VCPP also requires finding a motion plan of a ordered sequence of actions and a set of viewpoints that fulfill the coverage requirements. The planning algorithm (planner) for VCPP finds efficient and feasible

(if there is such a plan) motion plan such that the coverage constraints are satisfied, and the objective function is optimized.

VCPP consists of two sub planning tasks. The first task is to find viewpoints for the coverage requirement, which is named as view planning problem. The second task is to find collision-free paths and the visiting sequence of the viewpoints. We refer the second task as simply the path planning problem for short.

The view planning task is to identify the suitable viewpoints subject to coverage constraints. Usually a Set Covering Problem (SCP) is formulated for the view planning problem to select minimum number of viewpoints. In addition to choosing the viewpoints, it is also necessary to find the sequential motion plan of paths for the robot to visit the viewpoints. For the path planning problem, in addition to the pose-to-pose path-finding, a Traveling Salesman Problem (TSP) is formulated to find a feasible plan with minimum cycle time. The overall objective of the planning algorithm for VCPP is to minimize the total time required, as well as minimizing other additional objectives if presented.

1.3 Contributions and Thesis Organization

This thesis focuses on developing novel planning algorithms to improve the results of VCPP problem, with applications in inspection, surveillance and shape reconstruction using UAV and industrial manipulator. A brief summary of the major contributions is listed below.

Two view planning algorithms were developed in this thesis. In the first view planning algorithm, the randomized sampling and artificial potential field methods were used to generate high quality candidate viewpoints [9], such that better view planning results with higher coverage ratio and fewer viewpoints were achieved. In the second proposed view planning algorithm, voxel dilation, Medial Objects and Gaussian sampling were used to further improve efficiency of the viewpoints generation process. Random-Key Genetic Algorithm (RKGA) was also used in the optimization process to improve the the view planning results.

The view planning methods were also extended to the planning applications where detailed 3D features were to be captured when only public 2D map data was available [10]. The optimization problems in this scenario were also reformulated in the proposed method. The proposed planning method was also validated by sending a UAV to take the measurement at the planned viewpoints, and then reconstructing the detailed 3D model.

New formulations of VCPP were introduced for 3D shape inspection applications with industrial manipulator. The proposed planning method [11] automatically generated efficient motion plans for the inspection applications, by combining the view planning problem and path planning problem as a single planning problem and solving it in one sequencing optimization process using RKGA to achieve better results.

A motion planning algorithm was also developed for a robotic inspection system with kinematic redundancy [12]. The proposed method formulated the path planning problem as a Generalized Traveling Salesman Problem (GTSP) to address the kinematic redundancy. The overall planning problem was formulated as a Set-Covering-Generalized-Traveling-Salesman-Problem (SC-GTSP), and solved using RKGA.

A learning-based robotic calibration method was also developed to accurately position the vision sensors to the desired viewpoints [13]. The proposed calibration method combined Product-of-Exponential (POE) model and Gaussian Process regression to generate high calibration accuracy for the industrial manipulators.

The rest of the thesis is organized as follows:

- Chapter 2: review of the relevant robotic planning and coverage planning works in literatures;
- Chapter 3 to 5: discussion on the proposed view planning methods to place the viewpoints for different coverage planning applications with UAVs;
- Chapter 6 to 8: discussion on the proposed methods for coverage planning problems in the context of 3D shape inspection in production line with robotic manipulator;

- Chapter 9: overall conclusion and recommendation of the future directions.

Chapter 2

Literature Review

In this chapter, we discuss the relevant work of coverage planning problem and other relevant motion planning work in robotic applications. In our formulation, Visual Coverage Planning Problem (VCPP) is a high level planning problem, which consists of several sub planning problems. Thus, we list the relevant work of the overall VCPP problem and the sub planning problems in this Chapter as well.

2.1 Path Planning for Robotic System

One of the fundamental areas of robotic research is path planning. Path planning is usually to find the collision-free paths between robot poses in a given environment. Robotic path planning problem has been studied for many decades. In the early work, planning algorithms such as artificial potential field [14], Visibility Graph [15], cell decomposition [16] [17] were developed by robotic researchers.

In recent years, the sampling-based planning methods [18] became popular. The sampling-based planning algorithms randomly sample the Configuration Space (C-Space) or the state space of the robot, and then find a path based on these discrete samples. The multi-query planning method such as Probabilistic Roadmap (PRM) [19], single-query planning method such as Rapid-explore Random Tree (RRT) [20] and their variations [18] had been used in lots of applications because they are efficient and reliable in practice. Many of these sampling-based methods were proof *probabilistically complete* [21] but not optimal.

Later work [22] also proves that some sampling-based planning algorithms could be *asymptotically optimal* with modification. More recent work [23] [24] started to focus on the sampling strategies in order to further improve efficiency of the sampling-based planning method. Sampling-based methods were also combined with other methods to improve the performance. For example, Medial-Axis Probabilistic Roadmap (MA-PRM) [25] [26] and Medial-Axis Rapid-explore Random Tree (MA-RRT) [27] used Medial-Axis [28] to improve the sampling efficiency of exploring the C-space, in order to achieve better planning results.

The planning algorithms discussed in this section are designed for path-finding from one robot pose to another. This pose-to-pose planning problem is an essential part of robotic planning problems. Many other higher-level planning algorithms for more complicated applications are built on top of these algorithms.

2.2 Visual Coverage Planning Problem

In the coverage planning problems, the planning algorithms [8][29] are required to find a near-optimal robotic motion plan of a sequence of actions that fulfill the coverage constraints. Thus, in addition to the traditional pose-to-pose low-level planning, high level planning of sequential actions is also required, in order to automatically generate feasible and efficient motion plan for these coverage tasks.

In the VCPP applications such as inspection, surveillance and shape reconstruction, usually two planning sub-tasks are required: the view planning and the (sequential) path planning tasks.

2.2.1 View Planning Problems

View planning is an important sub-task in VCPP. It is to find the viewpoints that satisfy the coverage requirement, according to the predefined optimization objective(s). Depending on the availability of geometric model of the target objects, view planning can be categorized as model-based view planning or non-model-based view planning [30].

Model-based View Planning

For the model-based view planning, the geometric model of the target object is available beforehand (usually in polygonal mesh representation). Finding the optimal viewpoint set with a complete or high surface coverage is challenging, especially when the geometric shape of the target object is complex, and the coverage problems are often NP-hard. Since the 1980s researchers

have proposed different approximation methods to find the solution. Majority of the previous work used a two-step "generate-test" approach [30], with only few exceptions. For example, in the early 1990s, the aspect visibility model was applied to find the visible regions of surface patches, and the viewpoints were selected until full coverage is achieved [31].

Later, the "generate-test" approach [30] became popular. In 1995, the measurability matrix was introduced and it was found that the optimal solution from a candidate-viewpoint set generated from the surface of the view sphere enclosing the target object [32]. The measurability matrix was then modified and a candidate viewpoint set was generated using the "optimal scan zone" method, which placed viewpoints above every sampled surface patch by offsetting. After viewpoints generation, the optimization problem was then formulated as Set Covering Problem (SCP), and solved by integer linear programming [33]. Additional previously proposed methods are described in several survey papers: [34] [30] [35] [36] and [8].

The view planning problem in large scale was also studied by researchers [37] [38] in past years. View planning plays an important role to minimize the surveillance cost with optimal (fewest) viewpoints with complete coverage or a desired coverage ratio. Researchers [37] used an Unmanned Ground Vehicle (UGV) to monitor buildings, with the viewpoint positions in 2D environment. More recently, a planning method was presented for using a UAV for surveillance tasks in a 3D urban environment [39]. An evolutionary algorithm-based planning method was reported for a 3D urban environment that considered covering the top surfaces of buildings, roofs, along with the ground [38]. The model-based view planning usually generates near-optimal viewpoints for the coverage tasks offline, based on the optimization criterion according to different applications.

Non-Model-based View Planning

On the other hand, when the geometric model of the target object is not available before planning, non-model-based view planning problems can be formulated. Non-model-based view planning

is mainly applied to exploration, object reconstruction and modeling applications [30][40][41].

Most of the non-model-based view planning work [42] [43] used the Next-Best-View (NBV) framework to plan viewpoints without prior geometric information of the target object. The NBV methods iteratively selected new viewpoint online with the information acquired during sensing process, based on the information gain of the viewpoints, along with other optimization objectives and constraints [35][43].

2.2.2 Planning Sequential Paths of Viewpoints

In addition to solving the view planning problem, the sequential path planning problem needs to be solved. The path planning problem is to find the inverse kinematic (IK) solutions to position the vision sensor to the viewpoints, the collision-free paths, as well as the visiting sequence of the viewpoints.

Finding the IK solutions to position the vision sensor is a well studied topic for a 6-DOF robot. However, when there is kinematic redundancy involved [44][45], there are usually multiple solutions for a given robot pose. Thus, it could be difficult to select a good IK solution that leads to efficient overall planning results. Planning the motion between robotic poses usually utilizes the sampling-based methods discussed in Section 2.1 in C-Space. The sampling-based planning methods are efficient and reliable for querying the feasible paths between robot poses. The additional planning process is to determine a suitable sequence to visit each selected viewpoint. Moreover, collision-free robot paths must be identified to ensure the robot can move safely about its workspace as it visits the generated viewpoints. In this paper, the path planning problem refers to finding the robot path with the minimum cost through a set of way-points in the robot task space, similar to the work described in the literature [44][46].

The Traveling Salesman Problem (TSP) is usually formulated for the path planning problem of the coverage planning problem. Other extensions of TSP such as Traveling Salesman Problem with Neighbourhood (TSPN) [2] or Generalized Traveling Salesman Problem (GTSP) [46] [45]

are also formulated for the path planning problems in different applications.

2.2.3 Overall Visual Coverage Planning Problems

Another necessary step for VCPP is to combine the planning results of view planning problems and path planning problems to formulate an overall robotic motion plan. Early work usually combined the planning results directly by solve a SCP and TSP sequentially [47], which affects the quality of the optimization results.

Started from recent years, researchers started to consider traveling cost alongside with viewing cost [48]. The overall problem was formulated as Traveling View Planning Problem (Traveling VPP) by relaxing the Integer Linear Programming to Linear Programming problem. Later work [49] proposed algorithms that solve the view planning and path planning problems separately. Researchers [50][7] also proposed a sampling-based redundant roadmap method for large scale underwater inspection applications. The SCP and TSP were considered as a single combined optimization formulation, but several approximations were used and the the two NP-hard problems were also solved sequentially. A sampling-based planning method with consideration of differential constraints was proposed [51] to address the view planning and path planning problems simultaneously, but the proposed algorithm is mainly designed for coverage problems in 2D environment. Because of the computational cost, it is difficult to extend the planning algorithm to 3D complex environment.

2.3 Summary

In this chapter, we discussed the relevant robotic planning work from fundamental planning problems to visual coverage planning problems, a brief overview of the corresponding algorithms was also described.

Chapter 3

View Planning Using Sampling-based Method

In this chapter, we present a novel method that uses random sampling based method to generate the candidate viewpoints positions and potential field method to generate view directions. The proposed method here is general and applicable for view planning problem of any geometric shapes and sizes of target objects. The work has been published in [9].

3.1 Introduction

Nowadays, many commercial Unmanned Aerial Vehicles (UAVs) are equipped with an on-board camera that can be used for vision tasks such as visual inspection [1] and shape reconstruction [52]. These visual tasks require navigating the UAV around a target object whilst capturing a series of photographs which cover the surface area of the entire target object, or a designated portion of it. Shape reconstruction tasks are more demanding; each section of the target surface area must be captured at least twice from different positions to ensure a full 3D model of the target object can be reconstructed via triangulation. In this chapter, we propose a novel view planning method for the visual inspection and shape reconstruction tasks with UAVs.

In visual inspection and shape reconstruction applications discussed in this chapter, a UAV navigates about the target object and acquires surface information with an on-board vision sensor (as shown in Figure. 3.1). The goal of the view planning task is to minimize the number of individual viewpoints while completely covering the surface of the target object. In our application, the camera is attached to the UAV via a gimbal, which gives the camera six degrees-of-freedom (DOF) - three positional DOF and one rotational DOF from the UAV, and another two rotational DOF from the gimbal. The visible region of the camera sensor is defined by two parameters: Field of View (FOV) and Field of Depth (FOD). Since this is a model-based view planning problem, we assume that the rough shapes of the target objects are known, and that they are represented as polygonal surface patches.

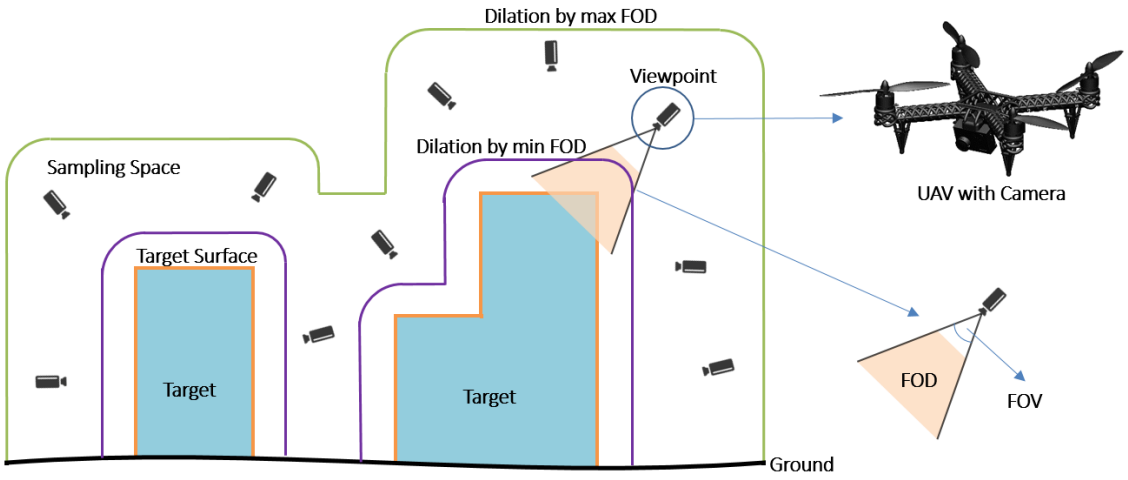


Figure 3.1: UAV visual coverage application example: the camera equipped UAV is able to acquire surface information of the target. The camera has a visible region that is defined by the FOV and FOD. The white space between the dilated boundary of maximum FOD (in green), the dilated boundary of minimum FOD (in purple) and ground (in black) is the sampling space in which the candidate viewpoints are sampled

3.2 Approach

In this work, a novel sampling-based view planning method is presented. The proposed approach utilizes an iterative two-step optimization framework, outlined in Figure 3.2. In the first step, candidate viewpoints are sampled within a constrained sampling space around the target object. These candidate viewpoints are passed to the second step, which selects the best subset of these viewpoints via combinatorial optimization. This two-step process is then iterated several times to improve the coverage ratio and reduce the number of required viewpoints. The remainder of this section presents the preprocessing step and the two iterative steps in more detail.

3.2.1 Preprocessing: Subdivision of Target Surface

In order to formulate the combinatorial optimization problem in later section, we firstly preprocess the target object by subdividing the surface of the target object into a mesh of triangular surface patches. In this paper, we use the Bubble Mesh [53] method to generate a well-shaped

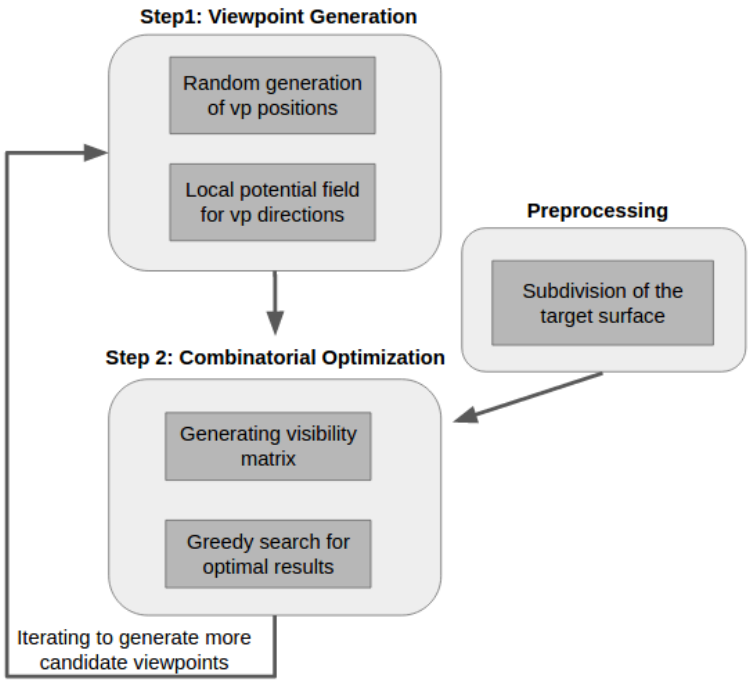


Figure 3.2: Proposed iterative two-step method for view planning problem.

triangular mesh that preserves the original shape well and also maintains patch uniformity. The subdivision example is shown in Figure. 3.3.

(a) Original mesh **(b)** After subdivision using Bubble Mesh [53]

Figure 3.3: Subdivision of model into a set of triangular surface patches

3.2.2 Viewpoints Generation

In this work, candidate viewpoints generation involves the following two steps:

- Randomized generation of viewpoint positions in the sampling space generated by taking difference of two dilation.
- Specification of viewpoint directions via probabilistic potential-field method.

Sampling space is required before we can begin sampling candidate viewpoints. In this chapter, we obtain the sampling space by using a binary voxel dilation [54] in 3D space. The dilation is a morphology operation defined by:

$$A \oplus B = \{a + b | a \in A, b \in B\}, \tag{3.1}$$

where A is the original 3D voxel model that is the target object in this chapter, and B represents the shape to be dilated.

As the camera's visibility is constrained by the FOD, two dilations of the target object are performed: First by the camera's maximum FOD, and then by its minimum FOD. The sampling space is obtained by taking the difference between the two dilations. The resultant sampling space can be considered as the complete feasible space, meaning that any viewpoint sampled outside this volume is unable to generate a valid view of any surface patches on the target object. Once complete, the set of candidate viewpoint positions are then randomly sampled inside this volume. The procedure to generate viewpoints is shown in Algorithm 1, noting that the solid model can be found from the mesh model of target object. Once the requisite number of viewpoint positions is generated, we move to assigning a suitable viewing direction for each sample, which is described in detail in next section.

After sampling the candidate viewpoint positions, the next step is to generate their corresponding viewing directions. In this chapter, we propose a novel probabilistic potential-field method to generate viewing directions. This is done by considering that nearby surface patches on the target object have an attraction force which is proportional to $1/d^2$. As shown in Eq. (3.2) below, the viewing direction for each point is calculated by summing nearby surface patch attraction forces. The resultant 3D force vector is then normalized to give a unit vector that specifies viewing direction of corresponding viewpoints.

Algorithm 1 Randomly generating viewpoint positions with voxel dilation

Input: The solid model of target object, T; a sphere with radius equals to maximum FOD, s_{max}; a sphere with radius equals to minimum FOD, s_{min}; and the number of viewpoints to be drawn, n_{vp}

Output: The candidate viewpoint set, $\mathbb{V}$;

1: $D_1 \leftarrow$ dilate(T, s_{max})
2: $D_2 \leftarrow$ dilate(T, s_{min})
3: $D \leftarrow D_1 - D_2$
4: **while** sizeof(V) $< n_{vp}$ **do**
5: $\quad v \leftarrow$ RandomSampleVP(D)
6: $\quad \mathbb{V} \leftarrow$ append($\mathbb{V}, v$)
7: **end while**
8: **return** $\mathbb{V}$

Formally, the viewing direction $\mathbf{v}$ of a given viewpoint located at $\mathbf{p_{vp}}$ is:

$$\mathbf{v} = \frac{\sum_i^n \frac{K\mathbf{d_i}}{\|\mathbf{d_i}\|^3}}{\| \sum_i^n \frac{K\mathbf{d_i}}{\|\mathbf{d_i}\|^3} \|}, \tag{3.2}$$

$$\text{where:} \quad \mathbf{d_i} = \mathbf{p_{vp}} - \mathbf{p_{patch_i}}$$

$$\text{for all:} \quad \{\mathbf{p_{patch_i}} | (\mathbf{p_{vp}} - \mathbf{p_{patch_i}})^T (\mathbf{p_{vp}} - \mathbf{p_{patch_i}}) < d_{max}\},$$

where K is a constant value; $\mathbf{p_{patch_i}}$ is the position of i^{th} surface patch; d_{max} defines the maximum inclusion distance, only surface patches located within this distance to the viewpoint will be included in the calculation of mean viewing direction.

After obtaining the viewing direction by potential-field method, we also add variance to the viewing direction. The variance modeled by Gaussian distribution in 3D space is added to the viewing direction, as shown in Eq. (4). The camera on the viewpoint is assuming always pointing to the viewing direction with y-axis pointing upwards.

$$\mathbf{v}' = \frac{\mathbf{v} + \mathbf{x_v}}{\|(\mathbf{v} + \mathbf{x_v})\|}, \tag{3.3}$$

where $\mathbf{x_v} \sim \mathcal{N}_3(\mathbf{0}, \mathbf{\Sigma})$ is the multivariate Gaussian distribution.

3.2.3 Combinatorial Optimization

The visibility matrix is a $m \times n$ binary matrix that measures Boolean visibility information of a given surface patch to a particular viewpoint. m represents the number of viewpoints, and n the number of surface patches. A surface patch is visible to a viewpoint if the following conditions are satisfied:

- The surface patch must be in the Field of View (FOV) of the sensor from the viewpoint.
- The surface patch must be in the Field of Depth (FOD) of the sensor from the viewpoint.
- The viewing angle β must be within a certain range predicted by the sensor specifications.
- There must be no occlusion, which means no solid element lies in between the surface patch and the viewpoint. This condition is calculated using a modified version of ray-triangle intersection algorithm [55].

In this chapter, we use a pinhole camera model and projection matrix [56] to model FOV visibility. By setting the skew parameter to 0, the ideal intrinsic matrix of a pinhole camera, $\mathbf{K}$, is:

$$\mathbf{K} = \begin{bmatrix} \alpha_x & 0 & p_x/2 \\ 0 & \alpha_y & p_y/2 \\ 0 & 0 & 1 \end{bmatrix} \in R^{3\times 3},$$

where α_x, α_y are the focal length in terms of pixel dimensions on x and y axe respectively; p_x, p_y are the number of pixels on the x and y axes, respectively.

The camera's extrinsic matrix, $\mathbf{E}$, is generated from the position, $\mathbf{T}$, and orientation, $\mathbf{R}$, of a given viewpoint:

$$\mathbf{E} = [\mathbf{R}, \mathbf{T}] \in R^{3\times 4}.$$

Given both the intrinsic and extrinsic matrix of the camera, the projection of a point $\mathbf{p} = [x, y, z, 1]^T \in R^{4\times 1}$ on the image plane can then be calculated as:

$$\mathbf{u} = \mathbf{K} \cdot \mathbf{E} \cdot \mathbf{p}, \tag{3.4}$$

where $\mathbf{u} \in R^3$.

To ensure visibility, the projection of a given point must be within the pixel range of the camera. Therefore, for a given point, if $0 \leq u_1/u_3 \leq p_x$ and $0 \leq u_2/u_3 \leq p_y$ after projection, then it is considered to be within the visible region of the given viewpoint, otherwise it is considered to be outside the visible region. The overall procedure to generate visibility matrix is shown in Algorithm 2.

Algorithm 2 Generating Visibility Matrix

Input: The set of surface patches, $\mathbb{P}$; the set of candidate viewpoints, $\mathbb{V}$; and the sensor parameters, s;

Output: The visibility matrix, $\mathbf{A}_{vm}$;

```
for each v_i ∈ 𝕍 do
    for each p_j ∈ ℙ do
        if isVisible(p_j, v_i, s) then
            A_vm(i, j) = 1
        else
            A_vm(i, j) = 0
        end if
    end for
end for
return A_vm
```

With the visibility matrix computed, the view planning problem in this paper can now be formulated as two types of Set Covering Problem (SCP): single coverage problem for visual inspection and multiple coverage problem for shape reconstruction.

The SCP is formulated to find the least number of viewpoints that cover for every surface patch:

$$\min \sum_{i=1}^{n} x_i, \quad \text{where } x_i \in \{0, 1\} \tag{3.5}$$

$$\text{s.t.} \quad \mathbf{A}_{vm} \cdot \mathbf{x} \geq \gamma,$$

where x_i is the i^{th} viewpoint, $x_i = 1$ means this viewpoint is selected and $x_i = 0$ means it is not selected; A_{vm} is the $m \times n$ visibility Matrix; $A_{vm} \cdot \mathbf{x}$ gives a $m \times 1$ vector, which specifies how many times each surface patch has been covered; γ is the coverage frequency, which in this case, is a $m \times 1$ vector. For inspection tasks, each element in γ is set to 1, whilst for reconstruction tasks, they are set to 2.

Algorithm 3 Greedy Search Algorithm for Set Covering Problem

Input: The set of surface patches, $\mathbb{P}$; the set of candidate viewpoints, $\mathbb{V}$; the visibility matrix, $\mathbf{A}_{vm}$; and the coverage frequency, γ

Output: The coverage ratio, δ; the number of resultant viewpoints n_{res}; and the resultant viewpoints set V_{res}

$V_{res} \leftarrow \emptyset$
$\mathbb{P}_{uncover} \leftarrow$ findUncovered($\mathbf{A}_{vm}, \gamma$)
$\mathbb{P} \leftarrow$ delete($\mathbb{P}$, $\mathbb{P}_{uncover}$)
$\delta = 1-$ sizeof($P_{uncover}$) / sizeof($\mathbb{P}$)
while $\mathbb{P} \neq \emptyset$ **do**
    $v \leftarrow$ maxCover($\mathbb{V}, \mathbb{P}, \mathbf{A}_{vm}$)
    $\mathbb{V}_{res} \leftarrow$ append($\mathbb{V}_{res}, v$)
    $\mathbb{V} \leftarrow$ delete($\mathbb{V}$, v)
    for each $p_j \in \mathbb{P}$ **do**
        if count($p_j, \mathbb{V}_{res}, \mathbf{A}_{vm}$) $> \gamma$, **then**
            $\mathbb{P} \leftarrow$ delete($\mathbb{P}$, p_j)
        end if
    end for
end while
n_{res} = sizeof($\mathbb{V}_{res}$)
return $\delta, n_{res}, \mathbb{V}_{res}$

SCP is a NP-hard problem with no available solution in polynomial time. However there are a number of approximate methods available, such as greedy algorithm [57], Dynamic Programming [58], and Genetic Algorithm [59], which can be used. In this chapter, we use a modified greedy search algorithm to solve SCP due to low computational cost and acceptable results for this application. The greedy algorithm used in this chapter is defined in Algorithm 3.

3.2.4 Iteration of the Two-Step Process

If there remains surface patches on the target object that have not been covered after solving the SCP, an iterative process is employed to generate additional viewpoints which target these specific patches. The same sampling space is used to randomly generate additional viewpoint positions. When calculating viewing directions, however, only the unseen surface patches and the patches visible from less than certain number of viewpoints in previous iterations are considered for the probabilistic potential-field method. The newly generated viewpoints are added to the previous candidate viewpoint set before performing the combinatorial optimization procedure outlined earlier. The iterative process is repeated a number of times to further improve the coverage ratio and overall results.

3.3 Experiments

3.3.1 Setup

In this section, we conduct a series of computational tests to compare the effectiveness of our method against two existing approaches used in inspection and reconstruction tasks. Our method is compared with two previous methods: offsetting method [33] and viewing sphere method [32][47]. All three methods use the general two-step "generate-and-test" framework, as well as the same optimization strategy in the second step. The difference between these methods is in how the candidate viewpoint positions and directions are generated.

The offsetting method generates candidate viewpoints by directly offsetting a distance from each surface patch. Viewing directions are then generated to point directly towards the corresponding patch. The view-sphere method generates candidate viewpoints on the surface of a virtual sphere enclosing the target object. Viewing directions are generated so that they point towards the centre of the sphere. The same combinatorial optimization strategies are used in all three methods. For the iterative sampling method proposed in this chapter, the maximum number of allowable iterations is set to three.

(a) Target object 1 with the bounding box size of $85m \times 100m \times 180m$

(b) Target object 2 with the bounding box size of $1.12m \times 1.37m \times 1.68m$

Figure 3.4: The target objects

The target objects used for the tests are shown in Figure. 3.4. The first target object used for view planning computation consists of $2,570$ individual triangular patches and stands at a height of approximately 180 meters. The second target object that consists of $2,566$ triangular patches is a smaller object, approximately $1/100th$ the size of the building structure.

Two computational tests have been conducted in this chapter. Firstly, an inspection task on Target object 1. For this inspection task, surface patches on the target object only need to be covered at least once. The view planning parameters for this test are given in Table 3.1. The second test involves a reconstruction task of Target object 2. For this reconstruction task, surface patches must be covered at least twice for triangulation purposes. The relevant parameters for this view planning process are listed in the table 3.2. In each of the two tests, the three view planning algorithms will be implemented and their performance compared. The results of the

tests are presented in the Section 3.3.2.

Table 3.1: View planning parameters for the target object 1

Focal Length (35mm equivalent)	20mm
Camera Pixels	$3,000 \times 2,000$
Viewing Angle	$80°, 90°$
FOD (meters)	(1, 50), (1,100)
Coverage Frequency	Single $\gamma = 1$

Table 3.2: View planning parameters for the target object 2

Focal Length (35mm equivalent)	30mm
Camera Pixels	$3,000 \times 2,000$
Viewing Angle	$70°, 80°$
FOD (meters)	(0.5, 5)
Coverage Frequency	Multiple $(\gamma = 2)$

For both tests, the number of viewpoints required n_{res} is calculated by the greedy search method proposed in Algorithm 3. The coverage ratio δ is computed by using the number of covered surface patches divided by the number of total surface patches, which is also from Algorithm 3.

3.3.2 Comnputational Tests

The results shown in Table 3.3 indicate that the proposed method requires fewer individual viewpoints to cover Target object 1 for the inspection application, and it also features the best coverage ratio. On average, our proposed method requires 44.8% and 54.6% fewer viewpoints compared to the other two methods, respectively. The 3D visualization of the resultant viewpoints is presented in Figure 3.5a.

Table 3.3: Results from Test 1.

Parameters	**Sampling**		**Offset**		**SPH**	
	n_{res}	δ	n_{res}	δ	n_{res}	δ
$(\mathbf{50}, \mathbf{90}^\circ)$	37	100%	74	99.9%	82	55.2%
$(\mathbf{50}, \mathbf{80}^\circ)$	45	100%	92	99.8%	92	54.0%
$(\mathbf{100}, \mathbf{90}^\circ)$	13	100%	22	100%	46	94.7%
$(\mathbf{100}, \mathbf{80}^\circ)$	22	100%	35	99.4%	51	89.6%

n_{res} is the number of viewpoints required, and δ is the coverage ratio. $(\mathbf{50}, \mathbf{90}^\circ)$: means maximum FOD, d_{max} is 50 meters and the maximum viewing angle $\beta_{max} = 90^\circ$

"Sampling" refers to the sampling-based method proposed in this chapter; "Offset" refers to the offsetting method [33]; "SPH" refers to the sphere method [32][47].

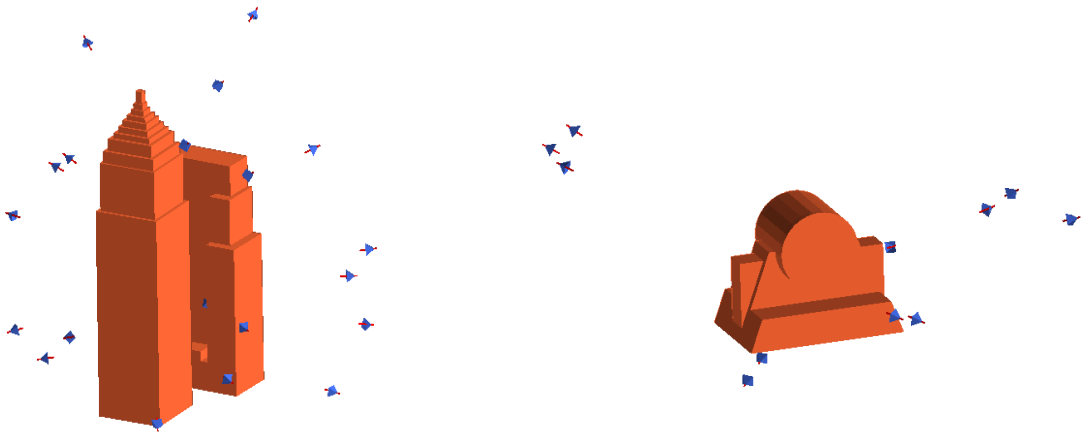
The results listed in Table 3.4 show that the proposed method requires fewer viewpoints to cover the Target object 2 for the reconstruction application, whilst maintaining the highest coverage ratio as well. On average, the proposed method requires 44.3% and 9.4% fewer viewpoints compared to the previous two methods on average. The 3D visualization of the resultant viewpoints is presented in Figure 3.5b.

Table 3.4: Results from Test 2.

Parameters	**Sampling**		**Offset**		**SPH**	
	n_{res}	δ	n_{res}	δ	n_{res}	δ
$(\mathbf{5}, \mathbf{80}^\circ)$	13	100%	25	100%	16	99.8%
$(\mathbf{5}, \mathbf{70}^\circ)$	19	100%	32	99.4%	19	99.4%

n_{res} is the number of viewpoints required, and δ is the coverage ratio. $(\mathbf{5}, \mathbf{80}^\circ)$: means maximum FOD, d_{max} is 5 meters and the maximum viewing angle $\beta_{max} = 80^\circ$

"Sampling" refers to the sampling-based method proposed in this chapter; "Offset" refers to the offsetting method [33]; "SPH" refers to the sphere method [32][47].



(a) The 3D visualization of the Target object 1 and planned viewpoints using the proposed method

(b) The 3D visualization of the Target object 2 and planned viewpoints using the proposed method

Figure 3.5: The 3D visualizations of computational tests

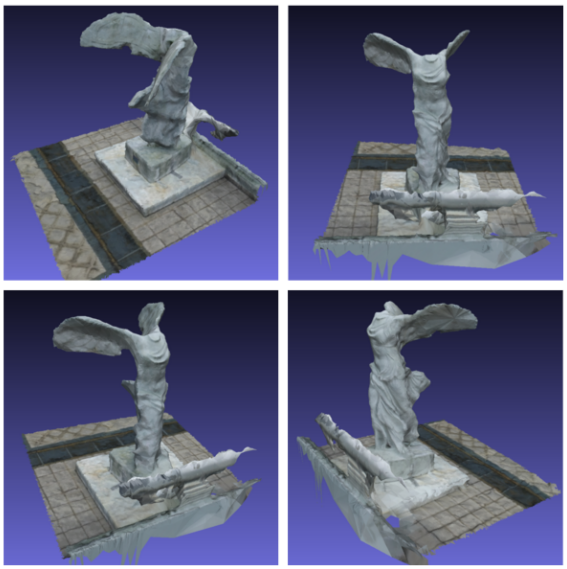
3.3.3 Field Test

The sampling-based view planning algorithm presented in this chapter is applied to find the viewpoints required for 3D reconstruction of the statue. The result publishes 16 individual viewpoints required to cover the statue. The UAV is piloted to each of the specified viewpoints to capture an image of the statue. The resulting reconstructed model of the statue, shown in Figure. 3.7a and Figure. 3.7b, is generated with the software VisualSFM [60][61] and CMP-MVS [62].

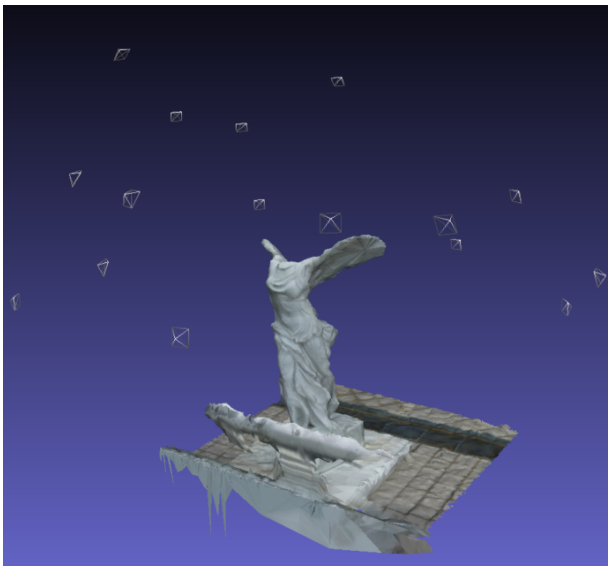
3.4 Summary

During the computational tests, it is observed that all three methods shares a similar time-frame required to generate their respective solutions. Further inspection of the timing reveals that the computational cost can be broken down into three key parts: The generation of candidate viewpoints, the generation of the visibility matrix and solving the SCP problem. Of these components, the most computationally expensive, by a wide margin, was encountered when generating the visibility matrix. All three methods use the same approach to do this, which takes around 20

Figure 3.6: The picture captured at the planned viewpoints



(a) Different viewing angles of the 3D reconstruction results of the statue



(b) The 3D reconstruction result of the statue with the camera positions

Figure 3.7: The 3D reconstruction result

minutes to complete (using single thread with Intel Core i5-3320M CPU and 8G RAM laptop). This result is encouraging, as it shows that our proposed method is able to out-perform previously available approaches without any significant penalty in computation time, which further enhances its suitability for real-world applications.

The results obtained from the tests performed earlier have indicated that our proposed method is able to outperform the others without requiring additional computation time. This can be attributed to a number of factors. Firstly, our method generates viewpoints via randomized sampling, which implies that the method is non-deterministic, allowing performance to be relatively consistent for a variety of different target objects. This is in contrast to the other two methods tested, which generate viewpoints deterministically from a small subset of all feasible space. This can give inconsistent performance with different target geometries, and in some cases, may not be able to provide a solution at all. Additionally, the potential-field method used to generate viewpoint directions ensures the visible surface area of the target object is maximized for each individual viewpoint. This effect is particularly useful for viewing difficult regions of the target object; when the iterative process is initiated, the easier to view surface patches will not be included in the potential-field calculations (as they have already been covered), meaning that all generated viewpoints will then point towards the difficult to capture regions of the target object's surface.

Chapter 4

View Planning Using Dilation and Medial Objects

In this work, we use a geometric approach to improve the candidate viewpoints sampling process. The proposed method uses voxel dilation, Medial Object and Gaussian Sampling to generate high quality candidate viewpoints. Random-Key Genetic Algorithm (RKGA) is also applied to the optimization problem in later step to achieve better planning results.

4.1 Introduction

Applications such as building inspection and surveillance with Unmanned Aerial Vehicle (UAV) have attracted increasing attentions in recent years, as a result of the rapid development of UAV technologies. Since these are repetitive tasks, the very first step of these applications is to identify an efficient plan of necessary viewpoints. The process is usually a view planning problem, which is also referred to as sensor planning problem [63], or sensor placement problem [64] in literatures. View planning is to find a set of viewpoints, including positions and orientations, that covers the required surface regions of the target objects [32][30]. In the inspection and surveillance applications of view planning [9][35], the common optimization objectives are usually to reduce the number of viewpoints and to increase the coverage ratio of the surface areas.

In this chapter, we will focus on model-based view planning problem for building inspection and surveillance application in 3D environment. The goal is to develop an algorithm that could find minimum number of viewpoints that has the high coverage ratio of the surface area of given target with acceptable computational cost.

4.2 Approach

In this chapter, we propose a novel two-step computational method for model-based view planning for building inspection and surveillance applications. The proposed approach combines geometric methods, sampling-based methods and combinatorial optimization methods to generate view planning results with good coverage ratio and fewer viewpoints for the coverage re-

quirement. The first step of the proposed method uses two geometric methods: voxel dilation and Medial Object (MO), together with Gaussian sampling to generate the candidate viewpoints. The second step finds the optimal subset of the candidate viewpoints by formulating the partial Set Covering Problem (SCP) and solving it with the Random-Key Genetic Algorithm and Greedy search method. Figure 4.1 illustrates the process flow of the proposed framework.

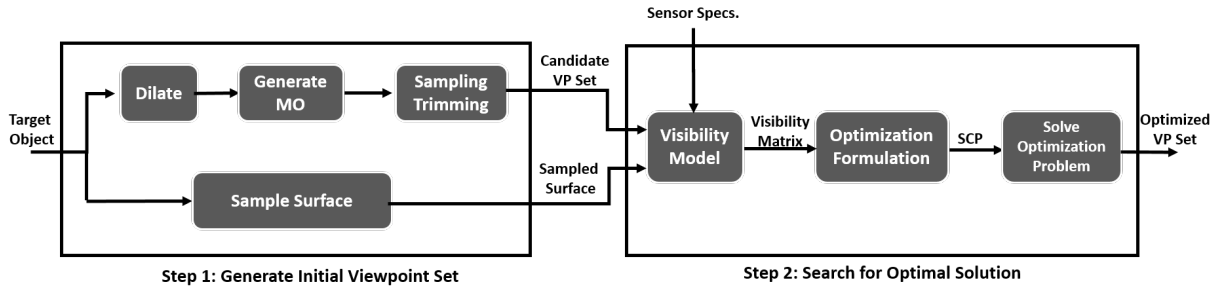


Figure 4.1: The proposed two-step computational method: the first step finds a set of candidate viewpoints and the second step finds the optimal subset of candidate viewpoints that covers all the surfaces of the target buildings

4.2.1 Generation of Candidate Viewpoints Set

The purpose of the first step is to generate a set of quality candidate viewpoints. Skipping this step and trying to find an optimal set of viewpoints directly would require optimization in high dimensional space that takes impractically high computational cost. Pre-selecting candidate viewpoints with good sampling strategies narrow down the search space and reduce the computational cost significantly.

High-quality candidate viewpoints should satisfy the following conditions:

- Candidate viewpoints should have a high coverage ratio. This is because the coverage ratio of a subset of candidate viewpoints selected in the second step is capped by the coverage ratio of the candidate viewpoints.
- Candidate viewpoints should include the ones that can see reentrant edges and corners of the target buildings.

- The number of candidate viewpoints should be comparable to the number of surface patches. Too many candidate viewpoints would lead to an unnecessarily high computational cost, and too few candidate viewpoints would lead to a low coverage ratio.

As illustrated in Figure 4.2, the proposed method generates such high-quality candidate viewpoints by: (1) voxelizing the building shapes and dilating the voxelized building shapes, (2) generating MO of the dilated volume, (3) removing MO points that lie on a subset of MO defined by dilated surfaces only, and (4) sampling candidate viewpoints using Gaussian sampling on the MO and generate the viewing direction using local potential field methods. The details of these steps are explained in Section 4.2.1 to 4.2.1.

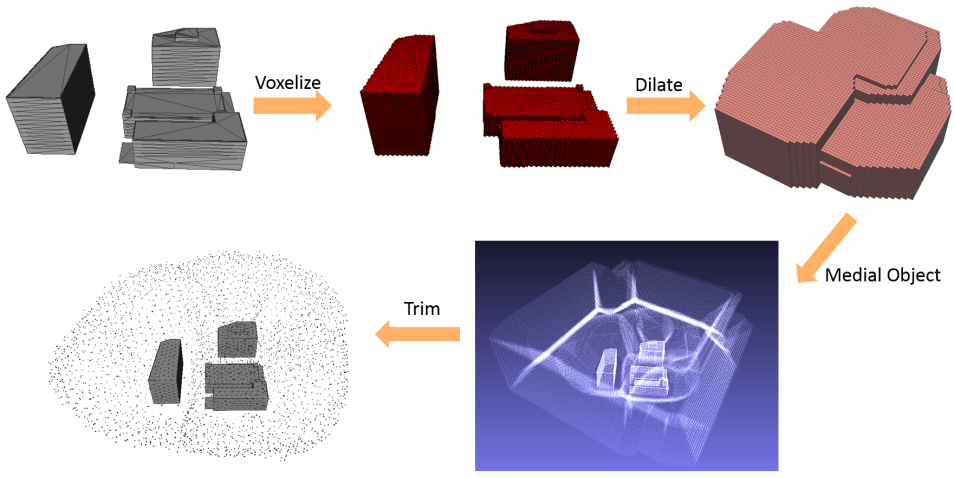


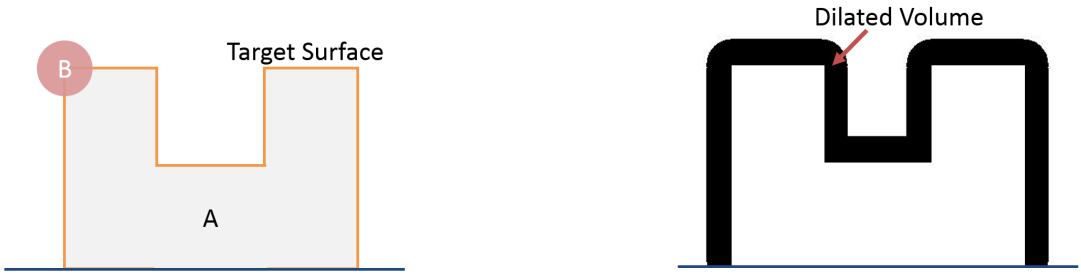
Figure 4.2: Generation of high-quality candidate viewpoints using voxel dilation and MO.

Binary Voxel Dilation

Binary dilation, also known as Minkowski Sum [54], is a basic operation in mathematical morphology. It is a convolution-like operation in 3D space, and generates a dilated object as illustrated in Figure 4.3. Dilation is widely used in computer graphics, motion planning, image processing and computational geometry. In Euclidean space, binary dilation is defined as:

$$A \oplus B = \{a + b | a \in A, b \in B\}.$$

In the proposed method, binary voxel dilation of the target object is used to generate the dilated volume (black part of right figure in Figure 4.3 (b)), whose MO will then be used to generate candidate viewpoint set in later steps. The dilated volume is obtained by subtracting the dilated object with the original target object.



(a) The original model, A; and dilation mask, B.

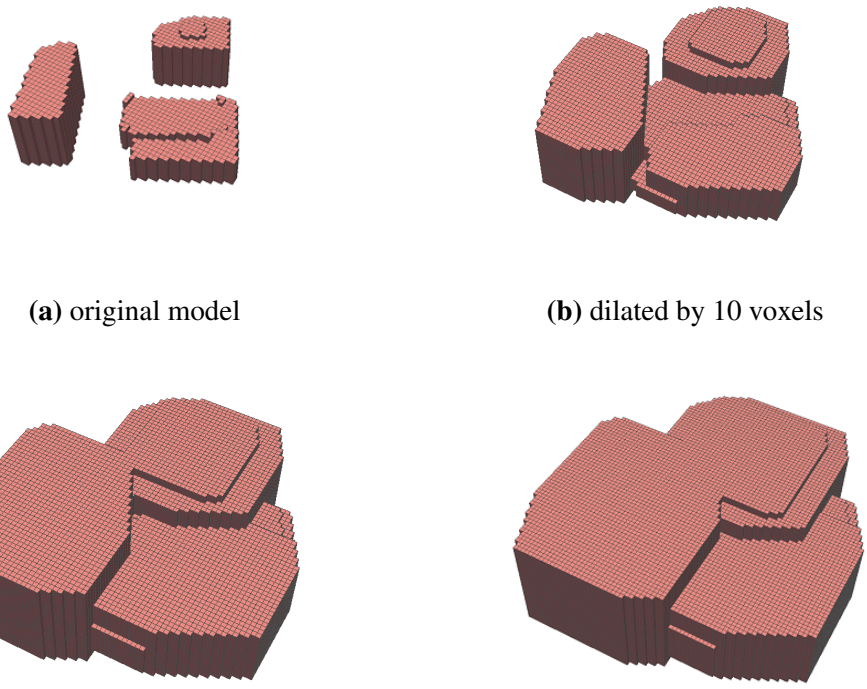
(b) The dilated volume (shown in black).

Figure 4.3: Example of the cross section of 3D dilation

In this chapter, the building inspection and surveillance applications require to cover all model surfaces except the bottom surface. In order to incorporate better with discrete voxel space, cube is used for dilation rather than sphere in our implementation. The examples of dilated models with different dilation sizes are shown in Figure 4.4. The size of dilation is determined by the FOD of the vision sensor. Normally when the viewing distance is closed to the maximum of the FOD, the viewing area of sensor is maximized. In the proposed method we find experimentally that dilating our target object by about 1.4 to 1.6 times the maximum of sensor FOD yields good results. The MO generated from the dilated volume is then located at a distance of about 0.7 to 0.8 times the maximum of sensor FOD from the target.

Medial Object Generation

Medial Objects (MO), also refereed as Medial Axis Transform (MAT) [28] in literature, are the skeleton of a target object, and consist of all points that have the same closest distance to at least two points on the original target object boundary [65][28]. The important property of MO



(a) original model

(b) dilated by 10 voxels

(c) dilated by 20 voxels

(d) dilated by 30 voxels

Figure 4.4: The original and dilated objects

is that it reduces the dimension of original model and preserves the geometric information of original model. The MO based methods have been used for robot pose-to-pose planning in past few years [26][66] to improve the efficient of exploring the space. However, to the best of our knowledge, it is the first time to use MO for coverage planning problems with voxel dilation and Gaussian sampling, in the UAV inspection and surveillance applications. In this chapter, we use the "Skeleton Sandbox" [67] to generate MO of a voxel model. The MO is defined as:

$$\mathbb{S} = \{s \in \mathbb{V} | \exists a, b \in \delta\mathbb{V}, a \neq b, \|a - s\| = \|b - s\| = min(D(s, v(v \in \delta\mathbb{V})))\},$$

where $\mathbb{S}$ is the point set of MO; $\mathbb{V}$ is the original model; $\delta\mathbb{V}$ is the boundary of original model; $D(x, y)$ is the Euclidean distance between two points.

In order to reduce the computational cost, We uniformly resample the MO points and delete

the MO points that do not point to the surface of target object.

Sampling Viewpoints Around Medial Object

In order to obtain the viewpoints from the MO, a sampling process is performed on the MO. We perform a sampling of Gaussian distribution with the mean value on the medial object, and a covariance matrix of $\sigma\mathbf{I} \in R^{3\times3}$. A viewpoint position is determined by randomly choosing a point on MO, and perform the Gaussian sampling strategy [68] with the mean on the chosen MO point.

In addition to the viewpoint positions, the viewing direction is obtained using probabilistic local potential field method. As documented in our previous work [9], the method assume that there are attraction forces from the surface of the target buildings, and the direction of the total attraction force on a certain viewpoint is the viewing direction of that viewpoint.

4.2.2 Select Viewpoints by Combinatorial Optimization

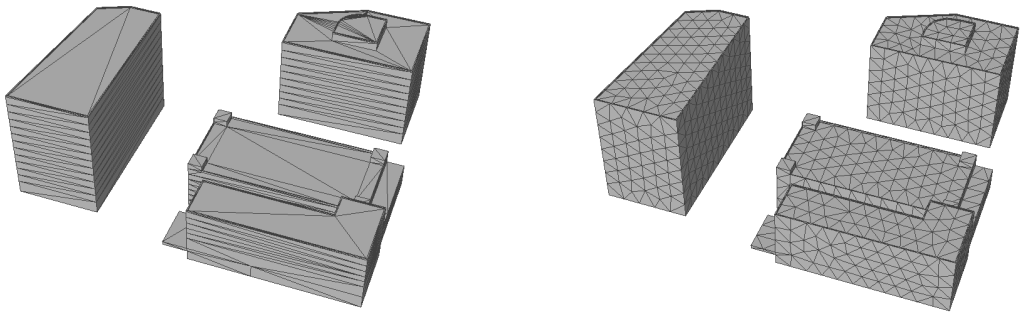
Because the visibility function that maps viewpoints to surface patches is non-parametric set-to-set mapping, the problem has to be formulated as a combinatorial optimization problem.

The previous step uses geometric information to reduce the number of viewpoints and surface patches in view planning problem to a reasonable number for the combinatorial problem in this step.

Subdividing the Model of Target Object

For the optimization problem formulation, another task is to subdivide the surface of the target object into a mesh of triangular surface patches, which is considered as sampling the surface. Subdividing the surface with reasonable patch size and preserve their original shape is an important task in this work; if the patch size is too large, the result may not be able to fully cover the surface because the visibility test checks the center point of the triangle only; if the patch size

is too small, the computational cost can become impractically high. In this chapter, we used the Bubble Mesh [53] method to generate a well-shaped triangular mesh that preserves the original shape well and maintains patch uniformity. Figure 4.5 shows the meshing results.



(a) The original mesh of the building

(b) The uniformly subdivided mesh using Bubble Mesh method (2,404 faces and 1,258 vertices)

Figure 4.5: Example: Subdivision of model into a set of triangular surface patches

Encoding Visibility Information with Visibility Matrix

In the proposed method, a visibility model is established to decide whether a surface patch can be viewed from a viewpoint. In the model, a surface patch is visible from a viewpoint if all the following conditions are satisfied:

- The surface patch must be in the Field of View (FOV) of the sensor from the viewpoint.
- The surface patch must be in the Field of Depth (FOD) of the sensor from the viewpoint.

$$d_{min} < (\mathbf{p_{vp}} - \mathbf{p_{patch}}) \cdot \mathbf{n_{vp}} < d_{max}$$

- The viewing angle β must be within a certain range predicted by the sensor specs.

$$\beta_{min} < \arccos(\frac{(\mathbf{p_{vp}} - \mathbf{p_{patch}}) \cdot \mathbf{n_{patch}}}{\|\mathbf{p_{vp}} - \mathbf{p_{patch}}\| \, \|\mathbf{n_{patch}}\|}) < \beta_{max}$$

- There must be no occlusion, which means no solid elements lies in between the surface patch and the viewpoint. This condition is calculated using a modified version of ray-triangle intersection algorithm [55], which has $O(mn^2)$ running time, where m is the number of viewpoints and n is the number of surface patches. The algorithm connects a viewpoint and surface patch with a line segment, and checks whether the line segment intersects with other triangles of the target object.

In this chapter, we use a pinhole camera model and projection matrix [56] to model the FOV visibility. by setting the skew parameter to 0, the ideal intrinsic matrix of a pinhole camera is:

$$\mathbf{K} = \begin{bmatrix} \alpha_x & 0 & p_x/2 \\ 0 & \alpha_y & p_y/2 \\ 0 & 0 & 1 \end{bmatrix} \in R^{3\times 3},$$

where α_x, α_y are the focal length in terms of pixel dimensions on x and y axis; p_x, p_y are the number of pixels on x and y axis.

The extrinsic matrix is decided by the position $\mathbf{T}$ and orientation $\mathbf{R}$ of a given viewpoint:

$$\mathbf{E} = [\mathbf{R}, \mathbf{T}] \in R^{3\times 4}.$$

Then given the intrinsic and extrinsic matrix of camera,the projection of a point $\mathbf{p} = [x, y, z, 1]^T \in R^{4\times 1}$ on the image plane is calculated:

$$\mathbf{u} = \mathbf{K} \cdot \mathbf{E} \cdot \mathbf{p}.$$

To ensure visibility, the projection of a given point must be within the pixels range of camera. Therefore, for a given point, if $0 \leq u_1/u_3 \leq p_x$ and $0 \leq u_2/u_3 \leq p_y$ after projection, then it is within the visible region of the given viewpoint with the camera specification, otherwise it is considered as not in the visible region.

The $m \times n$ visibility matrix is obtained by testing the visibility of m surface patches from n viewpoints using the above-mentioned visibility model. Each element of the visibility matrix takes 0 or 1 and indicates whether a given patch can be seen from a given viewpoint. Figure 4.7 shows an example of visible areas with a given viewpoint.

(a) **(b)**

Figure 4.6: The visibility model

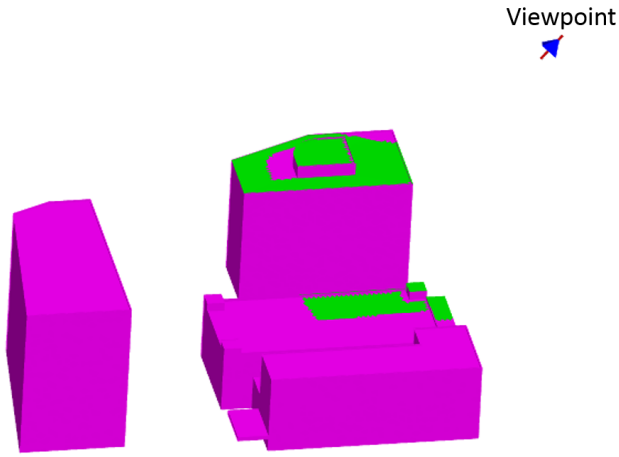


Figure 4.7: An example of visible areas with a given viewpoint. The visible areas are shown in green.

Set Covering Problem Formulation

After the visibility matrix has been obtained, the SCP is formulated to find the least number of viewpoints that cover every surface patch at least once:

$$\min \sum_{i=1}^{n} x_i, \quad \text{where } x_i \in \{0, 1\}, \tag{4.1}$$

$$s.t. \quad \mathbf{A} \cdot \mathbf{x} \geq \mathbf{c}, \tag{4.2}$$

where x_i is the i^{th} viewpoint, $x_i = 1$ means this viewpoint is selected and $x_i = 0$ means it is not selected; $\mathbf{A}$ is $m \times n$ visibility Matrix; the result of $\mathbf{A} \cdot \mathbf{x}$ is a $m \times 1$ vector, which indicates how many times each surface patches are covered; $\mathbf{c}$ is the measuring frequency, in this case, is $m \times 1$ vector with all 1s.

In this chapter, we reformulate the SCP to a partial Set Covering Problem for practical consideration. In the partial SCP formulation, instead of full coverage, a given coverage ratio δ of the target building is required. We use $\delta = 0.99$ in the work, which indicates 99.0 % coverage of the target buildings are required. The partial SCP formulation is shown as follows:

$$\min \sum_{i=1}^{n} x_i, \quad \text{where } x_i \in \{0, 1\} \tag{4.3}$$

$$\text{s.t.} \quad \sum_{i=1}^{m}((\sum_{j:x_j \neq 0}^{n} A_{ij}) \geq 1) \geq \delta * m, \tag{4.4}$$

where constraint (4.4) is the partial coverage constraints, which indicates that at least $\delta \times 100\%$ of the total surface patches should be covered in the optimization formulation.

Solve Partial SCP Using Random-Key Genetic Algorithm and Greedy Search

Several methods can be applied to solve the Set Covering Problem and its extensions, such as Greedy search algorithm [57], Dynamic Programming [58], and Genetic Algorithm [59].

In this chapter, Genetic Algorithm has been used in combining with greedy search because the result is good and the computational cost is reasonable. In order to handle the coverage constraint effectively, we use random-key to encode the information, and a Random-Key Genetic Algorithm (RKGA) [69][70] is used to solve the partial SCP. We use real number range from 0 to 1 as random keys to encode the information. The random keys are stored in the genes of the chromosome. The decoding process is to sort the genes by value. The fitness evaluation process adds the sorted gene to the solution set one-by-one, until the coverage constraint is satisfied. The objective is to minimize the number of viewpoints required for the given coverage ratio. In this chapter, a coverage ratio of 99.0% is used.

MATLAB Genetic Algorithm solver [71] has been used to solve partial SCP in this chapter. The number of chromosomes is set to 2,000 and the number of iteration is set to 240. These two parameters ensures that enough computational resources are allocated to GA such that the final results would be converged and consistent. In this chapter, the mutation rate is set to 0.3; and the

crossover rate is set to 0.7.

We also make use of the Greedy search [9] results of the partial SCP. In our method, a greedy search is first performed to find a solution of the partial SCP, then initial population of RKGA is randomly sampled near to the greedy search results. Combining the RKGA and greedy search method leads to better results and makes the RKGA converge faster.

4.3 Computational Experiments, Results and Discussion

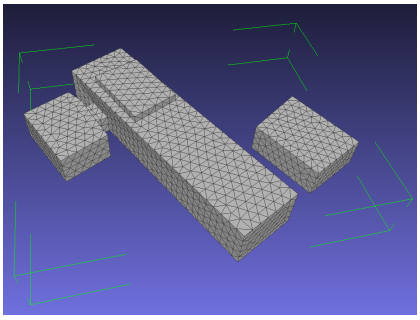
In the computational experiment, We have compared our method with two previous methods using four different sets of target buildings, with different sensor parameters. The results shown in this section demonstrate that our method outperforms the two previous methods in terms of the number of required viewpoints, as well as the coverage ratio, under different sensor parameters and different numbers of candidate viewpoints.

4.3.1 Setup

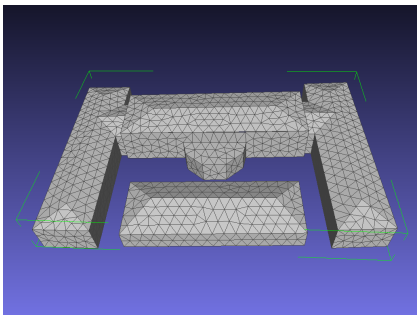
Figure 4.8 shows four different sets of target buildings used for testing the proposed method. We also vary the sensor parameters including viewing angle and FOD. In this chapter, we use 30 meters, 50 meters as the viewing ranges in our experiments, similar to our previous work [10]. In the computational experiments, we compare our method with the two previous methods:

- The "optimal scan zone" or "Offset" method [33] generates a set of candidate viewpoints directly on every surface patches of the building surfaces. and offsets them with a given distance. The viewing direction is defined so that it points to corresponding surface patch.
- The randomized sampling-based method [9][10] generates a set of candidate viewpoints by randomly sampling around the target buildings, within the range of the maximum FOD.

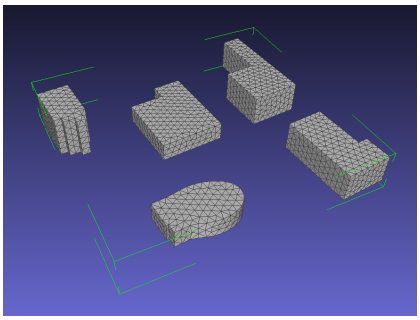
The candidate viewpoints are generated with the three different methods mentioned above, and the same optimization method is used in the second step. The computational time is about 20



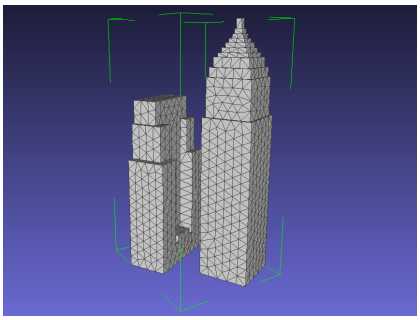
(a) Target building 1: bounding box size is $64 \times 79 \times 13$ meters



(b) Target building 2: bounding box size is $106 \times 71 \times 15$ meters



(c) Target building 3: bounding box size is $85 \times 79 \times 14$ meters



(d) Target building 4: bounding box size is $85 \times 100 \times 180$ meters

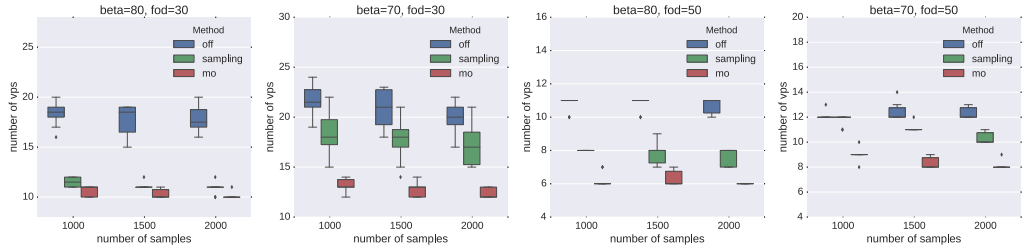
Figure 4.8: The polygonal geometric model and the bounding box of the target buildings

to 30 minutes with a Core i7 computer with 8 GB RAM. During the computational experiments, each method with different parameters is run 10 times in order to get a statistical comparison.

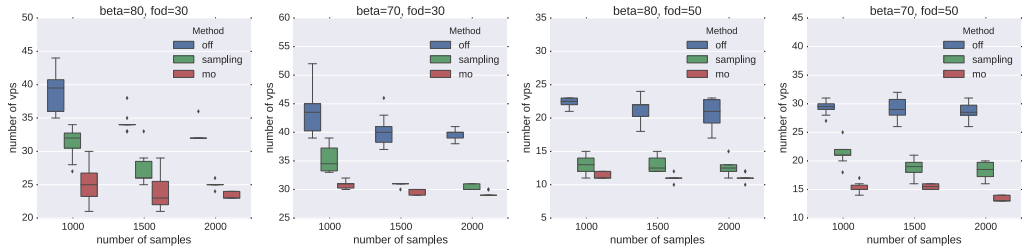
4.3.2 Results

Figure 4.9 and 4.10 compare the performance of our method with the two previous methods. It shows that our method consistently finds the least number of viewpoints and best coverage with different sensor parameters and different number of candidate viewpoints.

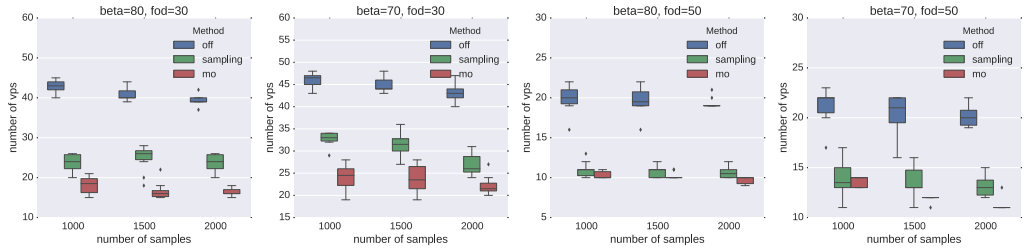
As shown in Figure 4.9, the proposed method finds 38.7%, 38.9%, 49.0%, 32.0% fewer viewpoints for the required coverage compared to the optimal scan zone method [33], and 19.9%, 14.4%, 17.6%, 12.1% fewer viewpoints for the required coverage compared to the randomized



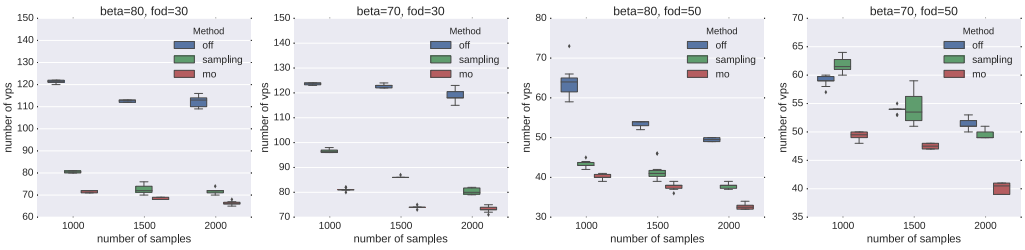
(a) Target building set 1



(b) Target building set 2



(c) Target building set 3



(d) Target building set 4

Figure 4.9: Number of resultant viewpoints for the required coverage. The proposed method is labelled as "mo" (shown in red); the randomized sampling-based method is labelled as "sampling" (shown in black); the "Optimal Scan Zone" or "Offset" method is labelled as "off" (shown in blue). beta is the maximum viewing angle, fod is the maximum viewing range. The y-axis is the number of required viewpoints, while the x-axis is the number of sampled candidate viewpoints

sampling based method [9][10] for the four target buildings respectively. Overall, the number of viewpoints required is **39.7%** less than that of optimal scan zone method and **16.0%** less than that of the recently proposed sampling-based method on average among the four sets of target buildings. The 3D visualization of the resultant viewpoints and the target buildings are shown in Figure 4.11.

Figure 4.10 shows the ratios of uncovered areas of the three methods with different sensor parameters. It shows that the proposed method consistently achieves less uncovered areas, which means better coverage ratio, as compared to the other two methods. That is mainly because the proposed MO-based method explores the space more effectively with the geometric information of the target buildings.

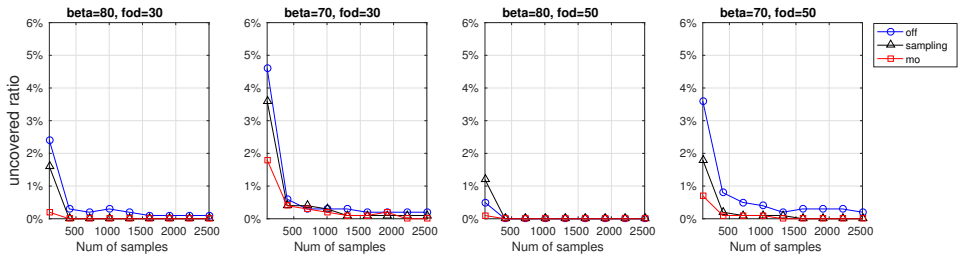
Based on the results shown above, we conclude that the proposed method works well for model-based view planning problem. Compared to existing methods, the proposed method has demonstrated better results with higher coverage ratios and fewer viewpoints required.

4.3.3 Discussion

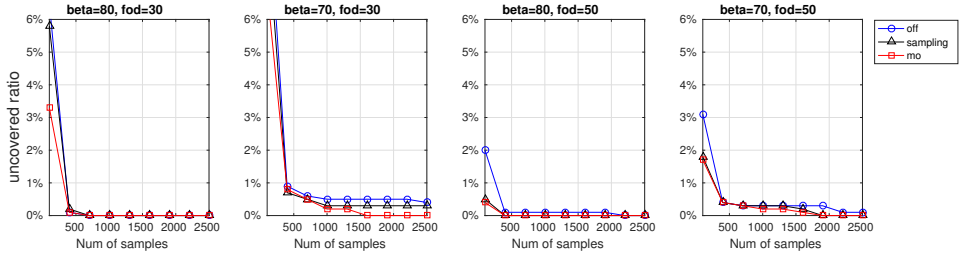
The results obtained in the previous section show that the MO-based method performs better than the other two methods, and there are several reasons for that.

The optimal scan zone method has demonstrated good coverage ratios for most target buildings. However, the results are less optimal, because it cannot identify and make use of important geometric features such as corners and edges, which limit its performance when there are many of them. In addition, incomplete coverage problem still exist because invalid viewpoints may be generated by certain non-convex geometries. The randomized sampling method usually leads to better results compared to the optimal scan zone method. However, sampling the viewpoints randomly uses less geometric information of the target buildings, which makes the candidate viewpoints generation process less effective.

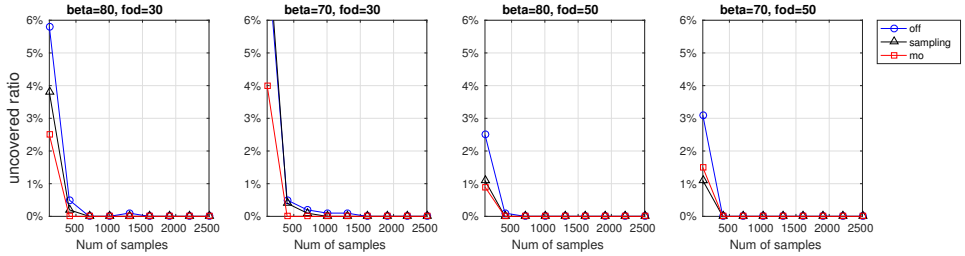
The MO-based method overcomes the above-mentioned problems. Generating candidate



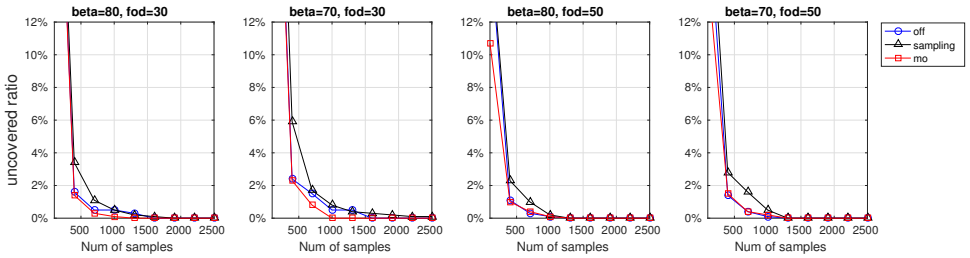
(a) Target building set 1



(b) Target building set 2



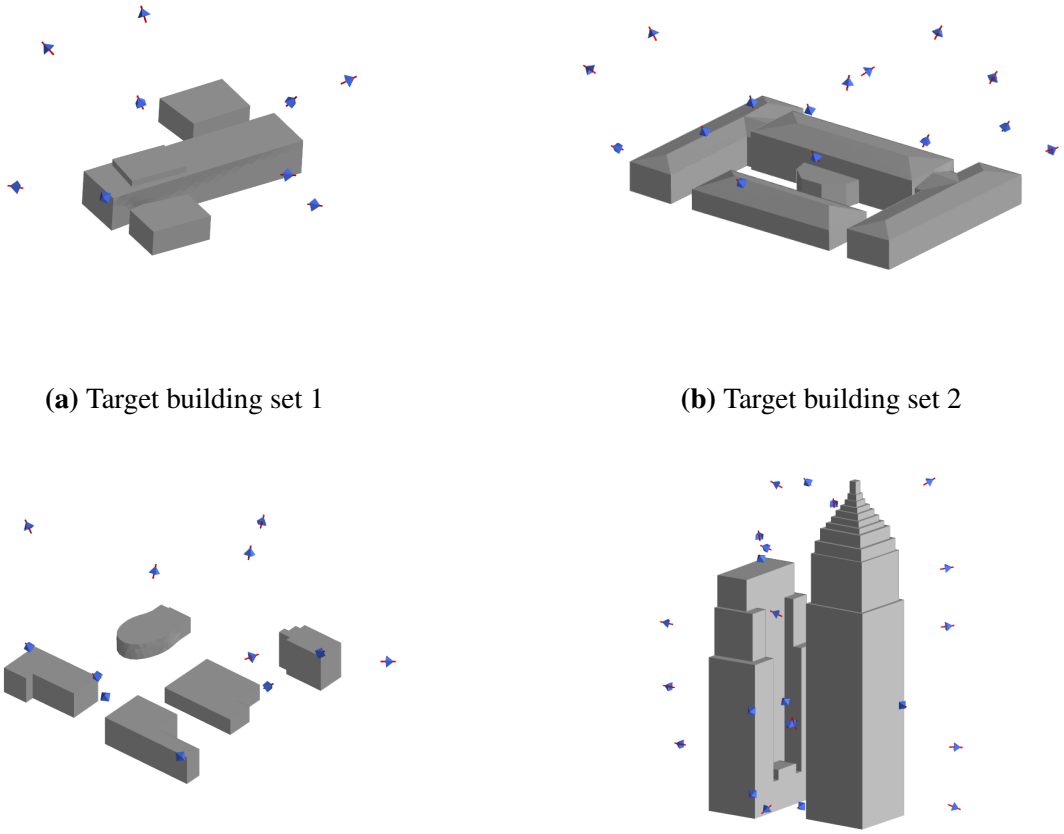
(c) Target building set 3



(d) Target building set 4

Figure 4.10: Percentage ratios of uncovered areas with different number of sampled candidate viewpoints. Smaller number indicates higher coverage ratio.

viewpoints by using Gaussian sampling around the MO of a dilated volume ensures that all the viewpoints obtained are valid in feasible space. In addition, MO is the skeleton of a dilated volume, which well preserves the geometric features of target buildings, yielding better results.



(a) Target building set 1

(b) Target building set 2

(c) Target building set 3

(d) Target building set 4

Figure 4.11: 3D visualization of the resultant viewpoints and the target buildings

4.4 Summary

This chapter presents a novel model-based view planning method for UAV building inspection and surveillance applications. Voxel dilation, Medial Object (MO) and Gaussian sampling methods are used to generate the candidate viewpoint set. The visibility information is encoded in the visibility matrix, and the problem is formulated as partial Set Covering Problem, which is then solved by Random-Key Genetic Algorithm and Greedy search methods. We have also demonstrated that the proposed method is able to find better solutions (fewer viewpoints and higher coverage ratio) compared to previous methods.

Chapter 5

View Planning for Building Reconstruction with UAV

In this work, we extend the viewpoint planning method to the planning problem for 3D building reconstruction when the 2D map data of the target building is available. The proposed method uses this preliminary 2D map data to assist in finding a series of suitable viewpoints for 3D reconstruction tasks for large scale outdoor building structures [10].

5.1 Introduction

In this chapter, we present a novel view planning method to tackle the planning problem for 3D building reconstruction when the 2D map data of the target building is available. The proposed method uses the preliminary map data to assist in finding a series of suitable viewpoints for 3D reconstruction tasks for large scale outdoor building structures.

Unlike model-based and non-model-based view planning methods presented in introduction, the view planning application discussed in this chapter is different in that only a limited amount of geometric information of the target object is available beforehand. With this in mind, we can describe the view planning problem presented in this chapter as a "semi-model-based" view planning problem. This prior information comes in the form of:

- 2D longitude and latitude information of each perimeter vertex of the target buildings,
- a rough estimation of building height based on the number of stories in the building,
- the specifications of the UAV's on-board camera sensor, and
- the requirements from the reconstruction application (e.g. the maximum viewing angle, the coverage frequency, etc.).

The view planning problem discussed in this chapter is now defined as finding the minimum number of viewpoints required for the 3D shape reconstruction of outdoor buildings, based on prior information.

5.2 Approach

The view planning method presented in this chapter involves two major steps, as outlined in Figure 5.1. Firstly, a rough model of the target building is generated from freely available online 2D map data. The longitude and latitude coordinates of the 2D footprint are mapped to Cartesian coordinates in Euclidean space, and a rough 3D model of the target building is then completed by extruding the footprint in Cartesian coordinates by the building's estimated height. This rough 3D model of the target building is used as an input to the second phase. A "generate-and-test" view planning approach is used to find suitable viewpoints whilst adhering to the requirements of the application and sensor specifications.

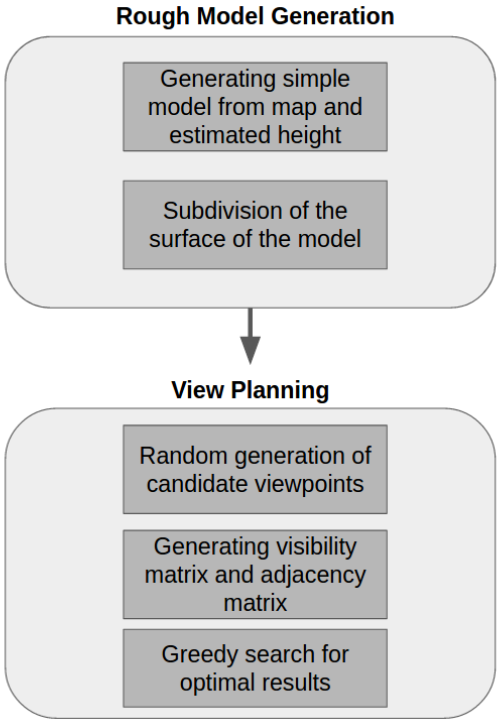


Figure 5.1: The outline of proposed method

5.2.1 Generate 3D Model of Buildings

The rough 3D polygonal model is generated in three steps using 2D map data and an estimated building height. First, the GPS coordinate information of each perimeter vertex of the target building are extracted from an online map. Then, the coordinates of each vertex is transformed from longitude and latitude to Cartesian coordinates using the Haversine formula [72] shown in Eqn. (5.1). The conversion is performed by selecting an arbitrary vertex, v_o, as the origin. x/y positions in Cartesian coordinates for the remaining vertices are then calculated as the distance along the longitude/latitude directions between themselves and v_o using Eqn. (5.1). In the third step, using an estimated height for each vertex, the 3D rough model is generated via extrusion, as shown in Figure 5.2.

$$d = 2r \arcsin\left[\sin^2\left(\frac{\alpha_2 - \alpha_1}{2}\right) + \cos(\gamma_1)\cos(\gamma_2)\sin^2\left(\frac{\gamma_2 - \gamma_1}{2}\right)\right]^{\frac{1}{2}} \tag{5.1}$$

where r is the radius of the earth. α_1, α_2 and γ_1, γ_2 represent the longitude and latitude information of two given points.

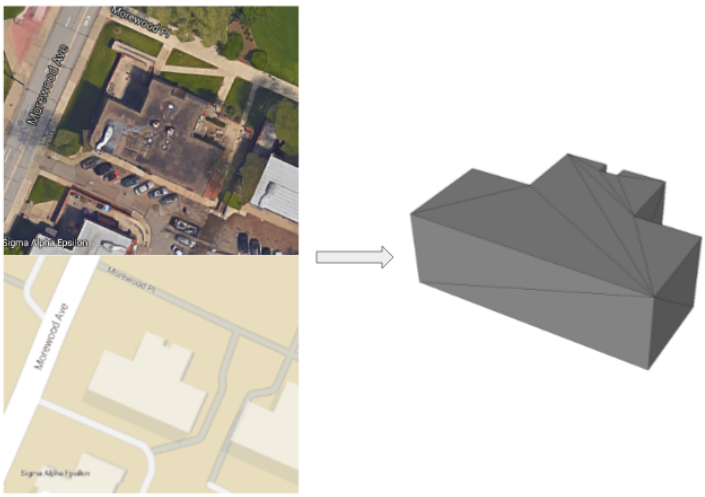


Figure 5.2: 2D map to 3D model; the two pictures on the left are the satellite image and 2D map data; the picture on the right are the 3D polygonal model generated from map data.

At this stage of the view planning process, the 3D polygonal model generated in the previous step is not suitable due to the existence of large variations in patch size and also low quality prop-

erties of its shape (e.g. sharp, narrow triangles present in the surface mesh). To overcome this, the Bubble Mesh [53] method is used to subdivide the model's surface patches into a well-shaped triangular mesh that preserves the original shape well, and also maintains patch uniformity. An example result of the subdivision process on an example target building model is shown in Figure 5.3.

(a) Original model with 34 triangles

(b) The model with 1187 triangles after subdivision using Bubble Mesh[53]

Figure 5.3: Subdivision of the target building

The view planning method proposed in this chapter is adapted from the "generate-test" framework described in previous literature [30]. First, a large sample of candidate viewpoints is generated around the target building model. To solve the view planning problem, a smaller subset of viewpoints is selected from this candidate set via combinatorial optimization, based on a series of problem-specific requirements.

5.2.2 Generate Candidate Viewpoints

In the candidate viewpoints generation step, a set of redundant viewpoints is generated for use in the later selection process. In this chapter, candidate viewpoints are randomly generated around the target building within a distance smaller than the maximum viewing range of the UAV's camera. The candidate viewpoints generation method is adopted from previous work[9]. Once

the generation of the candidate set is completed, visibility testing is performed between each surface patch of the target building model and each member of this candidate viewpoint set. Additionally, each pair of viewpoints in the candidate set is evaluated in terms of image registration feasibility constraints.

5.2.3 Generate Visibility Matrix and Adjacency Matrix

Before the viewpoint selection process can commence, a prepossessing step which introduces two matrices is performed. The visibility matrix $\mathbf{A}_{vm}$ is a 2D binary matrix which provides visibility information between each surface patch of the target model and viewpoints belonging to the candidate set. The adjacency matrix $\mathbf{A}_{am}$ is a 2D binary matrix that is used to measure connectivity information between viewpoints. The purpose of $\mathbf{A}_{am}$ is to provide a measure which will ensure the success of the final image registration process.

The visibility matrix is used as the basis for the SCP to find the most suitable set of viewpoints to solve the view planning problem. Visibility between a given viewpoint and a surface patch is evaluated by the following conditions:

- The surface patch must be located within the camera's Field-Of-View (FOV) (checked using a perspective projective camera model[56] to project surface patches onto the image plane of the camera) and Field-Of-Depth (FOD).
- The viewing angle must be less than a specified maximum.
- There must be a clear line of sight, i.e no physical blockages or occlusions between the viewpoint and surface patch (this occlusion condition is checked via a fast ray-triangle intersection algorithm[55]).

The resultant visibility matrix $\mathbf{A}_{vm}$ is a $n \times m$ matrix, where m and n represent the number of viewpoints and number of surface patches, respectively. The Boolean information encoded in the matrix represents the visibility between a given surface patch and viewpoint pair.

The adjacency matrix encodes connectivity information between viewpoints. Connectivity

is evaluated as the feasibility of an image registration process between two viewpoints. In the $m \times m$ adjacency matrix $\mathbf{A}_{am}$, two viewpoints are considered connected if the following two conditions are satisfied:

- The distance between two viewpoints must be smaller than D_{th}.
- There is at least λ_{min} overlapped viewing area shared by the two viewpoints.

Once the visibility matrix and the adjacency matrix are generated, the selection process via combinatorial optimization can commence.

5.2.4 Modified Set Covering Problem with Constraints

In order to address uncertainties present in the rough building model, and issues relating to the image registration process, the view planning problem discussed in this chapter is formulated as a modified SCP with additional constraints. In this circumstance, full coverage is not required. Instead, a certain coverage ratio, δ_d, is introduced, along with an additional constraint to ensure the resultant viewpoint set is fully connected. This ensures that the image registration process is performed to a suitably high quality. The SCP in this chapter is formulated to find the least number of viewpoints that cover every surface patch at least γ times:

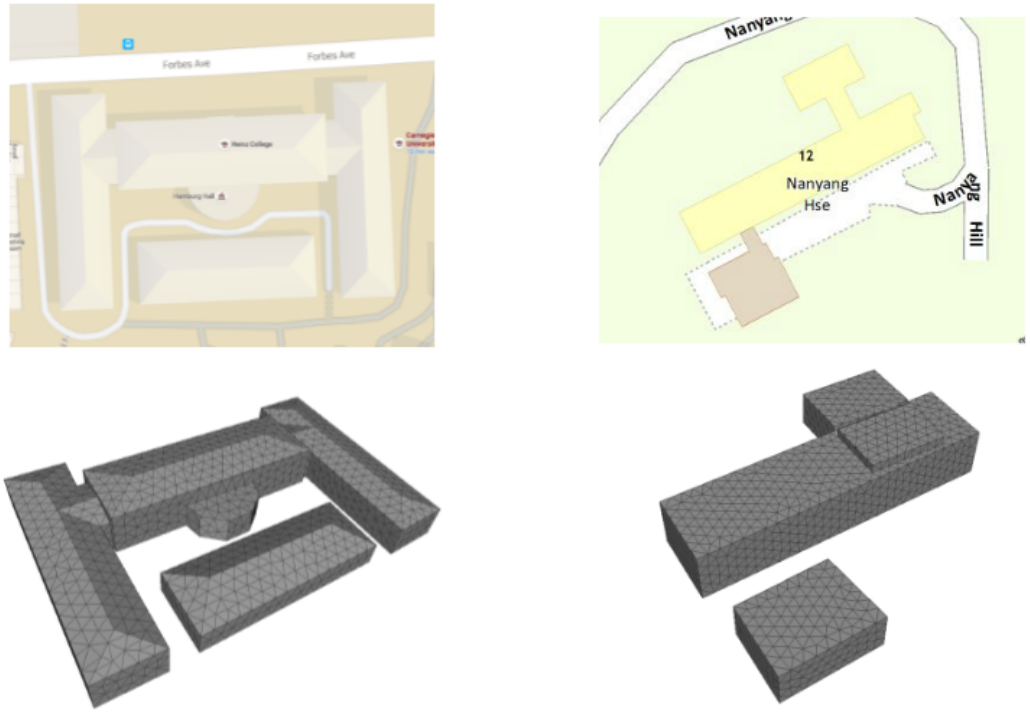
$$\begin{aligned} & \min \sum_{i=1}^{n} x_i, \quad \text{where } x_i \in \{0,1\} \\ \text{s.t.} \quad & \sum (\mathbf{A}_{vm} \cdot \mathbf{x} \geq \gamma) \geq \delta_d m \\ & \text{There exists a path between } (x_i, x_j), \forall x_i > 0, x_j > 0, \end{aligned}$$

where x_i corresponds to the i^{th} viewpoint; $x_i = 1$ denotes that the viewpoint is selected, whilst $x_i = 0$ infers that it is not. γ represents the required coverage frequency (the minimum number of times each surface patch must be captured from different viewpoints) and δ_d is the desired coverage ratio. The first constraint presented in the formulation is to enforce δ_d of the surface of the target building is covered at least γ times. The second constraint ensures the resultant

viewpoint set is fully connected and the 3D multi-view reconstruction process will then generate a full single model of the target object, as opposed to several separate partial models.

The combinatorial optimization problem formulated in this chapter is solved with the neighbourhood greedy search algorithm proposed in Alg. 4. The method continuously searches for the viewpoint with maximum coverage in the neighbourhood of the solution set. The neighbourhood is defined as the viewpoints connected to the current solution set. If such a viewpoint is found, it is added to the solution set. This process continues until the specified coverage requirements are met.

5.3 Results and Discussion



(a) Hamburg Hall in CMU

(b) Nanyang House in NTU

Figure 5.4: Target buildings

To validate the view planning method proposed in this chapter, we have conducted two com-

Algorithm 4 Neighborhood greedy search algorithm for modified SCP with constraints

Input: The set of triangular surface patches, $\mathbb{P}$; the set of candidate viewpoints, $\mathbb{V}$; the visibility matrix, $\mathbf{A}_{vm}$; the adjacency matrix, $\mathbf{A}_{am}$; the required coverage frequency, γ; and the desired coverage ratio, δ_d.

Output: The resultant coverage ratio, δ; the number of resultant viewpoints, n_{res}; and the resultant viewpoints set, $\mathbb{V}_{res}$.

$\mathbb{V}_{res} \leftarrow \emptyset$

$n_{numOfTriangles} =$ sizeof($\mathbb{P}$)

$\mathbb{P}_{uncover} \leftarrow$ findUncovered($\mathbf{A}_{vm}, \gamma$)

$\mathbb{P} \leftarrow$ delete($\mathbb{P}$, $\mathbb{P}_{uncover}$)

$\mathbb{V}_{res} \leftarrow$ append($\mathbb{V}_{res}$, random($\mathbb{V}$))

while $\mathbb{P} \neq \emptyset$ **and** $\delta < \delta_d$ **do**

    $\mathbb{V}_{neighbour} \leftarrow$ Neighborhood($\mathbb{V}_{res}, \mathbf{A}_{am}$)

    $v \leftarrow$ maxCover($\mathbb{V}_{neighbour}, \mathbb{P}, \mathbf{A}_{vm}$)

    $\mathbb{V}_{res} \leftarrow$ append($\mathbb{V}_{res}, v$)

    $\mathbb{V} \leftarrow$ delete($\mathbb{V}$, v)

    for each $p_j \in \mathbb{P}$ **do**

        if count($p_j, V_{res}, \mathbf{A}_{vm}$) $> \gamma$ **then**

            $\mathbb{P} \leftarrow$ delete($\mathbb{P}$, p_j)

        end if

    end for

    δ = sizeof($\mathbb{P}$)$/n_{numOfTriangles}$

end while

n_{res} = sizeof($\mathbb{V}_{res}$)

return $\delta, n_{res}, \mathbb{V}_{res}$

putational tests and one field test. These tests are performed on two separate target buildings. The first building, the Hamburg Hall, shown in Figure 5.4-(a), is located on the Carnegie Mellon University (CMU) campus in USA. The second target building, the Nanyang House, shown in Figure 5.4-(b), consists of a large three-storey building with two peripheral two-storey buildings, and is located on the Nanyang Technological University (NTU) campus in Singapore. Computational tests are performed by applying the view planning method outlined in 4 to rough 3D models generated for these two building structures. A field test is then conducted on the Nanyang House structure as well, where a UAV with on-board camera equipment is used to capture a single 2D image at each planed viewpoint. This set of images is then used to reconstruct a more detailed 3D model of the target building.

5.3.1 Computational and Field Experiment

The UAV used in this experiment is a DJI Phantom 3 [73]. The intrinsic parameters of the on-board camera are derived via standard chessboard calibration methods[74]. Eleven pictures of the 9×7 chessboard are taken at different poses; the resultant calibrated intrinsic matrix $\mathbf{K}$ is given as:

$$K = \begin{bmatrix} 2337.76 & 0 & 1991.83 \\ 0 & 2346.33 & 1466.06 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.2}$$

The remaining view planning parameters used in this chapter are listed in Table. 5.1.

The rough 3D polygonal model (shown in Figure 5.4), the camera intrinsic matrix (shown in Eqn. (5.2)), and the parameters in Table. 5.1 are used as input for the view planning method outlined in Fig 5.1. For Hamburg Hall, a total number of $2,858$ candidate viewpoints are generated; and for Nanyang House, a total number of $2,949$ candidate viewpoints are produced from our method. The results with different required coverage frequencies are shown in Table. 5.2. The 3D visualization of the planned viewpoints and the target buildings are shown in Figure 5.5 and

Table 5.1: View planning parameters

Viewing Angle	$75°$
FOD (meters)	$(1, 50)$
Required Coverage Frequency	$\gamma = 2, 3, 4$
Required Coverage Ratio	$\delta_d = 99.5\%$
Minimum Overlapping	$\lambda_{min} = 5$ patches
Maximum Connected Distance	10 m
Minimum Viewpoint Height	10 m

5.6.

Table 5.2: Comparison of Results with Different Required Coverage Frequency

	Number of viewpoints required		
	$\gamma = 2$	$\gamma = 3$	$\gamma = 4$
Hamburg Hall Building	33	52	68
Nanyang House Building	26	41	55

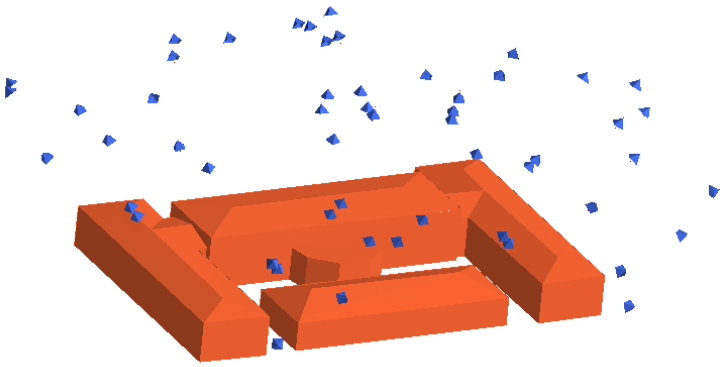


Figure 5.5: 3D visualization of the Hamburg Hall building and the planned viewpoints ($\gamma = 3$)

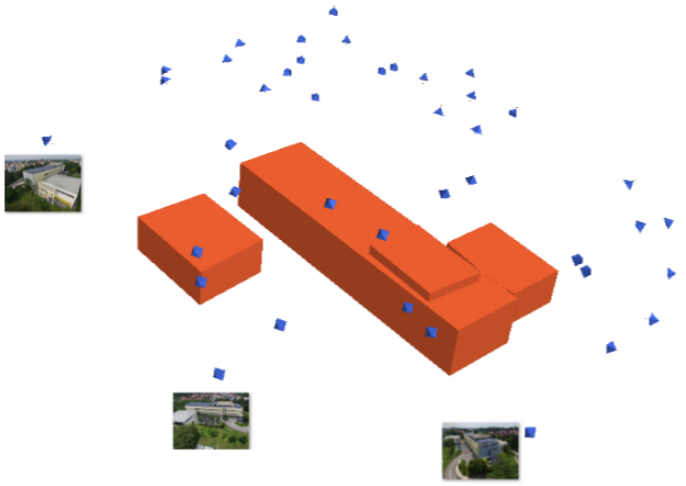
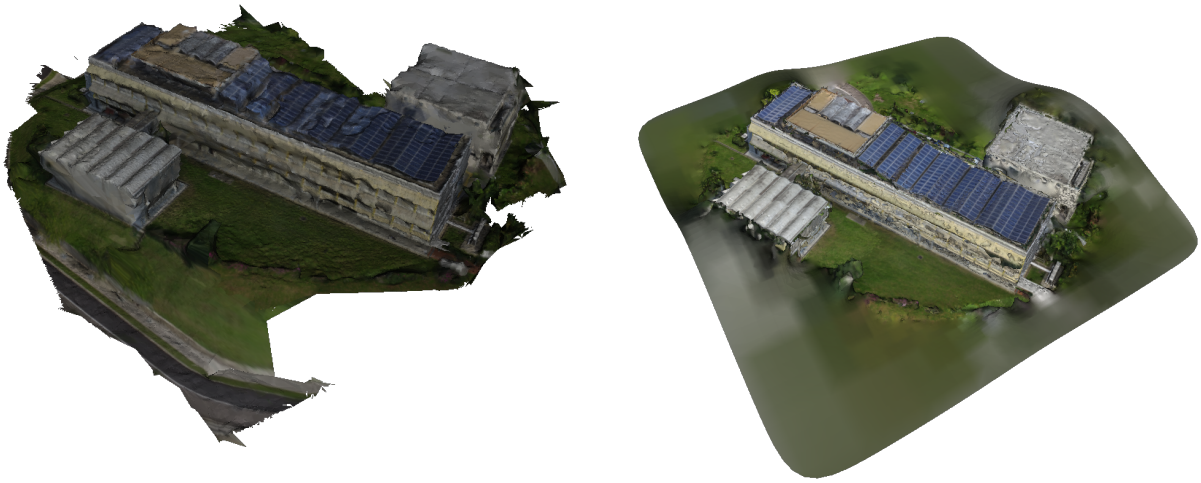


Figure 5.6: 3D visualization of the Nanyang House building, the planned viewpoints and examples of pictures taken at some planned viewpoints in a later field experiment ($\gamma = 3$)

Figure 5.7: The pictures taken at planned viewpoints

A field experiment was conducted on the Nanyang House structure by flying the UAV to capture a single 2D image at each planned viewpoint, as shown in Figure 5.6. The resultant collection of images, shown in Fig 5.7, is then used to reconstruct a 3D model of the target building. The 3D reconstruction is performed using the Structure From Motion (SFM) workflow. In this chapter, we use VisualSFM [60][75] for feature finding, feature matching, image registration and camera pose estimation. With the SIFT feature descriptor used in this chapter, 39 out of

41 planned viewpoints are successfully registered for 3D multi-view dense reconstruction. Two different free software packages, CMP-MVS [62] and Multi-View Environment (MVE) [76], are used for the dense reconstruction and surface reconstruction tasks with these images. The results of these reconstructions are presented in Figure 5.8a and 5.8b.



(a) The 3D reconstruction result by CMP-MVS[62]

(b) The 3D reconstruction result by MVE [76]

Figure 5.8: The 3D reconstruction results

5.3.2 Discussion

The experiments conducted in this chapter indicate that the proposed view planning method is able to identify suitable viewpoints for large-scale outdoor 3D building reconstruction tasks. The 3D models of the target buildings are successfully reconstructed, and show surface details of the target buildings. It is worth noting that the quality of the reconstructed model could be further improved by adjusting certain view planning parameters. For example, a smaller viewing angle, larger required coverage frequency, or smaller maximum FOD may lead to reconstruction results with higher surface detail. However, one must acknowledge that the natural downside of this

increment in detail is that more individual viewpoints would be generated in the solution, an example of this is shown in Table. 5.2.

Chapter 6

Calibration for Accurately Positioning the Vision Sensor

For industrial inspection application with high accuracy requirement, it is important to position the vision sensor to the planed viewpoints accurately. In this chapter, we proposed a novel learning-based robotic calibration method that minimizes the kinematic error and find the relative pose between the 3D scanner mounted on the end-effector and the Tool-Center-Point (TCP) of the robot. The work has been published on [13].

6.1 Introduction

Robotic calibration work has been studied for many years with many formulations. In this chapter, we use local POE method [77] to compensate the geometric error of the kinematics, and Gaussian Process to compensate the residual error. In this work, a novel calibration method is proposed where the POE model is used to model the robot kinematics, and the parameters within the model are calibrated based on measurement data. Gaussian Process (GP) regression is used to compensate for the residual errors. Joint deflections due to the payload are considered in the GP model to further reduce the pose error. The GP used in this chapter is based on Bayesian inference and is suitable for regression problems with a small number of feature sizes and a small amount of training data, while Nerual Network (NN)[78] usually work well when the training data is large. For robot calibration application, usually a few hundred measurement data points will be collected as training data. Thus, GP is expected to work better than NN for this application with small amount of training data.

The goal of robot calibration is to identify the robot model that minimizes the difference between the computed and measured end-effector pose. Given a robot with n joints and the kinematic model f, the theoretical or computed pose $\mathbf{T}$ of end-effector is given by:

$$\mathbf{T} = f(\mathbf{q}), \quad \mathbf{T} \in SE(3),$$

where $\mathbf{q} \in \mathbb{R}^{n\times 1}$ is the joint angles; n is the number of joints of the robot; $SE(3)$ stands for *Special Euclidean Group*.

However, in reality, the true kinematic model is slightly different from the idealized model, which causes pose error of the end-effector. Because such pose error affects the robot accuracy, calibration is required to modify the robot kinematic model to minimize the pose error of the end-effector.

The calibration is to find an accurate kinematic model f such that the difference between $f(\mathbf{q})$ and measurement $\mathbf{T}'$ is minimized:

$$\begin{aligned} &\min \sum \|\mathbf{T}'\mathbf{T}^{-1} - \mathbf{I}\| \\ =&\min_{\mathbf{f}} \sum \|\mathbf{T}' f(\mathbf{q})^{-1} - \mathbf{I}\|, \end{aligned}$$

where $\mathbf{I} \in \mathbb{R}^{4\times4}$ is the identity matrix.

In this chapter, GP is used to compensate for the residual errors and also to consider the effects of a payload at the end-effector. Thus the new overall kinematic model $\bar{f}$ can be written as $\bar{f}(\mathbf{q}, m) = G(\mathbf{q}, m)f(\mathbf{q})$ by appending GP in addition to the previous kinematic model f. Then the formulation used in this chapter is shown as follows:

$$\begin{aligned} &\min \sum \|\mathbf{T}'\mathbf{T}^{-1} - \mathbf{I}\| \\ =&\min_{\bar{\mathbf{f}}} \sum \|\mathbf{T}' \bar{f}(\mathbf{q}, m)^{-1} - \mathbf{I}\| \\ =&\min_{\mathbf{G},\mathbf{f}} \sum \|\mathbf{T}' f(\mathbf{q})^{-1} G(\mathbf{q}, m)^{-1} - \mathbf{I}\|, \end{aligned}$$

where m is the payload of the end-effector, and $G(\mathbf{q}, m) = \mathbf{T}_g \in SE(3)$ is the homogeneous transformation representation of the GP model ($g(\mathbf{q}, m) \in \mathbb{R}^{6\times1}$). In the proposed method, both the POE model and GP model are calibrated to minimize the errors between the measured and computed end-effector poses, the details of POE model and GP model is explained in later section.

6.2 Approach

This chapter presents a novel two-step calibration method that minimizes the robotic kinematic error. In the first step, the proposed method utilizes a POE model-based calibration to identify a set of model parameters that minimizes the errors between the computed and measured end-effector poses. Subsequently, GP is used to compensate for residual errors taking into account of payload at the end-effector.

6.2.1 POE Model Calibration

The POE calibration method is adopted from previous work [77]. Unlike the DH model, the POE model effectively models transformation between arbitrary frames, which makes it suitable to eliminate arbitrary misalignments in adjacent frames during calibration.

In the POE model, ${}^{i}\mathbf{T}_{i-1} \in SE(3)$ is a homogeneous transformation matrix that models the geometric relationship between adjacent frame i and $i-1$. Thus, the POE representation of the transformation matrix between two adjacent frames with a joint rotation of q_i is defined in the following form:

$$ {}^{i}\mathbf{T}_{i-1}(q_i) = {}^{i}\mathbf{T}_{i-1}(0)e^{\mathbf{s}_i q_i}, \tag{6.1} $$

where $\mathbf{s}_i \in se(3)$ is the twist that models the rotational/prismatic joint axis and ${}^{i}\mathbf{T}_{i-1}(0) \in SE(3)$ is the initial pose of the frame i, which could also be represented as $e^{\hat{\mathbf{p}}_i} = {}^{i}\mathbf{T}_{i-1}(0), \hat{\mathbf{p}}_i \in se(3)$. Thus, Eq. (6.1) can be rewritten as:

$$ {}^{i}\mathbf{T}_{i-1}(q_i) = e^{\hat{\mathbf{p}}_i}e^{\mathbf{s}_i q_i}. \tag{6.2} $$

The transformation matrix describing the relative pose of frame $n+1$ in frame 0 of a n-jointed robot is given by:

$$ {}^{0}\mathbf{T}_{n} = \prod_{i=1}^{n}({}^{i}\mathbf{T}_{i-1}(0)e^{\mathbf{s}_i q_i}){}^{n}\mathbf{T}_{n+1}(0) = \prod_{i=1}^{n}(e^{\hat{\mathbf{p}}_i}e^{\mathbf{s}_i q_i})e^{\hat{\mathbf{p}}_{\mathbf{n+1}}}, \tag{6.3} $$

where $\prod$ is defined as a non-commutative product of matrices:

$$\prod_{i=1}^{k} \mathbf{X}_i = \mathbf{X}_1 \mathbf{X}_2 ... \mathbf{X}_k$$

The goal of POE model calibration is to identify a set of parameters $\mathbf{P} = [\mathbf{p}_1^T, \mathbf{p}_2^T, ..., \mathbf{p}_{n+1}^T]^T \in \mathbb{R}^{6(n+1)\times 1}$ such that the error between the computed and measured end-effector poses is minimized. Note that $e^{\hat{\mathbf{p}}_i} = {}^{i}\mathbf{T}_{i-1}(0)$ models the zero pose transformation matrix of frame i to $i-1$ when $q_i = 0$; $\mathbf{p}_i \in \mathbb{R}^{6\times 1}$ is the vector representation of $\hat{\mathbf{p}}_i$.

Then the POE model calibration in the first step is formulated as a linear model $\mathbf{Y} = \mathbf{A}\mathbf{x}$ by taking the difference between the measured pose $\mathbf{T}'$ and the computed pose $\mathbf{T}$, where

$$\mathbf{Y} = [\mathbf{y}_1^T, \mathbf{y}_2^T, \cdots, \mathbf{y}_k^T]^T \in \mathbb{R}^{6k\times 1}$$

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_k \end{bmatrix} \in \mathbb{R}^{6k\times 6(n+1)}$$

$$\mathbf{x} = [\delta\mathbf{p}_1^T, \delta\mathbf{p}_2^T, ..., \delta\mathbf{p}_{n+1}^T]^T \in \mathbb{R}^{6(n+1)\times 1},$$

where k is the number of measured poses and corresponding joint angles used for calibration, n is the number of joints in the robot, and

$$\mathbf{y}_i = log({}^{0}\mathbf{T}'_n \cdot {}^{0}\mathbf{T}_n^{-1})^V \quad \mathbf{y}_i \in \mathbb{R}^{6\times 1}$$

$$\mathbf{A}_i = [\mathbf{Ad_0}, \mathbf{Ad_1}, ..., \mathbf{Ad_{n+1}}] \in \mathbb{R}^{6\times 6(n+1)}$$

$$\mathbf{y}_i = \mathbf{A}_i\mathbf{x},$$

where ${}^{0}\mathbf{T}'_n$ is the measured end-effector pose, and ${}^{0}\mathbf{T}_n$ is the computed pose using the robot kinematic model; $\log(\mathbf{T})^V$ converts the homogeneous transformation matrix $\mathbf{T} \in SE(3)$ to a $\mathbb{R}^{6\times 1}$ vector representation of the position and orientation [77]; $\mathbf{y}_i$ indicates the difference between the measured and computed poses; $\mathbf{x}$ is the update to the model parameters to minimize $\mathbf{y}_i$; $\mathbf{A_m}$ is the matrix mapping the model parameter errors to the end-effector errors; and, $\mathbf{Ad_i}$ is the adjoint representation of ${}^{i}\mathbf{T}_{i+1}$ [79] [77].

Let $\mathbf{A}^*$ be the pseudoinverse of $\mathbf{A}$, then $\mathbf{x}$ can be obtained by using $\mathbf{x} = \mathbf{A}^*\mathbf{Y}$. A new set of model parameters $\mathbf{p}$ can be obtained by updating $\mathbf{p}$ with $\mathbf{p} = \mathbf{p} + \delta\mathbf{p}$. This update is done iteratively until the errors $\mathbf{y}_i$ converge or a pre-defined number of iterations has passed.

6.2.2 GP Regression for Residual Error

Gaussian Process [80] is a commonly used machine learning method for regression problems. It has been used in robotic kinematics modelling in previous work [81]. GP models the observations as multivariate Gaussian distribution, thus the output of any new input could be predicted through Bayesian inference.

Given $(\mathbf{x_i}, y_i)$ as a pair of input and output, where $\mathbf{x_i} \in \mathbb{R}^{N\times 1}, y_i \in R$, then the multivariate Gaussian distribution model could be written as:

$$\mathbf{y} \sim N(\mathbf{0}, \mathbf{K}) \tag{6.4}$$

where $\mathbf{K}$ is the Gram matrix [82] that models the covariance between the inputs. In this chapter, the quadratic form [82] is used to determine $\mathbf{K}$:

$$K_{ij} = \theta_0 \exp(-\frac{\theta_1(\mathbf{x_i} - \mathbf{x_j})^2}{2}) + \theta_2 + \theta_3 \mathbf{x_i^T x_j} \tag{6.5}$$

where θ_i for $i = 0, \cdots, 3$ are the tunable variables.

Therefore, given the observations $(\mathbf{x}, \mathbf{y})$, if a new input $\mathbf{x}'$ is presented, the new distribution becomes:

$$\begin{bmatrix} \mathbf{y} \\ y' \end{bmatrix} \sim N(\mathbf{0}, \begin{bmatrix} \mathbf{K} & \mathbf{K_1} \\ \mathbf{K_1^T} & \mathbf{K_2} \end{bmatrix}). \tag{6.6}$$

Then the conditional probabilistic distribution of y' associated with new input $\mathbf{x}'$ is also given as a Gaussian distribution:

$$y'|\mathbf{y} \sim N(\mathbf{K_1 K^{-1} y}, \mathbf{K_2} - \mathbf{K_1 K^{-1} K_1^T}). \tag{6.7}$$

According to Eq. (6.7), the output of future input can be predicted based on known observations. Note that the computational complexity of GP is close to $O(n^3)$, where n is the number of

training data points, due to the inversion of the matrix, which makes GP less scalable. However, for a small amount of training data (e.g. a few hundred training data), the computational speed is acceptable.

The effect of the load (including the weight of the arm and the weight of the end-effector) on positioning errors of the end-effector can be formulated as a function of the joint angles and the weight of the end-effector, which can always be lumped into a 4×4 transformation matrix, such that $\mathbf{T}_{gp} = G(\mathbf{q}, m) \in SE(3)$.

After the POE calibration in the first step to identify the best set of model parameters to minimize the error between the measured and computed end-effector poses, there are still residual errors from non-geometric error sources that cannot be fully compensated by only changing the model parameters [83]. Let $\mathbf{p}'$ be the calibrated initial pose from the POE calibration, the remaining residual error is written as the difference of measured pose $\mathbf{T}'$ and the pose computed from calibrated POE model:

$$\epsilon = \log(\mathbf{T}'(\prod_{i=1}^{n}(e^{\hat{\mathbf{p}}_i{}'}e^{\mathbf{s}_i q_i})e^{\hat{\mathbf{p}}'_{\mathbf{n+1}}})^{-1})^V, \tag{6.8}$$

and the overall kinematic equation with GP compensation is:

$${}^0\mathbf{T}_n = \mathbf{T}_{gp}(\prod_{i=1}^{n}(e^{\hat{\mathbf{p}}_i{}'}e^{s_i q_i})e^{\hat{\mathbf{p}}'_{\mathbf{n+1}}}). \tag{6.9}$$

Therefore, in the second step, the minimization problem that minimizes the overall error with GP compensation is formulated as:

$$\min_{G} \| \log(\mathbf{T}'(\prod_{i=1}^{n}(e^{\hat{\mathbf{p}}_i{}'}e^{\mathbf{s}_i q_i})e^{\hat{\mathbf{p}}'_{\mathbf{n+1}}})^{-1}G(\mathbf{q}, m)^{-1})^V \|, \tag{6.10}$$

where $\mathbf{T}'$ is the measured pose of the end-effector; $\log \mathbf{T}^V$ converts the 4×4 homogeneous transformation matrix to 6×1 vector; $g(\mathbf{q}, m) \in \mathbb{R}^{6\times 1}$ is the GP model; and, $G(\mathbf{q}, m) \in SE(3)$ is the matrix exponential of $g(\mathbf{q}, m)$. As $G(\mathbf{q}, m)$ is an element of $SE(3)$, there exists at least one $\hat{g}(\mathbf{q}, m) \in se(3)$ such that $e^{\hat{g}(\mathbf{q},m)} = G(q, m)$, in which $\hat{g}(\mathbf{q}, m) \in se(3)$ is always associated with a unique vector $g(\mathbf{q}, m) \in \mathbb{R}^{6\times 1}$. The minimization problem formulated in Eq. (6.10) is then solved by computing the GP model.

Therefore, the residual pose error is modeled by GP such that given the error ϵ in Eq. (6.8) between the measured and computed poses, as well as the inputs $\mathbf{x} = [q_1, q_2, ...q_n, m]^T$, if a new input $\mathbf{x}'$ is presented, the error ϵ_i' ($i = 1, 2, \cdots, 6$) in each axis can be predicted by the following Gaussian distribution:

$$\epsilon_i' | \epsilon \sim N(\mathbf{K_1 K^{-1}} \epsilon, \mathbf{K_2} - \mathbf{K_1 K^{-1} K_1^T}) \quad i = 1, 2, \cdots, 6 \tag{6.11}$$

and the input to the GP is the joint angles and payload on the end-effector:

$$\mathbf{x} = [q_1, q_2, ...q_n, m]^T, \tag{6.12}$$

where q_i is the joint angle of i^{th} joint. Therefore, the minimization problem in Eq. (6.10) is then solved by finding the GP model that models the mapping from $\mathbf{x}$ to residual error ϵ.

Then the compensated $\mathbf{y}' \in \mathbb{R}^{6\times 1}$ can be written as:

$$\mathbf{y}' = \log(\mathbf{T}_{gp}(\prod_{i=1}^{n}(e^{\hat{\mathbf{p}}_i{}'} e^{s_i q_i}) e^{\hat{\mathbf{p}}'_{\mathbf{n+1}}}))^V \tag{6.13}$$

where $\mathbf{T}_{gp} = G(\mathbf{q}, m), \mathbf{T}_{gp} \in SE(3)$ is the compensation of error by GP model. As a machine learning model for compensation, $\mathbf{T}_{gp}$ could be put either before of after the POE model; in this chapter, it is put before the POE model in order to save computational cost.

6.3 Simulation and Experiments

In this section, the effectiveness of the proposed method is verified through simulation and experimental results. The proposed method is shown to significantly improve robot accuracy compared to the conventional base-tool calibration, POE model-based calibration and the POE+GP calibration without considering the payload at the end-effector in the GP model.

Different calibration methods are first compared using simulated data, followed by the same study using experimental data. The base-tool calibration method is adopted from previous work [78] in order to identify the relative pose of the robot base frame in the measurement frame and

the relative pose of the end-effector frame in the frame attached to the last link of the robot. The GP implementation is based on open source project scikit-learn [84] with quadratic covariance function. According to Eq. (6.10), (6.11) and (6.12), the training of GP model is conducted by using $\mathbf{x}$ in Eq. (6.12) as input of GP model and ϵ in Eq. (6.11) as output of GP model.

6.3.1 Simulation Results

In the simulation study conducted using Matlab and Python, the ABB IRB 4400 robot model is used with slightly perturbed kinematic parameters. The loading effect is simulated by adding different payloads on the end-effector, and the loading model used in this simulation is adopted from previous work [83]. Joint data are randomly generated within the workspace, a few loading weights (1kg, 2.5kg, 3.5kg, 5kg and 7.5kg) are included and randomly combined with the joint data.

A total of 300 training data points were randomly generated within the workspace, and a 10-fold cross validation was performed. The results shown in Figure. 6.1 indicate that the proposed method has the highest calibration accuracy compared with the other three calibration methods on a 10-fold cross validation of the training data.

Subsequently, all 300 training data points were used to obtain the calibrated model of the robot, and the updated model was tested using three different testing data sets, each consisting of 100 testing data. As shown in Table 6.1, POE+GP calibration considering the robot payload yields the highest calibration accuracy among the four calibration methods using the simulated testing data. The proposed method reduces the norm pose error by 87.5%, 71.3%, 63.1% on average compared to base-tool calibration, POE calibration and POE + GP calibration without considering the payload respectively.

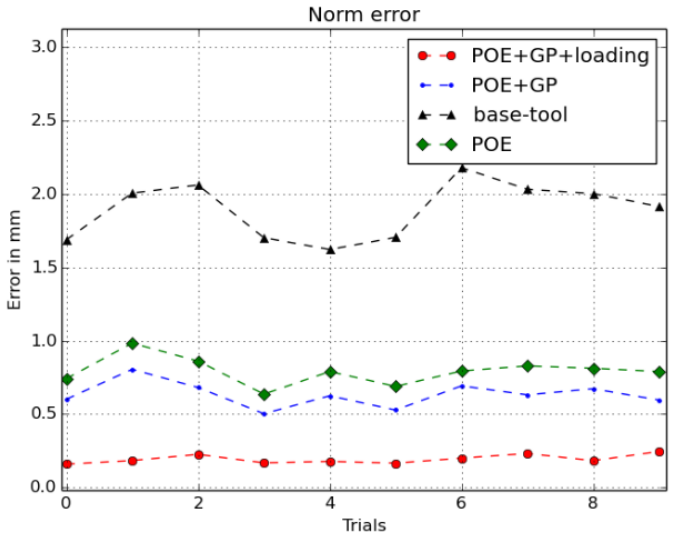


Figure 6.1: 10-fold cross validation of training data, comparing all methods with simulation data

Table 6.1: Simulation Results on Test Dataset

	Mean Error of Dataset (mm)		
	1	**2**	**3**
Base-tool Calibration	1.94	1.93	1.89
POE Calibration	0.90	0.81	0.81
POE + GP	0.71	0.63	0.62
POE + GP with Loading	0.23	0.24	0.25

6.3.2 Experiment Results

The experiment was carried out using an ABB IRB-4400 robot. The position and orientation of the robot end-effector was measured using Leica Absolute Tracker AT901-MR with a T-MAC 6 Degree-of-Freedom (DOF) sensor, which has a measurement accuracy of $30\mu m$. A few hundred measurements were collected randomly within the workspace; 5 mechanical tools with different weights $(1.46kg, 3.94kg, 4.77kg, 6.26kg, 9.72kg)$ were used in the experiment. The experimen-

tal setup is shown in Figure. 6.2.

The experiment was divided into two parts. In the first part, the experiment was performed with the T-MAC 6 DOF sensor on the end-effector without any additional loads, in order to demonstrate the effectiveness of using GP to compensate the residual error. In the second part, measurement data was collected by attaching different payloads at the end-effector. After randomly dividing the measurement data into training and testing data sets, a 10-fold cross validation was performed on the training data and finally the calibrated models, obtained with the four calibration methods using the training dataset, were tested using the testing data. The details of the experiments are shown in the later sections.

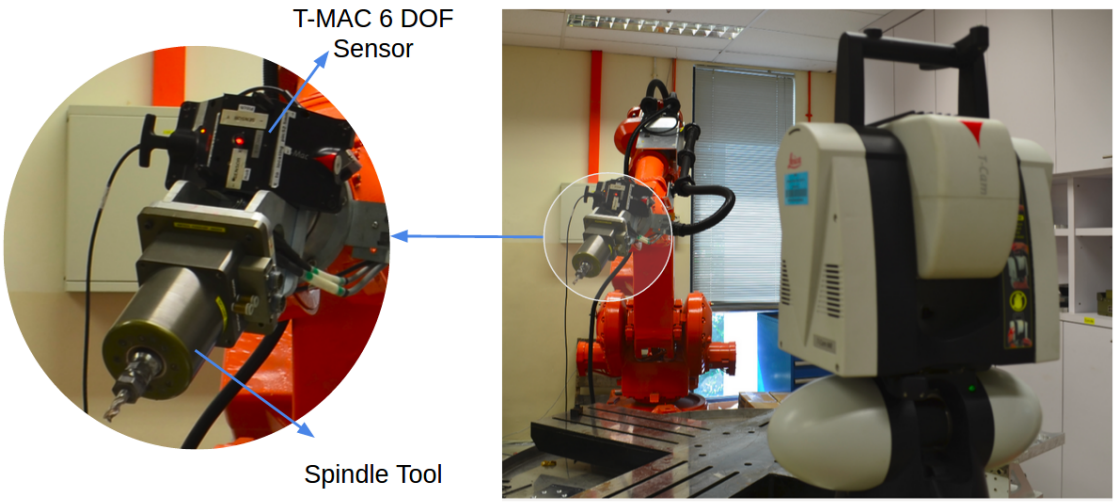


Figure 6.2: The experiment setup

In the first part of the experiment, POE calibration with GP compensation was adopted using the training data set obtained through experiments using the same payload at the end-effector. In this part, only the T-MAC 6 DOF sensor was mounted on the end-effector, which has a total weight of $1.49kg$. As the payload was the same, there was no difference between the POE+GP with and without considering the payload. Thus, the 10-fold cross validation was only performed among 3 calibration methods using the training data set. The calibration result is shown in Figure. 6.3, where the mean norm errors of base-tool calibration, POE calibration and POE +

GP calibration are $1.10mm$, $0.80mm$ and $0.40mm$, respectively.

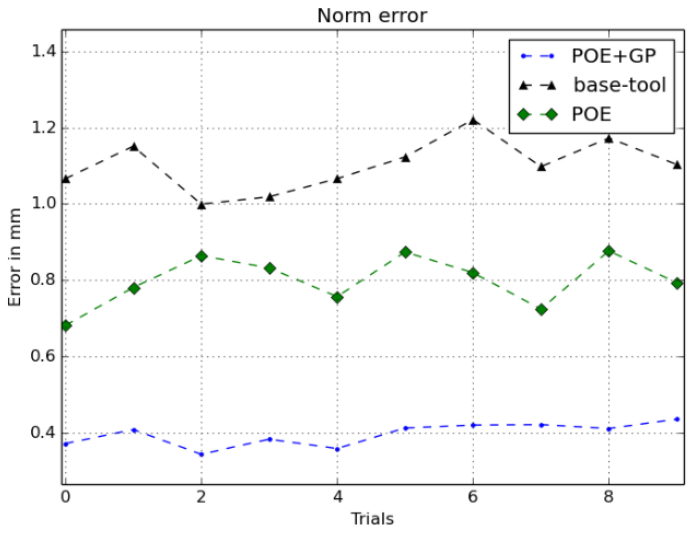


Figure 6.3: 10-fold cross validation of training data set, comparing three methods with experimental data with the same loading

In the second part of the experiment, the experiment data set of 5 mechanical tools with different weights $(1.46kg, 3.94kg, 4.77kg, 6.26kg, 9.72kg)$ was used. After collecting the data, the data is randomly divided into a training data set and three testing data sets. A 10-fold cross validation on the 300 training measurements was performed, and the calibrated models were obtained using the 4 calibration methods and the training data. The results illustrated in Figure. 6.4 demonstrate the effectiveness of the proposed method.

To further validate the proposed method, the calibrated models were tested using 3 different testing data sets, each consisting of 100 measurements, which have not been used to train the models. As shown in Table 5.2, POE+GP calibration considering payload yields the highest accuracy among the four calibration methods. The proposed method reduces the norm pose error by 65.5%, 50.2%, 48.2% on average compared to base-tool calibration, POE calibration and POE + GP calibration without considering payload respectively.

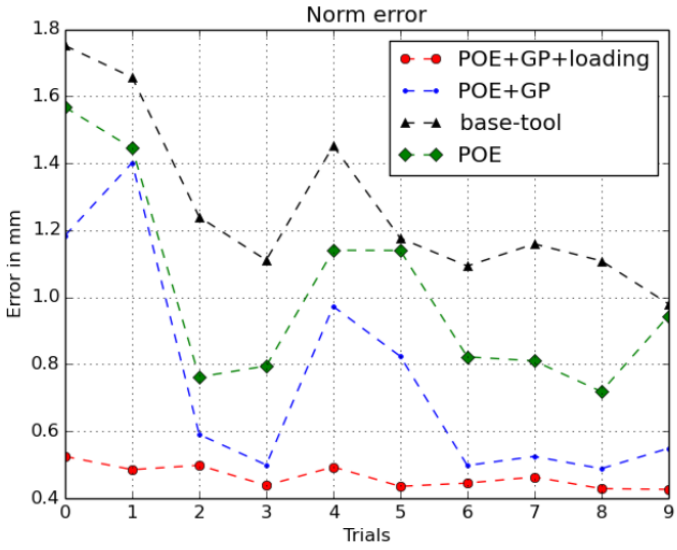


Figure 6.4: 10-fold cross validation of training data set, comparing all methods with experimental data

Table 6.2: Experiment Results on Test Dataset

	Mean Error of Dataset (mm)		
	1	**2**	**3**
Base-tool Calibration	1.17	1.34	1.76
POE Calibration	0.87	0.82	1.30
POE + GP	0.83	0.78	1.28
POE + GP with Loading	0.43	0.46	0.57

6.4 Summary

The results presented in this section show that the proposed method of POE+GP with consideration of loading effects yields the least error compared to existing methods. This is possibly due to a number of factors. Firstly, GP regression is suitable for small amount of training data set because of its robustness and smoothness, and it is also considered as a non-parametric model which should work well in many situations with little effort on tuning GP model parameters. One well-known limitation of GP is its poor scalability, but in this problem it is not a major issue as

the size of the training data set is relatively small (e.g. a few hundred training data). In addition, the experimental results confirmed that the proposed new scheme considering the payload at the end-effector improves the modeling accuracy of the GP.

Chapter 7

Coverage Motion Planning for Industrial Inspection

7.1 Introduction

In this chapter, we propose a novel planning method for the industrial inspection applications [85][4], where a sensor-equipped industrial manipulator is used to perform a 3D shape inspection task. The proposed method first formulates the SCP and TSP individually for the given target object, then combines the two problems as a single sequencing optimization problem, and solves it simultaneously using Random-Key Genetic Algorithm (RKGA). The proposed method also generates the resultant trajectory as a ROS-compatible format that can be directly used for practical applications. Compared to traditionally formulated and solved separately [47], the proposed method achieves better planning results by requiring less inspection time. The main contributions of the planning method proposed in this chapter are:

- a novel sequencing optimization problem formulation and a RKGA-based approach for the combined SCP and TSP problems, which leads to better results;
- an encoding/decoding strategy of RKGA that handles the coverage and TSP constraints in the fitness evaluation of chromosome, which simplifies the optimization process;
- a combined SCP and TSP formulation for robotic inspection application with validation and benchmark of real-world robotic data; and
- a overall framework to generate efficient motion plan for the industrial shape inspection applications with robotic manipulator and 3D scanner.

7.2 Problem Formulation

For the robotic inspection task discussed in this paper, a 3D scanner is mounted on a 6 Degree-Of-Freedom (DOF) industrial manipulator and the target object is placed in front of the robot. The robot is required to begin the inspection process from its home position, travel to the planned viewpoints and take the required measurements with its on-board sensor; after all the required data is captured, the robot should return to its initial home position to prepare for the next in-

spection. The overall planning process is to find the set of required viewpoints together with the corresponding robot poses, the visiting sequence for the viewpoints, as well as collision-free robot paths between them, such that the overall cycle time required for the inspection task is minimized.

7.2.1 SCTSP Formulation

In this paper, we formulate the overall industrial inspection process in such a way that the SCP and TSP are solved simultaneously, which we term as the Set-Covering-Traveling-Salesman-Problem (SCTSP).

The problem is formulated in discrete space. Given the viewpoint set $\mathbb{V}$ with n viewpoints, surface patch set $\mathbb{S}$ with m patches of the target object, and a undirected Graph $G(\mathbb{V}, E)$, where e_{ij} or $e(v_i, v_j)$ is the edge between v_i and v_j, and it represents the traveling cost between viewpoints v_i and v_j; and a $m \times n$ binary matrix $\mathbf{A}$ that represents the visibility information between viewpoint set $\mathbb{V}$ and surface patch set $\mathbb{S}$.

The problem is formulated in discrete space by sampling. The notations to form the optimization problem are listed as follows. The candidate viewpoint set $\mathbb{V}$ has n viewpoints. The surface patch set $\mathbb{S}$ has m patches of the target object. A undirected Graph $\mathbf{G}(\mathbb{V}, \mathbb{E})$ encodes the traveling costs information among the viewpoints. e_{ij} or $e(v_i, v_j) \in \mathbb{E}$ is the edge between v_i and v_j, and it represents the traveling cost between viewpoints v_i and v_j. A $n \times m$ binary visibility matrix $\mathbf{A}$ represents the binary visibility information between candidate viewpoint set $\mathbb{V}$ and surface patch set $\mathbb{S}$.

Similar to the Traveling View Planning Problem formulated in previous work [48] but with different constraints formulation, the SCTSP problem is to find a subset $\mathbb{V}' \in \mathbb{V}$, such that the coverage constraint of the target surface is satisfied; the traveling cost of a Hamiltonian tour in $\mathbb{V}'$ and the inspection cost are minimized. The formulation is shown in Eqn. (7.1)-(7.7) below:

$$\min_{\mathbf{b},\mathbf{c}} \quad \underbrace{\sum_{i=1}^{n} \omega_i b_i}_{\text{inspection cost}} + \underbrace{\sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} c_{ij} e(v_i, v_j)}_{\text{traveling cost}} \tag{7.1}$$

$$\text{subject to:} \quad \sum_{i=1}^{n} b_i A_{ij} \geq 1 \quad \forall j \tag{7.2}$$

$$\sum_{v_i \in \mathbb{V}', v_i \neq v_j} c_{ij} = 1, \{j | \forall v_j \in \mathbb{V}'\} \tag{7.3}$$

$$\sum_{v_j \in \mathbb{V}', v_i \neq v_j} c_{ij} = 1, \{i | \forall v_i \in \mathbb{V}'\} \tag{7.4}$$

$$\sum_{v_i, v_j \in L} c_{ij} \leq |L| - 1, L \subset \mathbb{V}', |\mathbb{V}'| - 2 \geq |L| \geq 2 \tag{7.5}$$

$$\text{where:} \quad b_i, c_{ij} \in \{0, 1\} \tag{7.6}$$

$$\mathbb{V}' = \{v_i | b_i = 1, i = 1, 2, \cdots, n\}, \tag{7.7}$$

where w_i is the inspection cost at viewpoint v_i; (7.2) is the coverage constraints; (7.3), (7.4), (7.5) are the the constraints of DFJ formulation for TSP within $\mathbb{V}'$, adapted from [86][87]. The objective is to minimize the summation of the inspection cost and the traveling cost at the same time. In this paper, the cost is measured as the cycle time of the overall inspection task.

7.2.2 Formulation for Sequencing SCTSP

It is difficult to handle the constraints (7.2)-(7.5) directly with Integer Linear Programming method. To simplify the optimization process, we proposed a novel sequencing SCTSP formulation, and applied RKGA to solve it. The proposed sequencing SCTSP formulation is: finding a ordered viewpoint set $\mathbf{X}$, such that every element $x_i \in \mathbf{X}$ is from $\mathbb{V}$; the coverage constraint is

satisfied; the sum of the inspection cost and the traveling cost is minimized.

$$\min_{\mathbf{X}} \quad \underbrace{\sum_{x_i \in \mathbf{X}} \omega_i x_i}_{\text{inspection cost}} + \underbrace{\sum_{i=1}^{n_X - 1} e(x_i, x_{i+1}) + e(x_{n_X}, x_1)}_{\text{traveling cost}} \tag{7.8}$$

$$\text{subject to:} \quad \sum_{i=1}^{n_X} A_{x_i,j} \geq 1 \quad \forall j, \tag{7.9}$$

where n_X is the size of $\mathbf{X}$, $A_{x_i,j}$ is the j^{th} element in the row corresponding to viewpoint x_i. The visiting sequence is then directly determined by the ordered set $\mathbf{X}$, so the TSP constraints are removed from the formulation. For the coverage constraint, every surface patch is checked about the visibility condition with the selected viewpoints, as shown in (7.9).

7.3 Proposed Method

In this chapter, we propose a novel model-based, offline motion planning method for the robotic inspection applications. The proposed method first generates necessary data for the view planning and path planning problems individually, and then moves on to solve the overall SCTSP problem in a single sequencing optimization process with RKGA. The overall process is summarized in the following three steps:

1. First, randomized sampling is used to generate a set of candidate viewpoints $\mathbb{V}$ around the target object; the corresponding robot poses for each viewpoint are also generated at this stage.
2. In the second step, visibility and traveling cost are evaluated for the later optimization process:
    - A visibility matrix $\mathbf{A}$, which encodes visibility information between each surface patch on the target object and candidate viewpoint.
    - The undirected graph $\mathbf{G}(\mathbb{V}, \mathbb{E})$ with the edge set $\mathbb{E}$ that encodes the traveling costs between candidate viewpoints in the configuration space of the robot.

3. Finally, a RKGA algorithm is used to solve the formulated sequencing SCTSP problem.

As shown in Figure 7.1, the proposed method takes the robot model, the sensor parameters and the target object model as input, automatically generate a efficient motion plan that satisfies the inspection requirements. Further details of the proposed method are presented over the remaining part of this section.

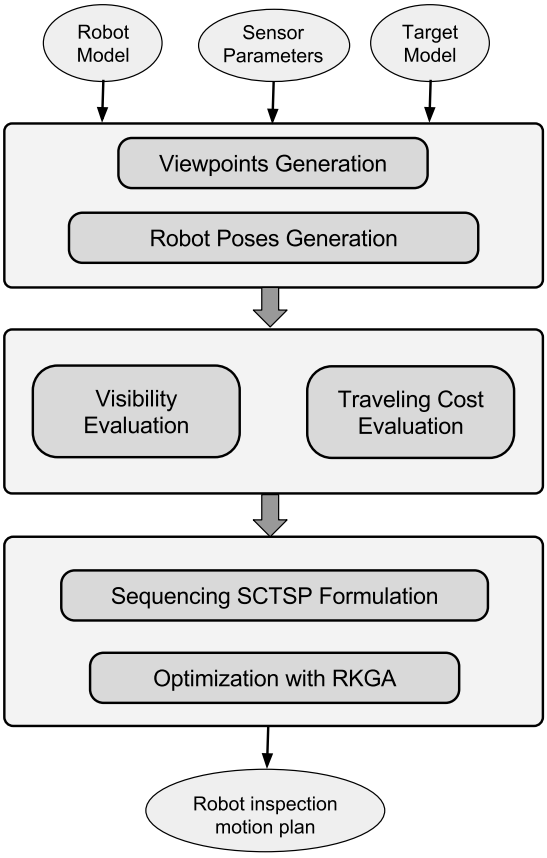


Figure 7.1: The brief flowchart of the proposed method

7.3.1 Viewpoints and Robot Poses Generation

The first step of the proposed method is to generate a set of candidate viewpoints $\mathbb{V}$ about the target object. For each generated viewpoint, a corresponding robot pose is also generated via inverse kinematics.

Sampling Viewpoints

In this chapter, a method of randomized sampling is used to generate candidate viewpoint positions. The sampling process of the viewpoints is performed in Euclidean space around the target object, within the maximum viewing range of the sensor. Redundant viewpoints are generated in this step.

In addition to the position, the viewing direction is chosen by potential field method [9]. This method ensures each viewpoint is oriented in a direction that captures a weighted average of nearby surface patches on the target object. The viewing direction $\mathbf{v}$ of a given viewpoint located at $\mathbf{p_{vp}}$ is:

$$\mathbf{v} = \frac{\sum_i^N \frac{K\mathbf{d_i}}{\|\mathbf{d_i}\|^3}}{\| \sum_i^N \frac{K\mathbf{d_i}}{\|\mathbf{d_i}\|^3} \|}, \tag{7.10}$$

$$\text{where} \quad \mathbf{d_i} = \mathbf{p_{vp}} - \mathbf{p_{patch_i}}$$

$$\text{for all} \quad \{\mathbf{p_{patch_i}} | (\mathbf{p_{vp}} - \mathbf{p_{patch_i}})^T (\mathbf{p_{vp}} - \mathbf{p_{patch_i}}) < d_{max}\},$$

where K is a constant value; $\mathbf{p_{patch_i}}$ is the position of i^{th} surface patch; d_{max} defines the maximum inclusion distance, only surface patches located within this distance to the viewpoint will be included in the calculation of the mean viewing direction.

Generating Robot Poses for each Viewpoint

A valid robot pose must be defined for each member in the candidate viewpoint set. To do this, each candidate viewpoint is input into the IKFast [88] inverse kinematics algorithm to generate a corresponding robot pose. The Flexible Collision Library (FCL) [89] is then used to test the robot pose for collision with the surrounding environment. If either of these tasks fails, we then rotate the viewpoint about its optical axis until a valid robot pose is found. We are able to do this, as rotating a given viewpoint about its optical axis does not significantly alter the sensors ability to capture the geometric data of a given scene. The subsequent rotated viewpoint, and if it is a valid robot pose, then replaces its former counterparts in the candidate viewpoint set. If no valid

pose for a given viewpoint is possible, the viewpoint is removed from the candidate viewpoint set.

7.3.2 Evaluation of Viewpoints and Robot Poses

In the second step, two properties from each candidate viewpoint are evaluated. First, a measure of visibility between each surface patch on the target object and each candidate viewpoint is calculated and stored in a two-dimensional visibility matrix $\mathbf{A}$. Then, the edges of the Graph $\mathbf{G}(\mathbb{V}, \mathbb{E})$ are found by evaluating the associated traveling costs for the robot to travel between the candidate viewpoints.

Evaluation of Visibility

With the 6-DOF position and orientation of the viewpoints, the visibility of the surface patches can be then evaluated. A triangular surface patch is considered visible from a given viewpoint if all the following conditions are met for all of its vertices:

- The vertices of the surface patch must be in the Field-Of-View (FOV) of viewpoint.
- The vertices of the surface patch must be in the Field-Of-Depth (FOD) of viewpoint.
- The viewing angle must be within the specified range.
- There is no occlusion in between the viewpoint and surface patch.

The visibility criterion is adapted from our previous work [9], and is similar to the methods documented in [47] and [33]. The visibility evaluation process generates a $n \times m$ binary visibility matrix $\mathbf{A}$, where n is the number of candidate viewpoints, and m is the number of surface patches on the target model. The information encoded in this matrix indicates whether a given surface patch is visible to a particular viewpoint. For example, with the value of A_{ij}, 1 indicates the surface patch j is visible to the viewpoint i, whereas 0 means it is not visible.

Evaluation of Traveling Costs

The required collision-free robot path between the robot poses at two given viewpoints is computed using the MoveIt [90] framework from Robot Operating System (ROS) [91]. The framework uses the Open Motion Planning Library (OMPL) [92] for path planning and FCL [89] for collision detection. The local planner used in this paper is the Rapidly exploring Random Tree connect (RRT-connect) algorithm [20], similar to the adaptation in the previous work [44]. RRT-connect is a variation of RRT, and operates by generating poses randomly in the robot's configuration space and initiates exploration from both starting and goal poses. The method is considered probabilistically complete [21], and has been used in this work as it is able to generate high quality robot paths between poses in a short amount of time. Note that in the proposed framework, other local planner such as RRT* can also be used to generate the pose-to-pose paths.

The cubic spline is used to parametrize the generated robot paths with speed and acceleration data of the robot's actuators. This allows the robot paths to be converted into the robot trajectories. The subsequent traveling times can then be computed. The traveling costs (time) between the robot poses at the viewpoints are calculated and stored in the corresponding edges of $\mathbf{G}(\mathbb{V}, \mathbb{E})$.

7.3.3 Random-key Genetic Algorithm for SCTSP Problem

With the visibility matrix and $\mathbf{G}(\mathbb{V}, \mathbb{E})$ generated in the previous sections, the SCTSP is now formulated as a variation of sequencing problem. A Random-Key Genetic Algorithm (RKGA) is then applied to solve it in a single optimization process.

The Random-key Genetic Algorithm (RKGA)

The RKGA method is a popular technique used in many combinatorial optimization problems, such as the TSP problem [93] and other sequencing problems [69]. Similar to standard GA method, in RKGA the population consists of many chromosomes; each chromosome featuring a

number of genes. Different from the traditional GAs, the RKGA makes use of random real numbers as the key stored in each gene of the chromosome. A decoder then deciphers information in the genes of the chromosome, before evaluating the fitness/cost score.

In this chapter, we use real number range from 0 to 1 as the random key to encode the information. The decoding strategy first sorts the chromosome by the random key value stored in the genes; then instead of using all sorted genes for fitness evaluation in the chromosome in literature [69], the proposed method continually adds the sorted genes to solution set and evaluate the fitness until the coverage constraint is satisfied. This approach bypasses the coverage constraint by evaluating coverage criteria in the fitness function of RKGA, as shown in Algo. 5. For example, a chromosome with the genes $(0.31, 0.25, 0.84, 0.64, 0.93)$ is first decoded as a sequence of $2 \rightarrow 1 \rightarrow 4 \rightarrow 3 \rightarrow 5$, then during the fitness evaluation, the viewpoint with a index number 2 will firstly be added to the solution set, the cost will be evaluated accordingly. Meanwhile the fitness function will evaluate the coverage constraint as well; if the coverage constraint is not satisfied, the next viewpoint with index number 1 will be added to the solution set, this process repeatedly add n_x viewpoints to the solution set until the coverage constraint is satisfied.

The encoding/decoding strategy used in this paper has the advantages of not dealing with the constraints directly, which simplifies the optimization process. For example, if we use the Eqn. (1) - (7.7) as a Integer Linear Programming (ILP) problem, the coverage constraint (7.2) and TSP subtour elimination constraints (7.3)-(7.5) must be handled by the optimization algorithm. In addition, note that though written in a compact form, the coverage constraint (7.2) actually have lots of linear equations with integer components.

Solving the Sequencing SCTSP using RKGA

With the sequencing formulation of the SCTSP problem at Section 7.2, the visibility matrix and $\mathbf{G}(\mathbb{V}, \mathbb{E})$, a optimization formulation with adaptation of RKGA is shown in Eqn. 7.11.

In this paper, we assume constant inspection time ω_0 for each viewpoint; and the set covering constraint (7.13) is relaxed such that a certain coverage ratio δ is desired instead of full coverage. Additionally, as mentioned in Section 7.2, since the proposed method is designed for production line, the robot should start at its home pose, and return to the home pose at after finishing the inspection task. The formulation is shown below:

$$\min_{\mathbf{x}', n_x} \quad \underbrace{\omega_0 n_x}_{\text{inspection cost}} + \underbrace{\sum_{i=1}^{n_x - 1} e(x'_i, x'_{i+1}) + e(x'_h, x'_1) + e(x'_{n_x}, x'_h)}_{\text{traveling cost}} \tag{7.11}$$

$$\text{where} \quad x'_i \in \mathbb{V} \tag{7.12}$$

$$\text{subject to} \quad \sum_{i=1}^{n_x} ((\sum_{j \in \mathbf{x}'} \mathbf{A_j}) \geq 1) \geq \delta * m, \tag{7.13}$$

where $\mathbf{x}'$ is the ordered set of the selected viewpoints; $e(x'_i, x'_{i+1}) \in \mathbb{E}$ is the robot traveling cost between its poses at viewpoints x'_i and x'_{i+1}; x'_h represents the robot home pose, so $e(x'_h, x'_1)$ means the traveling cost between the home pose and the pose at first viewpoint in the solution, $e(x'_{n_x}, x'_h)$ means the traveling cost between the home pose and the pose at last viewpoint in the solution; (7.13) is the coverage constraint; $\mathbf{A}$ is the $n \times m$ visibility matrix; $\mathbf{A_j}$ is the j^{th} row of matrix $\mathbf{A}$; δ is the required coverage ratio. In this formulation, we slightly adjust the SCP so that a certain coverage ratio instead of full coverage is required, which is similar to the formulation reported in literature[10]. In RKGA, the chromosome $\mathbf{x}$ is a list with random key x_i stored in its genes, the ordered viewpoints set $\mathbf{x}'$ is obtained from the chromosome $\mathbf{x}$; the coverage constraint (7.13) is evaluated in the fitness function of RKGA.

In the proposed approach, the number of selected viewpoints n_x is not directly encoded in the chromosome. Instead, n_x is determined by the algorithm's fitness function, our approach only chooses the first n_x viewpoints in the ordered set $\mathbf{x}'$ that satisfy the required coverage constraints. By doing this, all offspring generated will be able to generate feasible solutions, and we can then bypass the coverage constraint introduced by the SCP, as well as the TSP subtour elimination constraints. The details of cost function that evaluates the cost of a given chromosome are shown

Algorithm 5 Evaluating the Cost of a Given Chromosome

Input: $\mathbf{x}$: the chromosome with the random keys stored in its genes; $\mathbf{G}(\mathbb{V}, \mathbb{E})$: the graph; $\mathbf{A}$: the visibility matrix; δ: the desired coverage ratio; ω_0: inspection cost at viewpoint.

Output: y: the fitness of the input chromosome; n_x: the number of first few viewpoints needed for the required coverage;

$y = \omega_0$
$n_x = 0$
$\mathbf{x}' \leftarrow$ decodeBySort($\mathbf{x}$)
$y = y + e(x'_h, x'_1)$
for each $x'_i \in \mathbf{x}'$ and $i > 1$ **do**
    $y = y + \omega_0 + e(x'_i, x'_{i+1})$
    $n_x = n_x + 1$
    $\delta' \leftarrow$ findCoverageRatio($\mathbf{A}, x'_i$)
    if $\delta' > \delta$ **then**
        break
    end if
end for
$y = y + e(x'_{n_x}, x'_h)$
return y, n_x

in Algo. 5, where "decodeBySort($\mathbf{x}$)" returns a sorted list by sorting the value of $\mathbf{x}$.

7.4 Simulation, Experiment and Discussion

In this chapter, the validity and effectiveness of the proposed method is verified through both simulation and experiment. The performance of the proposed method is compared to the more traditional approaches where the SCP and TSP are solved individually. Two other approaches are used in these comparative tests: The first approach solves both SCP and TSP using two separate GAs (labelled as GA-SCP-GA-TSP); the other method solves the SCP using a greedy method [9] and then solves the TSP using a RKGA solver (labelled as Greedy-SCP-GA-TSP). In this paper, we assume that the inspection time at different viewpoints carries the same cost, making the total inspection cost directly proportional to the number of required viewpoints, the assumption is reasonable because the inspection time is only related to the 3D scanner, the 3D reconstruction of the polygonal model is done in post-processing. Traveling costs are lifted from the edge set $\mathbb{E}$ of $\mathbf{G}(\mathbb{V}, \mathbb{E})$ generated in the earlier process.

Table 7.1: Sensor parameters

FOV	$30°, 21°$
Camera Pixels	536×371
Viewing Angle	$75°$
FOD (meters)	$(0.40, 1.00)$
Coverage ratio	$\delta = 99.5\%$

In both simulation and experiment, the robot used is an ABB-IRB-4600 6-DOF industrial manipulator. An Artec Eva 3D scanner is mounted on the end-effector of the robot. According to its data sheet, the Artec scanner has a FOD of $0.40m$ to $1.00m$, as well as a FOV of $30°$ and $21°$ in the horizontal and vertical directions, respectively. In this chapter, the sensor parameters

used for planning are listed in Table 7.1.

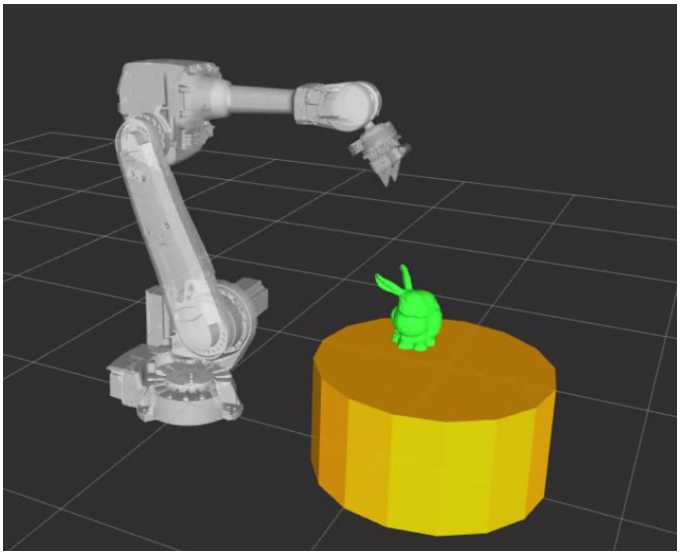
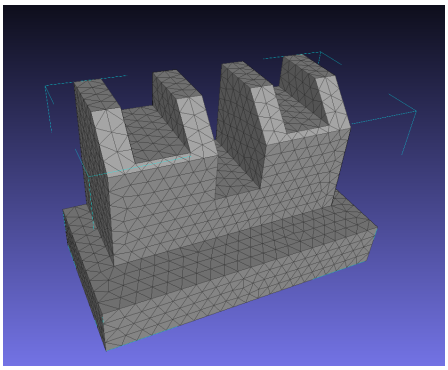
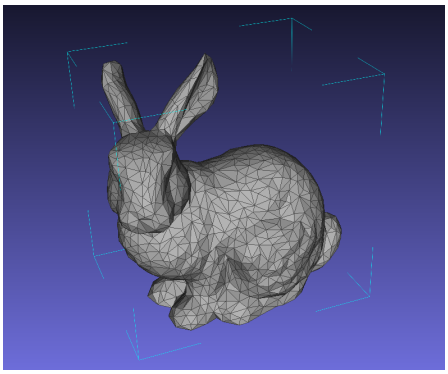


Figure 7.2: The simulation setup: a target object (in green) is placed on a machining table (in orange) for inspection; the ABB IRB-4600 industrial manipulator (in grey) has a 3D scanner mounted on the robot end-effector. The pose displayed in the figure is used as home pose for this chapter



(a) Target object 1: the bounding box size is $0.24 \times 0.47m \times 0.30m$



(b) Target object 2: the bounding box size is $0.20m \times 0.30m \times 0.30m$

Figure 7.3: The target objects

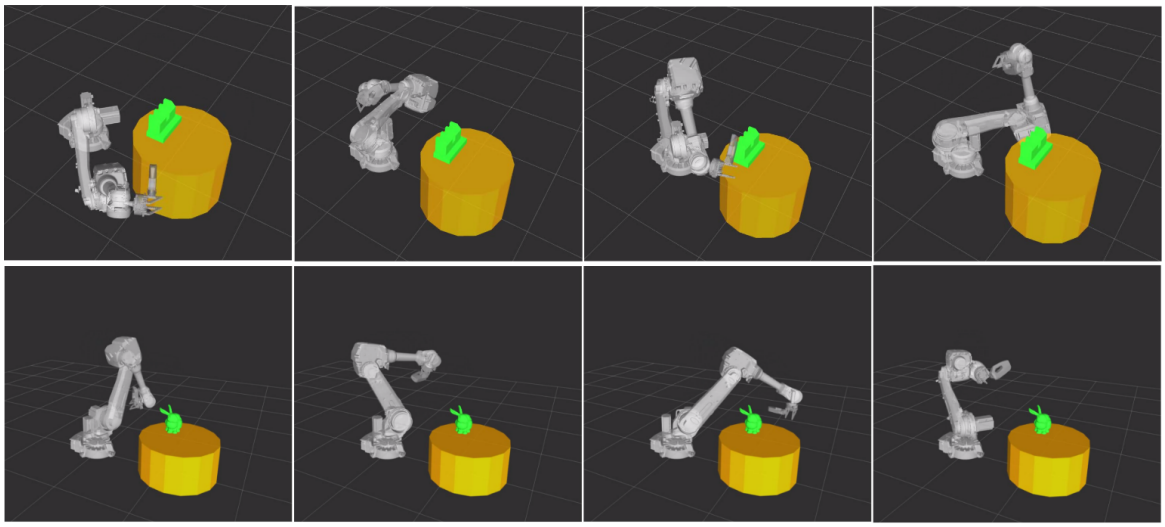


Figure 7.4: Example of robot poses to position the 3D scanner to desired viewpoints for inspecting target object 1 and 2

7.4.1 Setup

The simulation-based tests performed in this work were carried out using ROS [91]. ROS provides an built-in visualization environment (Rviz), planning interface (MoveIt), motion planning library (OMPL [92]) and collision detection (FCL [89]). Inverse kinematic queries were solved using IKFast [88]. Note that though multiple inverse kinematics solutions may exist for certain poses, we do not address this issue in the chapter. The multiple solution problem of inverse kinematics can be addressed by a sampling strategy.

In the simulation environment, a machining table is positioned in front of the industrial manipulator, as shown in Figure 7.2. The table is modelled as cylinder to speed up collision detection processing, while the bounding box of the target object is used for collision check between the robotic arm and the target object. Two simulation tests, each featuring different target objects, shown in Figure 7.3, were performed. The respective target object is placed on the circular table 0.7m in height for the first test and 0.5m for the second test; the target object is positioned 0.9m away from the robot base location, as shown in Figure 7.2. A constant inspection time of 1 second at each viewpoint was used.

7.4.2 Simulation Results

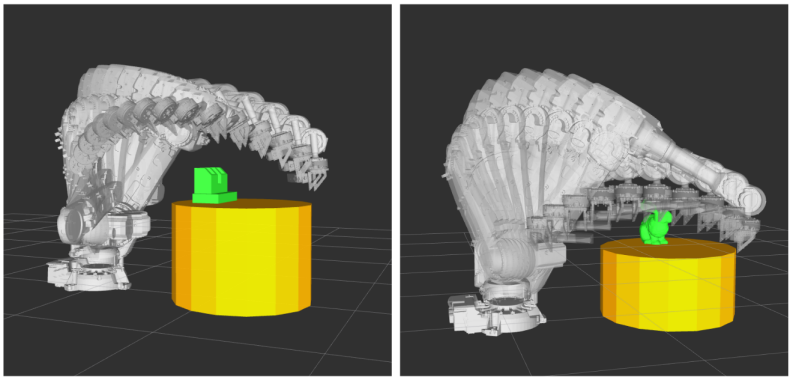


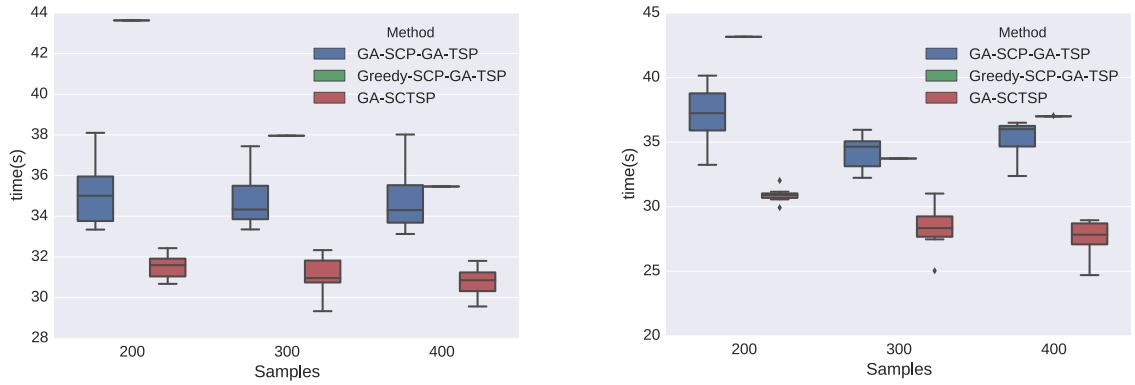
Figure 7.5: The example trajectory for inspecting both target objects

The results for the two simulations are presented in Table 7.2, 7.3 and Figure 7.6, where T_{avg} is the average cycling time based on ten trials, T_{std} is the standard variation of the cycling time, and N_{vp} is the average number of viewpoints required for the inspection. In Figure 7.6a and 7.6b, different numbers of candidate viewpoints are used in the comparison; the same set of sampled candidate viewpoints are used for the three different methods. Note that for Greedy-SCP-GA-TSP method, because the greedy method finds deterministic results of viewpoints, and the scale of the TSP problem within the resultant viewpoints is small, the results for Greedy-SCP-GA-TSP have very little variations.

The results of the first simulation indicate that the proposed method (labeled as GA-SCTSP) is able to generate a better solution for the given inspection task, as shown in Table 7.2. For the overall cycle time, respective reductions of 10.8% and 19.6% in time were recorded compared to the previous methods. Example poses and robotic trajectories are shown in Figure 7.4 and Figure 7.5. Figure 7.6a also shows the results of different methods and different number of viewpoints generated.

In the second test, the advantage of the proposed method was observed to improve further. In terms of number of required viewpoints, the GA-SCP-GA-TSP method performs best, requiring fewer viewpoints than the GA-SCTSP and Greedy-SCP-GA-TSP methods. However in terms of

overall time costs, the proposed method required 18.6% and 23.3% less cycling time compared to the previous methods on average, as shown in Table 7.3. A 28.4% overall time reduction compared to Greedy-SCP-GA-TSP methods is recorded when using 200 sampled viewpoints. Example poses and robotic trajectories are shown in Figure 7.4 and 7.5.



(a) Targat Object 1 **(b)** Targat Object 2

Figure 7.6: Total cycling time required of different methods for inspecting target objects (based on ten trials)

Table 7.2: Results for inspecting target object 1

	200 samples			**300 samples**			**400 samples**		
	T_{avg}	T_{std}	N_{vp}	T_{avg}	T_{std}	N_{vp}	T_{avg}	T_{std}	N_{vp}
GA-SCP-GA-TSP	35.1	1.50	9.2	34.9	1.40	9.3	34.8	1.61	9.1
Greedy-SCP-GA-TSP	43.6	0.00	12.0	38.0	0.00	11.0	35.5	0.00	10.0
GA-SCTSP	31.3	0.62	9.2	31.1	0.91	9.3	30.8	0.71	8.9

Table 7.3: Results for inspecting target object 2

	200 samples			**300 samples**			**400 samples**		
	T_{avg}	T_{std}	N_{vp}	T_{avg}	T_{std}	N_{vp}	T_{avg}	T_{std}	N_{vp}
GA-SCP-GA-TSP	37.1	2.17	8.3	34.2	1.25	8.5	35.2	1.51	7.4
Greedy-SCP-GA-TSP	43.2	0.00	11.0	33.7	0.00	9.0	37.0	0.02	8.0
GA-SCTSP	30.9	0.53	9.2	28.4	1.57	9.1	27.5	1.46	8.7

7.4.3 Experiment Results

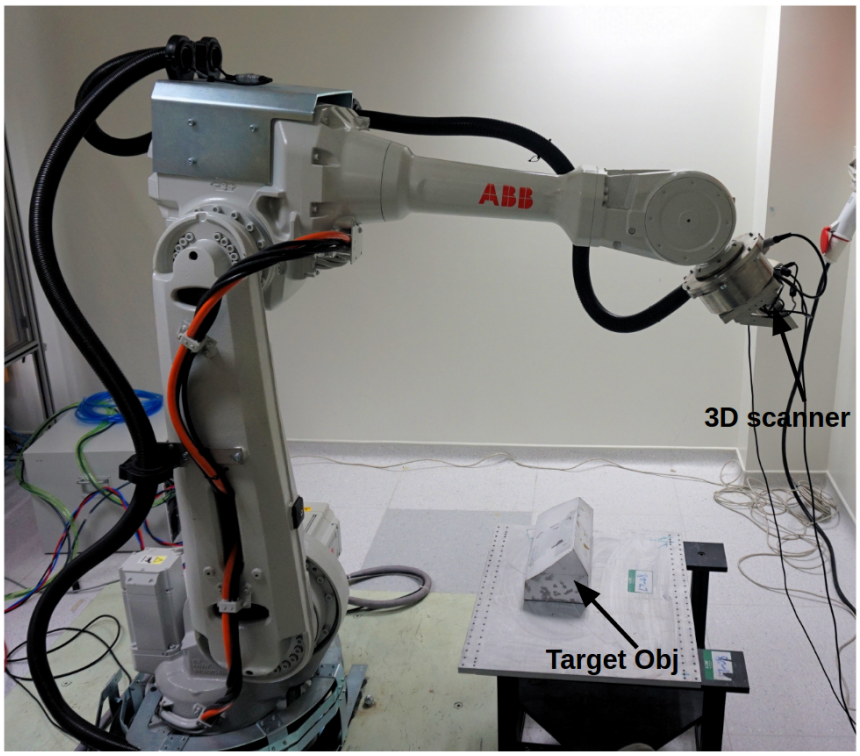


Figure 7.7: Experimental setup

To perform the planned robot motions with the real world robotic hardware, the ROS-industrial package [94] is used, which allows the trajectories generated in the simulation environment to be sent to the real-world robot controller via Ethernet directly. These trajectory commands control the robot so it moves to the planned viewpoints, and the scanner is configured via I/O signals to perform scanning operations at the required moments. Example planned poses and their corresponding actual poses are shown in Figure 7.8. Once all required scans were captured, the Artec SDK library is used, along with robot pose feedback data, to stitch together an accurate composite CAD model of the target object, as shown in Figure 7.10. During the experiment, the inspection plan is computed using 400 samples of viewpoints. With the proposed method, 19.2 seconds are required for the inspection plan, while GA-SCP-GA-TSP requires 23.8 seconds and Greedy-SCP-GA-TSP requires 25.7 seconds for the task.

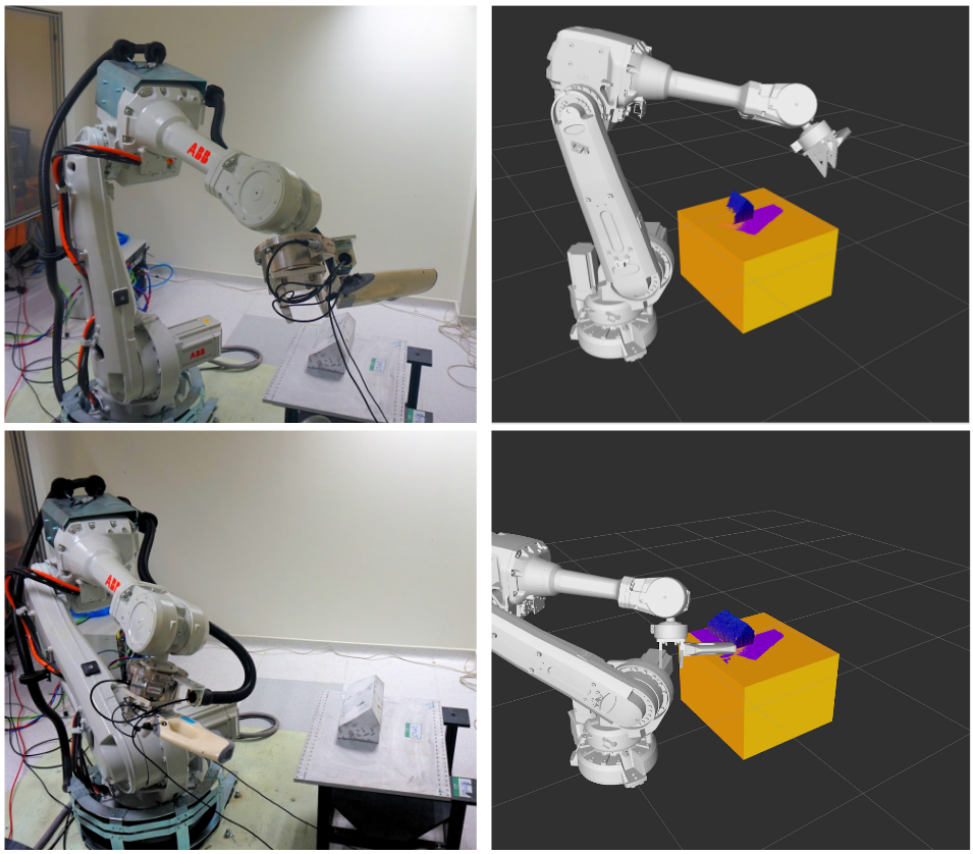
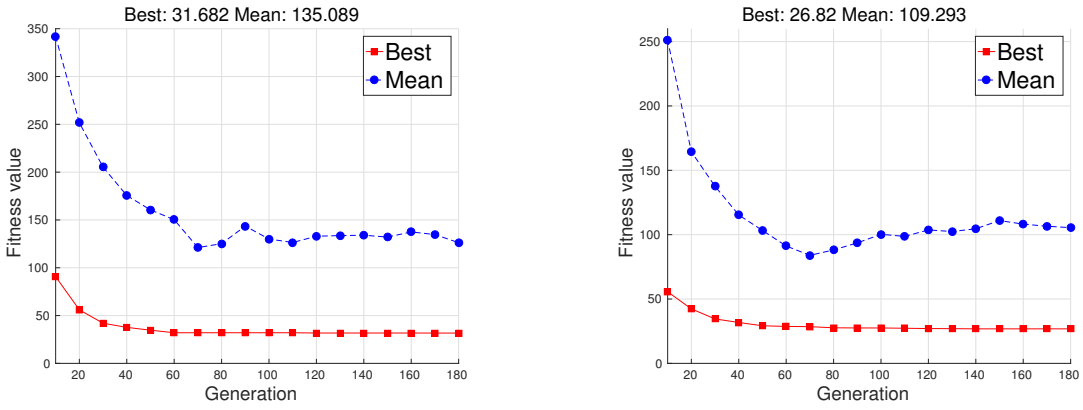


Figure 7.8: Visualization of example inspection poses during the experiment and partial scan in Rviz and example actual robot poses for inspection during the experiment.

7.4.4 Covergence and Computational Cost

As shown in Figure 7.9, it is observed that the results of RKGA start to converge before 100 generations. 400 samples are used in the figure. The score of the best known chromosome is shown in red, while the average score is shown in blue.

The runtime of 400 samples with the proposed method is about 3 hours on a Intel i7-5500U laptop CPU with 12G RAM. The computation of the pose-to-pose trajectories and traveling costs takes most of the computational time.



(a) Targat Object 1

(b) Targat Object 2

Figure 7.9: Convergence of RKGA

7.4.5 Discussion

Both the simulation and experiment show that integrating the SCP and TSP components of the generalized planning process together can provide a better solution than solving the SCP and TSP separately. This can be attributed to the fact that when the problems are solved separately, the SCP solver is not aware of the traveling cost, which results in sub-optimal solutions. By integrating these two problems during the optimization process, the solver will continue to search for a minimized overall cost even after the SCP solution has already been solved.

This method could also be extend to inspection or reconstruction tasks with UAV [10][9]. In these applications, traveling times are proportionally much larger than viewing time, so we may expect further improved results with the method proposed in this chapter. Future work is proposed to explore this concept.

7.5 Summary

In this chapter, we propose a novel method for industrial inspection applications by addressing the view planning and path planning problems simultaneously via a sequencing SCTSP formulation. The proposed approach automatically generates time-efficient, collision-free motion plan

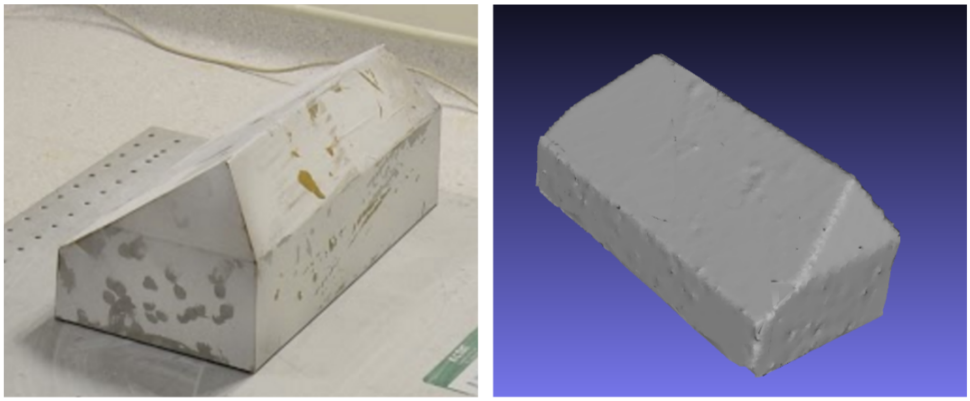


Figure 7.10: The photo and 3D polygonal recontruction model of the target object

that satisfies the requirement of the robotic inspection task. The effectiveness of the method was demonstrated via simulation, where it showed significant improvement over existing methods that solve the two problems independently. In these tests, the method was observed to generate robot paths with an improvement of up to 28.4% in required cycle time. This indicates that the method would prove useful in generating paths for the inspection tasks on an automated production line, where the repetitive nature of the task demands a short cycle time for such operations. An experiment with an ABB robotic inspection system was also conducted to validate the overall approach.

Chapter 8

Coverage Motion Planning with Redundant Robotic System

8.1 Introduction

In recent years, redundant robotic system consisting of a 6-DOF manipulator and 1-DOF turntable has been used by industrial inspection tasks [46][44]. These types of redundant robotic system introduce an infinite number inverse kinematic solutions for each potential viewpoint, which drastically complicates the overall planning process. Motion planning work of a redundant robotic system for inspection applications has been formulated as Traveling Salesman Problem with Neighbourhood (TSPN) [44] or Generalized Traveling Salesman Problem (GTSP) [45]. In these formulations, the authors assume that the viewpoints for inspection are known a priori, so the focus is only on the path planning problem. The viewpoint planning is absent in these work.

In this chapter, we propose a formulation for the shape inspection application with a redundant robotic system that comprises of a 6-DOF manipulator and 1-DOF turntable. In addition, a novel motion planning method for the formulated problem is also proposed. In our work, the overall problem is formulated as combined viewpoint planning and path planning problem. In doing so, we are then able to solve them in one single optimization step with a Random-Key Genetic Algorithm (RKGA) to produce superior results. The main contributions of the planning method proposed in this chapter are listed below:

- A combined sequencing Set-Covering-Generalized-Traveling-Salesman-Problem (SC-GTSP) optimization formulation for inspection applications with a redundant robotic system,
- a motion planning method that solves the proposed sequencing SC-GTSP problem formulation using RKGA, and
- validation of the proposed method by benchmarking it with previous methods.

8.2 Proposed Problem Formulation for The Visual Coverage Planning Problem

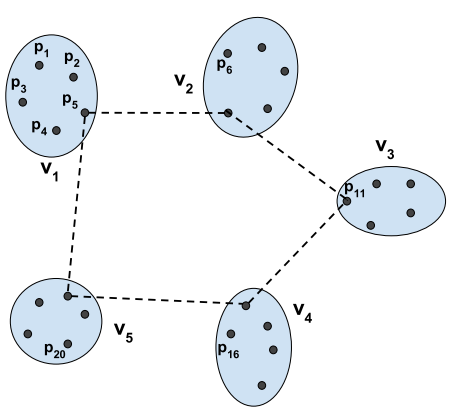
The visual coverage planning problem discussed in this chapter is to find a set of suitable viewpoints, a set of valid robot poses that position the scanner at each viewpoint, and the visiting sequence of the viewpoints. Thus, two combinatorial optimization problems need to be solved: the SCP and GTSP problems. The SCP requires finding a subset that satisfies the viewpoint coverage constraints. The GTSP is an extension of the original TSP problem; it requires finding the shortest Hamiltonian tour of all the clusters. This chapter proposes a combined version of SCP and GTSP problem for the shape inspection application with a redundant robotic system; the proposed formulation is referred as Set-Covering-Generalized-Traveling-Salesman-Problem (SC-GTSP).

In this chapter, we use the following notations in defining the optimization problem:

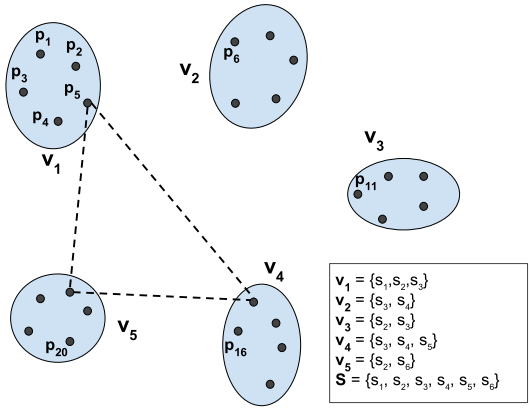
- $\mathbb{S}$: a set of triangular surface patches, s.
- $\mathbb{P}$: a set of robot poses, p.
- $\mathbb{V}$: a set of viewpoints, v; since a redundant robotic system is used [2], there are many poses to achieve one viewpoint, we denote the relationship as $v_q = \{p_{q_1}, p_{q_2}, \cdots, p_{q_l}\}$. In the context of GTSP, v_q is usually called *cluster*.
- $\mathbb{E}$: a set of traveling cost, where $e(p_i, p_j)$ is the traveling cost between robot poses p_i and p_j.

8.2.1 GTSP Formulation for The Redundant Robotic System

The formulation of SCP follows previous formulation [10]. For the path planning of a redundant system, a GTSP formulation is applied. In this formulation, multiple inverse kinematic solutions to place the scanner to the same viewpoint are sampled, the solutions are used to form a cluster. An example of GTSP is shown in Figure 8.1a.



(a) GTSP: finding shortest Hamiltonian path of clusters



(b) SC-GTSP example: a subset of clusters that covers the surface set $\mathbb{S}$ need to be visited

The GTSP formulation of the path planning problem in this chapter is defined as follows. Given the robot pose set $\mathbb{P}$, the viewpoint set $\mathbb{V}$, and n non-empty subsets $v_1, v_2, \cdots, v_n \in \mathbb{V}$; and the set $\mathbb{E}$ is the distance (cost). The objective is to select an element $p_i \in \mathbb{P}$ from every cluster $v_q, q = 1, 2, \cdots, n$, such that the Hamiltonian tour of all the selected elements is minimized.

8.2.2 SC-GTSP Formulation of the Inspection Problem with Redundant Robotic Systems

In our inspection task, redundant viewpoints are sampled such that not all sampled viewpoints need to be visited in order to provide complete coverage of the target object. We only need to visit a set of viewpoints that provides the required coverage. Therefore, the overall optimization problem becomes: finding a subset of robot poses $\mathbb{P}' \in \mathbb{P}$, such that the selected poses are from different clusters, the coverage constraint is satisfied, and the Hamiltonian tour of subset of the clusters $\mathbb{P}'$ is minimized. The visibility information between viewpoint set $\mathbb{V}$ and surface patch set $\mathbb{S}$ is represented by a $n \times m$ binary visibility matrix, $\mathbf{A}$. An example of SC-GTSP is shown in Figure 8.1b.

The objective is to minimize the sum of the inspection cost and traveling cost. In this chapter,

the overall cost is measured as the cycle time of the overall inspection task. The formulation is shown in Eq. (8.1) to Eq. (8.7).

$$\min_{\mathbb{P}',C} \quad \underbrace{\sum_{p_i \in \mathbb{P}'} \omega_i}_{\text{inspection cost}} + \underbrace{\sum_{p_i \in \mathbb{P}'} \sum_{p_j \in \mathbb{P}', i \neq j} c_{ij} e(p_i, p_j)}_{\text{traveling cost}} \tag{8.1}$$

$$\text{subject to:} \quad \sum_{p_i \in \mathbb{P}'} A_{ik} \geq 1 \quad \forall k, k = 1, 2, \cdots m \tag{8.2}$$

$$\sum_{p_i \in \mathbb{P}', i \neq j} c_{ij} = 1, \{j | \forall p_j \in \mathbb{P}'\} \tag{8.3}$$

$$\sum_{p_j \in \mathbb{P}', i \neq j} c_{ij} = 1, \{i | \forall p_i \in \mathbb{P}'\} \tag{8.4}$$

$$\sum_{p_i, p_j \in L} c_{ij} \leq |L| - 1, L \subset \mathbb{P}', |\mathbb{P}'| - 2 \geq |L| \geq 2 \tag{8.5}$$

$$c_{ij} \in \{0, 1\} \tag{8.6}$$

$$p_i \in v_q, p_j \notin v_q, i \neq j, \forall q, q = 1, 2, \cdots n, \tag{8.7}$$

where w_i is the inspection cost at viewpoint v_i; Eq. (8.3) is the coverage constraints; and Eqs. (8.4), (8.5), (8.6) are the TSP subtour elimination constraints within the subset $\mathbb{P}'$, adapted from [86][87]. Constraint in Eq. (8.7) states that any two robot poses in the solution $\mathbb{P}'$ are not from the same cluster. The binary variable c_{ij} indicates that whether a certain path is chosen or not.

Because this formulation is difficult to solve directly, we reformulate the problem in a way that RKGA can be applied.

8.2.3 Reformulation of Sequencing SC-GTSP

The proposed sequencing SC-GTSP formulation is: finding an ordered set $\mathbf{X}$ such that: every element $x \in \mathbf{X}$ is also in $\mathbb{P}$; any two elements are from different clusters; the coverage constraint

is satisfied; and the sum of inspection cost and traveling cost is minimized.

$$\min_{\mathbf{X}} \quad \underbrace{\sum_{x_i \in \mathbf{X}} \omega_i}_{\text{inspection cost}} + \underbrace{\sum_{i=1}^{n_X - 1} e(x_i, x_{i+1}) + e(x_{n_X}, 1)}_{\text{traveling cost}} \tag{8.8}$$

$$\text{subject to:} \quad \sum_{i=1}^{n_X} A_{x_i,j} \geq 1 \quad \forall j, j = 1, 2, \cdots m \tag{8.9}$$

$$x_i \in v_q, x_j \notin v_q, i \neq j, \forall q, q = 1, 2, \cdots n, \tag{8.10}$$

where $A_{x_i,j}$ is the j^{th} element in the row corresponding to viewpoint x_i; n_X is the size of the ordered set $\mathbf{X}$. The traveling sequence is then determined by $\mathbf{X}$, so the TSP constraints are no longer in the formulation.

8.3 Proposed Method

We propose a novel sampling-based motion planning method for an industrial inspection application. We formulate the view planning problem as a SCP, and the path planning problem as a GTSP. The two NP-hard problems are then formulated as a sequencing SC-GTSP problem, and solved by RKGA. The objective of the proposed method is to identify a set of viewpoints and corresponding robotic poses, as well as the visiting sequence and collision-free paths such that the inspection requirement is met and the overall travelling time to visit the viewpoint set is minimized.

In the propose motion planning method, after generating a set of candidate viewpoints $\mathbb{V}$, for each viewpoint v_q, we generate a cluster of robot poses $\{p_{q_1}, p_{q_2}, \cdots, p_{q_l}\}$ that position the scanner at the specified viewpoint. Visibility and traveling costs are then evaluated. Finally, RKGA is used to solve the optimization problem presented in Section 8.2.3. A brief summary of the proposed method is shown in Figure 8.2. The proposed planning algorithm takes the mesh model of the target object, the parameters of the 3D vision sensor, and the robot model as the input, produces a ROS-compatible motion plan as the output. The details of each step in Figure

8.2 are explained in the rest of this section.

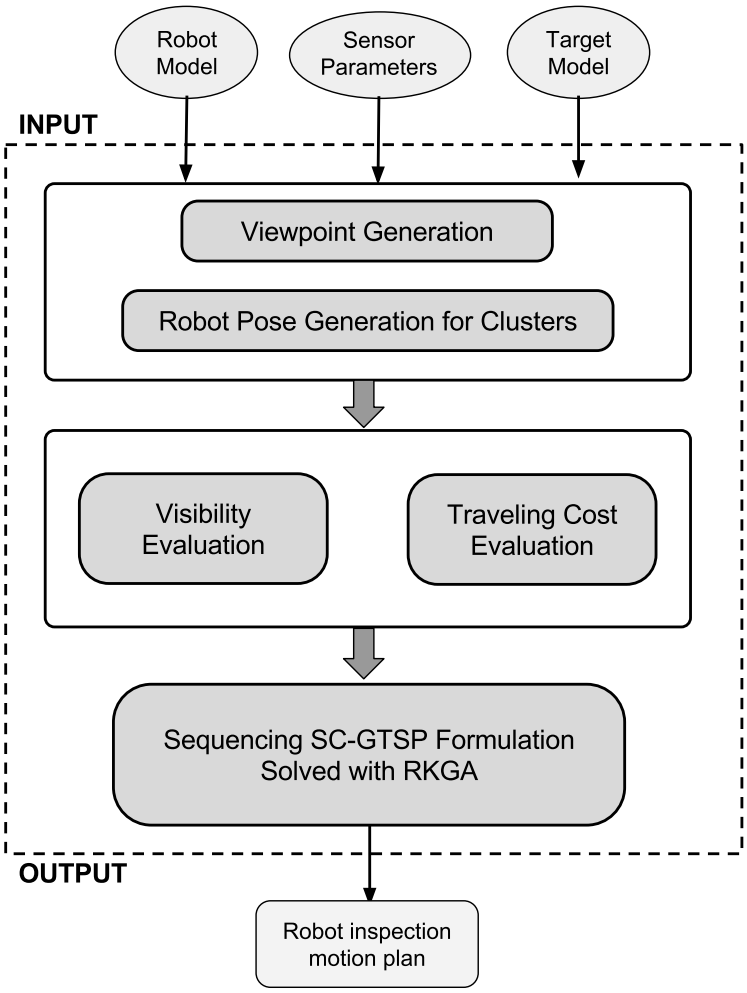


Figure 8.2: The flowchart of the proposed method

8.3.1 Sampling-based Method for Viewpoint and Robot Pose Generation

The first step of the proposed method is to generate candidate viewpoint set $\mathbb{V}$ about the target object. Because of the redundancy of the robotic system, there are multiple poses that place the scanner to the same viewpoint. For each generated viewpoint, therefore, a cluster of corresponding robot poses is also generated via inverse kinematics.

Random Sampling of Viewpoints

A randomized sampling method is used to generate a redundant set of candidate viewpoint positions. The sampling of these viewpoints is performed in Euclidean space around the target object, within the maximum viewing range of the sensor. In addition to the position, the viewing direction is chosen by potential field method [9]. This method ensures that each viewpoint is oriented in a direction that points to a weighted average of nearby surface patches on the target object.

Generating GTSP Robot Poses Clusters for Each Viewpoint

Because a redundant robotic system is used, there are many potential robot poses that position the scanner to any given viewpoint. Random sampling and pose checking are used to generate sample robot poses. In order to generate robot poses for each viewpoint (cluster), we first randomly generate angles for the turntable, and then generate inverse kinematics solutions of the 6-DOF robot based on the turntable angles. If a robot pose is valid, then it is added to the cluster corresponding to certain viewpoint. We continue generating robot poses until the required number is fulfilled for the GTSP set. We generate an equal number of poses in each cluster of the GTSP.

We use a random sampling strategy to generate the poses in each cluster. The robot pose p_{q_k} for viewpoints v_q is first generated through the IKFast inverse kinematics algorithm [88], and the collision check is done through Flexible Collision Library (FCL) [89]. If not enough valid poses are found for a viewpoint, the viewpoint is removed from the sampled candidate viewpoint set.

8.3.2 Evaluation of Viewpoints and Robot Poses

In the second phase of the proposed method, visibility and traveling costs are evaluated. The visibility between each viewpoint and surface patch are evaluated and stored in an $n \times m$ binary matrix, such that the visibility information can be obtained by accessing the corresponding rows

and columns. The pose-to-pose traveling cost among robot poses is then evaluated and stored in the cost set $\mathbb{E}$, where $e(p_i, p_j)$ is the traveling cost measured in time between robot poses p_i and p_j.

Evaluation of Visibility

For the visibility checking between viewpoints and surface patches, a surface patch is visible to a viewpoint if the following conditions are satisfied:

- The surface patch must be in the Field of View (FOV) of the sensor from the viewpoint.
- The surface patch must be in the viewing range of the sensor from the viewpoint.
- The viewing angle must be within a range given by the sensor specifications.
- There must be no occlusion between the viewpoint and surface patches.

These visibility criteria are adapted from previous work [9] [11]. The visibility evaluation process generates the visibility matrix A, The information encoded in this matrix indicates whether a given surface patch is visible to a particular viewpoint: 1 indicates the surface patch is visible to the viewpoint, whereas 0 means it is not visible.

Evaluation of Pose-to-Pose Traveling Costs for GTSP

$e \in \mathbb{E}$ represents the traveling costs between robot poses of GTSP. In this step, the pose-to-pose traveling costs between nodes from different clusters are computed. In the proposed method, Rapid-exploring Random Tree (RRT)-connect [20] algorithm implemented in Open Motion Planning Library (OMPL) [92] is used to query pose-to-pose paths among all the generated poses. The FCL is used for checking collision with the surrounding environment. The MoveIt[90] package from Robotic Operating System (ROS) [91] is used to interface with these packages. After using RRT-connect to compute a collision-free path between robot poses, the path is further parameterized with cubic spline to generate the trajectory, and the robot traveling time is obtained. This information is stored in corresponding e.

8.3.3 Solving the Sequencing Optimization Problem Using RKGA

In the optimization phase, we formulate a sequencing SC-GTSP optimization process using the data from previous steps, and then solve it with RKGA.

RKGA and Encoding/Decoding Strategy

RKGA has been used for sequencing optimization in previous research [69][70]. RKGA encodes information using random-keys stored in the genes of the chromosome. After decoding the random-keys, the fitness value can be computed, as explained in details in [69].

The encoding strategy follows the previous work that is used for encoding GTSP [93]: the random key is a real number within the range of 1 to $l + 1$, where l is the number of elements in the cluster of GTSP. For the decoding strategy, the fractional parts of the random-keys are sorted to determine the sequence, while the integer part is used to select the element in the clusters. A decoding example of a chromosome with five genes is shown below:

$$(2.28, 1.17, 4.74, 4.56, 3.97)$$

$$2(1) \rightarrow 1(2) \rightarrow 4(4) \rightarrow 3(4) \rightarrow 5(3),$$

where in this chromosome, the random-key 1.17 is decoded as $2(1)$ that means the first element in the second cluster.

For the fitness evaluation in RKGA, the first few elements in the sorted genes are added to the solution set until the coverage constraint is met. By doing so, the coverage constraint is evaluated in the fitness function, and the optimization problem becomes unconstrained.

Solving Sequencing SC-GTSP Using RKGA

The sequencing SC-GTSP formulation is slightly modified. Instead of full coverage, a coverage ratio δ is required. We also assume a constant inspection cost ω_0 for each viewpoint. In addition, since the method is designed for production line, the robot should start from its home pose; after

visiting all viewpoints, it should return to its home pose. The modified formulation is shown in Eqs. (8.11)-(8.13).

$$\min_{\mathbf{x},n_x} \quad \underbrace{\omega_0 n_x}_{\text{inspection cost}} + \underbrace{\sum_{i=1}^{n_x-1} e(x_i, x_{i+1}) + e(x_h, x_1) + e(x_{n_x}, x_h)}_{\text{traveling cost}} \tag{8.11}$$

$$\text{subject to:} \quad \sum_{i=1}^{n}((\sum_{j=x_1}^{x_{n_x}} \mathbf{A_j}) \geq 1) \geq \delta * m, \tag{8.12}$$

$$x_i \in v_q, x_j \notin v_q, i \neq j, \forall q, q = 1, 2, \cdots n \tag{8.13}$$

where $\mathbf{A}$ is the visibility matrix that encodes the coverage information between viewpoints set $\mathbb{V}$ and surface patches set $\mathbb{S}$; x_h is the home pose of the robot. The chromosome $\mathbf{x}'$ stores the random keys, after sorting the fractional part, a sequence $\mathbf{x'_{frac}}$ is obtained, with the integer part $\mathbf{x'_{int}}$ that represents the selection in the clusters, the ordered set $\mathbf{x}$ is obtained from $\mathbf{x'_{frac}}$ and $\mathbf{x'_{int}}$, the fitness of the chromosome can be evaluated.

The fitness evaluation strategy is shown in Algo. 6, with the RKGA for SC-GTSP problem, a subset of GTSP clusters is selected such that only first n_x genes that satisfy the coverage constraint is selected, the first n_x genes are then used to calculate the fitness of the chromosome. We also use the swap improvement heuristics proposed in [93] to improve the results.

8.4 Simulations and Results Comparison

In the chapter, we utilize ROS[91] as a platform for the validation of the proposed method. An ABB IRB-4600 industrial manipulator is used. Four target objects with different size and geometry are used for motion planning, as shown in Figure 8.3. The surfaces of these target objects are uniformly re-sampled using Bubble Mesh[53].

Algorithm 6 Evaluating the Fitness of a Given Chromosome

Input: The chromosome with the random keys stored in its genes, $\mathbf{x}'$; the traveling cost, $\mathbb{E}$; the visibility matrix, A; and the desired coverage ratio, δ; inspection cost at viewpoint ω_0;

Output: The cost of the input chromosome, y, and the number of first few viewpoints needed for coverage constraint, n_x

$y = \omega_0$

$n_x = 0$

$\mathbf{x}'_{\mathbf{int}}, \mathbf{x}'_{\mathbf{frac}} \leftarrow$ Separate($\mathbf{x}'$)

$\mathbf{x} \leftarrow$ Sort($\mathbf{x}'_{\mathbf{frac}}$)

$y = y + e(x_h, \text{findInd}(x_n, \mathbf{x}'_{\mathbf{int}}))$

for each $x_i \in \mathbf{x}$ and $i > 1$ **do**

    $y = y + \omega_0$

    $y = y + e(\text{findInd}(x_i, \mathbf{x}'_{\mathbf{int}}), \text{findInd}(x_{i+1}, \mathbf{x}'_{\mathbf{int}}))$

    $n_x = n_x + 1$

    $\delta' \leftarrow$ findCoverageRatio(A, x_i)

    if $\delta' > \delta$ **then**

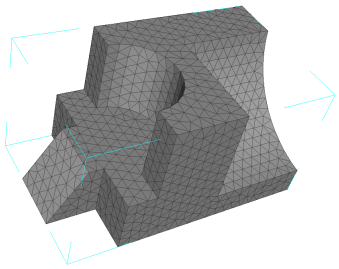
        break

    end if

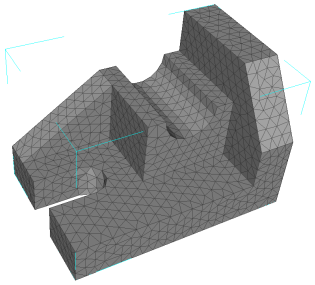
end for

$y = y + e(\text{findInd}(x_{n_x}, \mathbf{x}'_{\mathbf{int}}), x_h)$

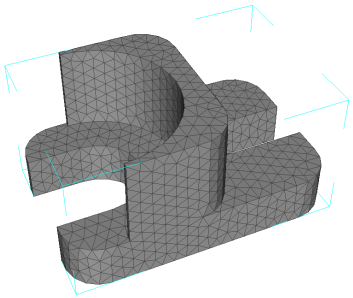
return y, n_x



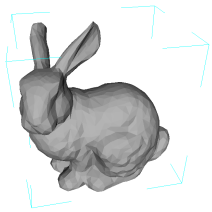
(a) Target Object 1: the bounding box size is $0.69 \times 0.41m \times 0.36m$



(b) Target Object 2: the bounding box size is $0.67m \times 0.45m \times 0.45m$



(c) Target Object 3: the bounding box size is $0.47 \times 0.69m \times 0.36m$



(d) Target object 4: the bounding box size is $0.22m \times 0.30m \times 0.30m$

Figure 8.3: The target Objects

8.4.1 Setup

As shown in Figure 8.4, a turntable of 0.5 meters in height is placed 1.6 meters away in front of the robot. The turntable is modelled as a cylinder in order to reduce the computational cost for collision check. The target object is placed at the center of the turntable. The turntable has 1-DOF that rotates along z-axis. Two different speeds of the turntable are used, and the labels "HS" and "LS" are used to represent $6.98rad/s$ and $2.88rad/s$ as the maximum speed, following the same setup from previous work [44]. We compare the results of 200, 300 and 400 randomly sampled viewpoints. For each viewpoint (cluster), five corresponding robot poses are generated. An Artec Eva 3D scanner is used. Table 8.1 lists the parameters for visibility and coverage requirement. We assume a constant inspection time of one second for taking measurement at

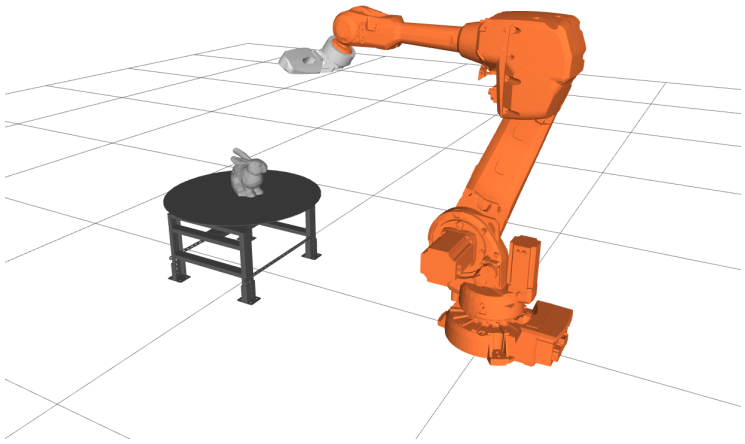


Figure 8.4: Setup of the motion planning: a 3D scanner (in light grey) is mounted on the end-effector of the industrial manipulator (in orange), the target object (in dark grey) is placed on the turntable (in black), the displayed robot pose is the home pose

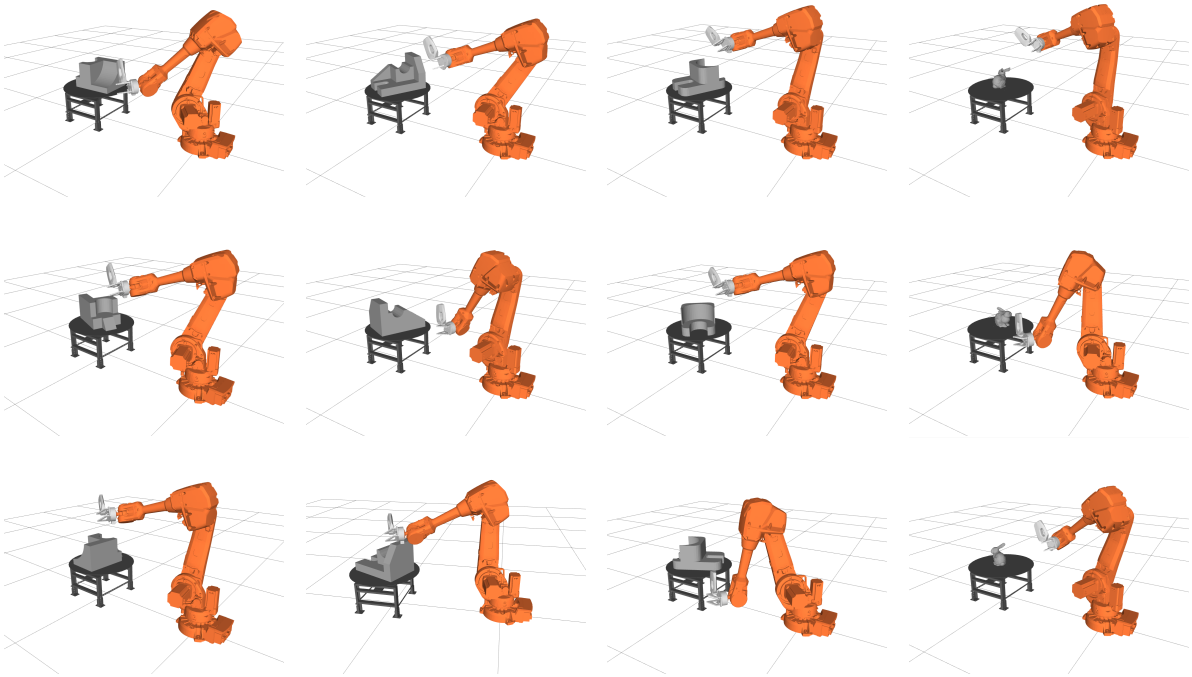


Figure 8.5: Example robot poses for inspection

each viewpoint.

Table 8.1: Sensor parameters

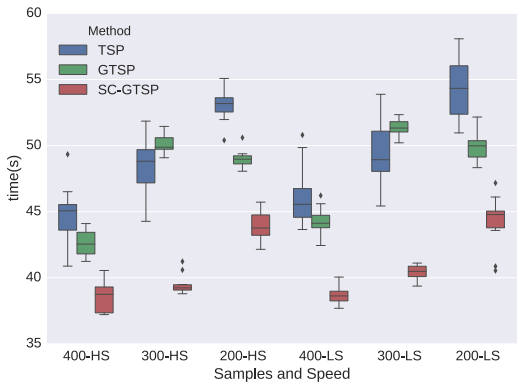
FOV	$30°, 21°$
Camera Pixels	536×371
Viewing Angle	$75°$
Viewing Range (meters)	$(0.4, 1.0)$
Coverage Ratio δ	99.5%

8.4.2 Results Comparison

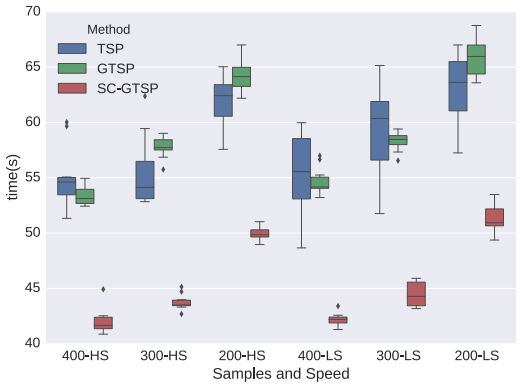
In this chapter, we benchmark our method with two previous approaches. The first approach to be compared solve the SCP with Greedy method [9], and GTSP with RKGA (labeled as GTSP); the other approach (labelled as TSP) solves SCP and TSP as a SCTSP problem by RKGA without considering the redundancy [11], and a random valid pose is used for the TSP problem. Ten trials of each method have been performed for the results comparison to average out the non-deterministic behaviour of the used methods. The results are plotted in Figure 8.6.

The benchmarked results show that the proposed method outperforms the previous methods in all planning cases. The results for Target Object 1 are shown in Table 8.2; the proposed method requires 14.5% and 17.2% less total inspection time than previous approaches. The simulation results with Target Object 2 also indicate that the proposed method performs better compared to the other methods; as shown in Table 8.3, the proposed method yields the total inspection time 22.7% and 21.9% less than other methods. In the results comparison of Target Object 3 shown in Table 8.4, the proposed method requires 18.7% and 20.5% less inspection time. The forth results comparison shows that the proposed method performs better than the previous method by requiring 17.6% and 23.3% less total inspection time, as shown in Table 8.5. The largest inspection time reduction is found in results comparison of Target Object 4, where 28.1% reduction has been achieved. Figure 8.5 shows the examples of robotic poses on selected viewpoints, and Figure 8.6 shows the detailed results distributions of ten trials. Example

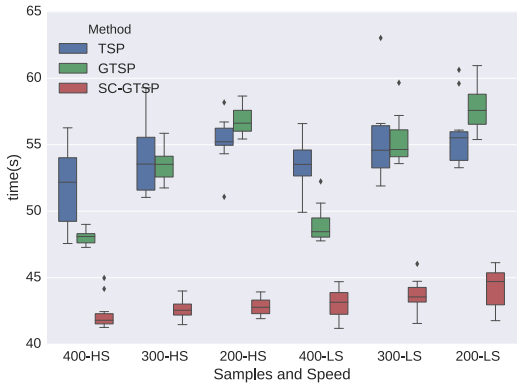
resultant trajectories are shown in Figure 8.7.



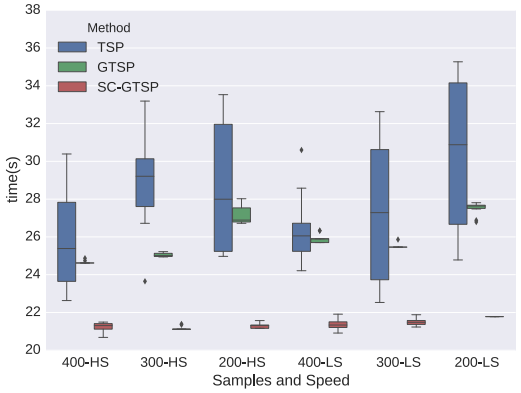
(a) Targat Object 1



(b) Target Object 2



(c) Target Object 3



(d) Target Object 4

Figure 8.6: The required inspection time for the target objects (based on ten trials)

Table 8.2: Average total time for inspecting Target Object 1

	200 samples		**300 samples**		**400 samples**	
	HS	**LS**	**HS**	**LS**	**HS**	**LS**
TSP	53.0	54.4	48.5	49.5	44.8	46.2
GTSP	49.1	50.0	50.1	51.3	42.6	44.2
SC-GTSP	43.9	44.2	39.5	40.4	38.5	38.7

Table 8.3: Average total time for inspecting Target Object 2

	200 samples		300 samples		400 samples	
	HS	**LS**	**HS**	**LS**	**HS**	**LS**
TSP	61.9	63.1	55.5	59.4	55.0	55.3
GTSP	64.3	65.8	57.8	58.3	53.3	54.7
SC-GTSP	49.9	51.4	43.8	44.5	42.0	42.2

Table 8.4: Average total time for inspecting Target Object 3

	200 samples		300 samples		400 samples	
	HS	**LS**	**HS**	**LS**	**HS**	**LS**
TSP	55.3	55.8	53.9	56.2	51.8	53.4
GTSP	56.7	57.8	53.5	55.4	48.0	49.0
SC-GTSP	42.8	44.2	42.7	43.8	42.3	43.1

Table 8.5: Average total time for inspecting Target Object 4

	200 samples		300 samples		400 samples	
	HS	**LS**	**HS**	**LS**	**HS**	**LS**
TSP	28.7	30.3	29.0	27.4	25.9	26.4
GTSP	27.2	27.5	25.1	25.5	24.7	25.9
SC-GTSP	21.3	21.8	21.2	21.5	21.2	21.3

8.4.3 Discussion

The results from Section 8.4.2 show that the proposed method outperforms the traditional approaches. The main reason is that the traditional approaches have separate optimization processes. In the previous methods, finding the minimum inspection cost in SCP optimization does

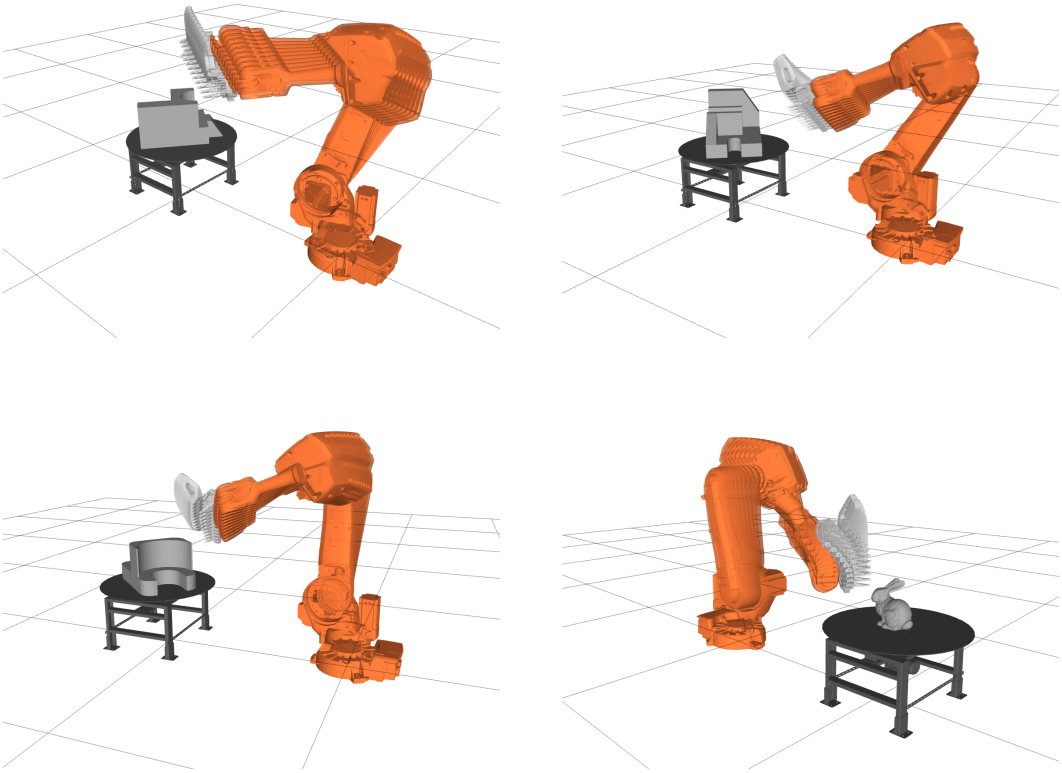


Figure 8.7: The example robotic manipulator trajectories.

not consider the traveling cost. Thus, in final optimization result, there might be less inspection cost but more overall cost because traveling cost is high. In addition, considering the redundancy with a GTSP formulation of the traveling cost improves the optimization results. The redundant robotic systems also enlarge the workspace of the robot, such that larger target object can be inspected. With ROS-industrial package [94], the resultant motion plan can be directly used by the robotic manipulator.

8.5 Summary

In this chapter, a novel motion planning method is proposed for the 7-DOF redundant robotic system for industrial inspection. The proposed method combined the SCP and GTSP as a single

sequencing SC-GTSP optimization problem, and solve it using RKGA to achieve better optimization results. Several planning cases have been performed using the proposed method to validate its effectiveness. With these cases, the proposed method performed better than the previous approaches. Our future work will consider other optimization techniques for the combined SC-GTSP problem.

Chapter 9

Conclusion

In this chapter, we conclude the work of this thesis and make recommendation for future directions.

9.1 Conclusion

In summary, this thesis presents several novel techniques to improve the planning results of Visual Coverage Planning Problem (VCPP), for inspection, surveillance and shape reconstruction applications with robotic manipulators and UAVs.

First, two view planning algorithms have been developed in order to generate better candidate viewpoints, hence better view planning and overall VCPP results can be then achieved. The first view planning method utilizes the voxel dilation to create a sampling space, and then randomly samples viewpoint positions in the dilated volume. The viewing direction is generated through local probabilistic potential field method [9]. The method significantly reduces the sampling space while maintaining good exploration of the feasible space. The sampling-based view planning method makes the viewpoint generation process more effective, and thus achieves high coverage ratio and less number of viewpoints for required coverage.

Additionally, a Medial Object (MO) based view planning method has also been developed to further improve the efficiency of candidate viewpoint generation, as an important improvement of the sampling-based view planning method. The MO-based method samples the viewpoints around MO using a Gaussian sampling strategy. Similar to other MO-based pose-to-pose planning methods (MA-RRT [27], MA-PRM [26] [25]), the MO-based view planning method explores the space more efficiently by using geometric information of the environment. Thus, the MO-based view planning method is able to achieve higher coverage ratio and fewer viewpoints for required coverage, as compared to previous methods.

The view planning methods have been extended to 3D shape reconstruction of building application, using 2D publicly available map data and the estimated height. It is found that the adapted view planning method effectively finds the viewpoints to reconstruct the 3D model of

the buildings with good quality.

In addition to the view planning work with UAV, the robotic manipulator is used as a platform for VCPP problem in industrial shape inspection applications. We first propose a learning-based robotic calibration method using Product-Of-Exponential (POE) and Gaussian Process regression to achieve high calibration accuracy of the robot manipulator. The proposed calibration method helps us to place the vision sensor to desired viewpoints more accurately. For the inspection application with robotic manipulator, we formulate the overall VCPP problem by combining the view planning and path planning problems. A Random-Key Genetic Algorithm (RKGA) is used to solve the combined sequencing Set-Covering-Traveling-Salesman-Problem (SCTSP). The proposed planning method achieves better results compared to previous methods.

Moreover, we also have proposed a planning algorithm for the industrial inspection applications using a redundant robotic system. A redundant robotic system with 6 Degree-of-Freedom (DOF) robot manipulator and 1 DOF turntable is used for these inspection cases to achieve larger workspace and shorter cycle time. With combining the view planning problem and path planning problem, we formulate the overall VCPP as a Set-Covering-Generalized-Traveling-Salesman-Problem (SC-GTSP) and solve it using RKGA. We have also demonstrated that less cycle time is required compared to the previous methods.

9.2 Future Work

Motion planning for VCPP has many promising future directions that worth further exploration. Several recommendations for future work are listed below:

- VCPP with multiple robots is an interesting direction for future work. The multiple robot coverage planning problem is more complicated and useful in practice. Especially for UAV application, using multiple UAVs for the coverage tasks such as inspection and surveillance would increase the efficiency significantly. Finding an efficient and feasible plan for multiple robots with acceptable computational cost is of great interests in practice.

- Extending current work to online planning and re-planning is also very useful. In many applications, although the geometric model is available, the pose of the target object is not known prior to the planning process. There might be non-negligible uncertainties of the poses in certain production environment. The planning algorithm with online planning and re-planning capability would be able to deal with these cases.
- Semi-model-based and non-model-based coverage planning with single or multiple robot(s) also worth exploration in the future. Learning-based methods can be applied to these planning work to mimic human behaviour, in order to explore the unknown environment.
- Planning with consideration of continuous path instead of discrete viewpoints for the coverage is a promising future direction, for example, planning continuous path for a UAV with RGB camera is an interesting and challenging topic.
- Considering heterogeneous sensors for coverage planning problem is a practically interesting direction. When working with practical applications, multiple vision sensors with different specifications are usually used for the coverage, thus, coverage planning with heterogeneous sensors is necessary in these instances.
- Another research direction worth considerations is to extend the current planning algorithm to other similar coverage planning applications such as planning for spray painting, 3D printing, and wielding.

Bibliography

[1] J. Martinez-De Dios and A. Ollero, "Automatic detection of windows thermal heat losses in buildings using uavs," in *WAC'06. World Automation Congress, 2006.* IEEE, 2006, pp. 1–6.

[2] I. Gentilini, "Multi-Goal Path Optimization for Robotic Systems with Redundancy based on the Traveling Salesman Problem with Neighborhoods," Ph.D. dissertation, Carnegie Mellon University, May 2012.

[3] M. Mahmud, D. Joannic, and J.-F. Fontaine, "3d digitizing path planning for part inspection with laser scanning," in *International Conference on Quality Control by Artificial Vision*. International Society for Optics and Photonics, 2007, pp. 635 603–635 603.

[4] C. N. Macleod, G. Dobie, S. G. Pierce, R. Summan, and M. Morozov, "Machining-based coverage path planning for automated structural inspection," *IEEE Transactions on Automation Science and Engineering*, 2016.

[5] R. Raffaeli, M. Mengoni, M. Germani, and F. Mandorli, "Off-line view planning for the inspection of mechanical parts," *International Journal on Interactive Design and Manufacturing*, vol. 7, no. 1, pp. 1–12, 2013.

[6] H. Choset, "Coverage for robotics, a survey of recent results," *Annals of mathematics and artificial intelligence*, vol. 31, no. 1, pp. 113–126, 2001.

[7] B. Englot and F. S. Hover, "Sampling-based coverage path planning for inspection of complex structures." in *ICAPS*, 2012.

[8] R. Almadhoun, T. Taha, L. Seneviratne, J. Dias, and G. Cai, "A survey on inspecting structures using robotic systems," *International Journal of Advanced Robotic Systems*, vol. 13, no. 6, p. 1729881416663664, 2016.

[9] W. Jing, J. Polden, W. Lin, and K. Shimada, "Sampling-based view planning for 3d visual coverage task with unmanned aerial vehicle," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2016, pp. 1808–1815.

[10] W. Jing, J. Polden, P. Y. Tao, W. Lin, and K. Shimada, "View planning for 3d shape reconstruction of buildings with unmanned aerial vehicles," in *International Conference on Control, Automation, Robotics and Vision*. IEEE, 2016, pp. 1–6.

[11] W. Jing, J. Polden, P. Y. Tao, C. F. Goh, W. Lin, and K. Shimada, "Model-based coverage motion planning for industrial 3d shape inspection applications," to appear, IEEE Conference on Automation Science and Engineering. IEEE, 2017.

[12] W. Jing, J. Polden, C. F. Goh, M. Rajaraman, W. Lin, and K. Shimada, "Sampling-based coverage motion planning for industrial inspection application with redundant robotic system," to appear, IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2017.

[13] W. Jing, P. Y. Tao, G. Yang, and K. Shimada, "Calibration of industry robots with consideration of loading effects using product-of-exponential (POE) and gaussian process (GP)," in *IEEE International Conference on Robotics and Automation*. IEEE, 2016, pp. 4380–4385.

[14] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[15] T. Lozano-Pérez and M. A. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Communications of the ACM*, vol. 22, no. 10, pp. 560–570, 1979.

[16] H. Choset and P. Pignon, "Coverage path planning: The boustrophedon cellular decomposition," in *Field and service robotics*. Springer, 1998, pp. 203–209.

[17] F. Lingelbach, "Path planning using probabilistic cell decomposition," in *IEEE International Conference on Robotics and Automation*, vol. 1. IEEE, 2004, pp. 467–472.

[18] M. Elbanhawi and M. Simic, "Sampling-based robot motion planning: A review," *IEEE Access*, vol. 2, pp. 56–77, 2014.

[19] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[20] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *IEEE International Conference on Robotics and Automation*, vol. 2. IEEE, 2000, pp. 995–1001.

[21] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[22] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[23] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed rrt*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 2997–3004.

[24] ——, "Batch informed trees (bit*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs," in *IEEE International Conference on Robotics and Automation*. IEEE, 2015, pp. 3067–3074.

[25] C. Holleman and L. E. Kavraki, "A framework for using the workspace medial axis in prm planners," in *IEEE International Conference on Robotics and Automation*, vol. 2. IEEE, 2000, pp. 1408–1413.

[26] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, "Maprm: A probabilistic roadmap planner with sampling on the medial axis of the free space," in *IEEE International Conference on*

Robotics and Automation, vol. 2. IEEE, 1999, pp. 1024–1031.

[27] J. Denny, E. Greco, S. Thomas, and N. M. Amato, "Marrt: Medial axis biased rapidly-exploring random trees," in *IEEE International Conference on Robotics and Automation*. IEEE, 2014, pp. 90–97.

[28] M. Price, C. Stops, and G. Butlin, "A medial object toolkit for meshing and other applications," in *Proceedings of 4th International Meshing Roundtable*. Citeseer, 1995, pp. 219–229.

[29] 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.

[30] W. Scott, G. Roth, and J.-F. Rivest, "View planning for automated 3d object reconstruction inspection," *ACM Computing Surveys*, vol. 35, no. 1, 2003.

[31] K. A. Tarabanis, R. Y. Tsai, and P. K. Allen, "The mvp sensor planning system for robotic vision tasks," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 1, pp. 72–85, 1995.

[32] G. H. Tarbox and S. N. Gottschlich, "Planning for complete sensor coverage in inspection," *Computer Vision and Image Understanding*, vol. 61, no. 1, pp. 84–111, 1995.

[33] W. R. Scott, "Model-based view planning," *Machine Vision and Applications*, vol. 20, no. 1, pp. 47–69, 2009.

[34] K. A. Tarabanis, P. K. Allen, and R. Y. Tsai, "A survey of sensor planning in computer vision," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 1, pp. 86–104, 1995.

[35] S. Chen, Y. Li, and N. M. Kwok, "Active vision in robotic systems: A survey of recent developments," *The International Journal of Robotics Research*, vol. 30, no. 11, pp. 1343–1377, 2011.

[36] A. Mavrinac and X. Chen, "Modeling coverage in camera networks: A survey," *International journal of computer vision*, vol. 101, no. 1, pp. 205–226, 2013.

[37] U. Nilsson, P. Ogren, and J. Thunberg, "Optimal positioning of surveillance ugvs," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 2539–2544.

[38] L. Geng, Y. Zhang, J. Wang, J. Y. Fuh, and S. Teo, "Mission planning of autonomous uavs for urban surveillance with evolutionary algorithms," in *IEEE International Conference on Control and Automation*. IEEE, 2013, pp. 828–833.

[39] E. Semsch, M. Jakob, D. Pavlicek, and M. Pechoucek, "Autonomous uav surveillance in complex urban environments," in *IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technologies*, vol. 2. IET, 2009, pp. 82–85.

[40] Y. Li and Z. Liu, "Information entropy-based viewpoint planning for 3-d object reconstruction," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 324–337, 2005.

[41] J. I. Vasquez-Gomez, L. E. Sucar, and R. Murrieta-Cid, "View/state planning for three-dimensional object reconstruction under uncertainty," *Autonomous Robots*, vol. 41, no. 1, pp. 89–109, 2017.

[42] C. Potthast and G. S. Sukhatme, "Next best view estimation with eye in hand camera," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[43] J. I. Vasquez-Gomez, L. E. Sucar, and R. Murrieta-Cid, "View planning for 3d object reconstruction with a mobile manipulator robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 4227–4233.

[44] I. Gentilini, K. Nagamatsu, and K. Shimada, "Cycle time based multi-goal path optimization for redundant robotic systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1786–1792.

[45] S. Alatartsev, S. Stellmacher, and F. Ortmeier, "Robotic task sequencing problem: A survey," *Journal of Intelligent & Robotic Systems*, vol. 80, no. 2, p. 279, 2015.

[46] L. B. Gueta, R. Chiba, J. Ota, T. Ueyama, and T. Arai, "Coordinated motion control of a

robot arm and a positioning table with arrangement of multiple goals," in *IEEE International Conference on Robotics and Automation*. IEEE, 2008, pp. 2252–2258.

[47] S. Chen and Y. Li, "Automatic sensor placement for model-based robot vision," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 34, no. 1, pp. 393–408, 2004.

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

[49] P. Wang, K. Gupta, and R. Krishnamurti, "Some complexity results for metric view planning problem with traveling cost and visibility range," *IEEE Transactions on Automation Science and Engineering*, vol. 8, no. 3, pp. 654–659, 2011.

[50] B. Englot and F. S. Hover, "Three-dimensional coverage planning for an underwater inspection robot," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1048–1073, 2013.

[51] G. Papadopoulos, H. Kurniawati, and N. M. Patrikalakis, "Asymptotically optimal inspection planning using systems with differential constraints," in *IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 4126–4133.

[52] A. Bircher, K. Alexis, M. Burri, P. Oettershagen, S. Omari, T. Mantel, and R. Siegwart, "Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics," in *IEEE International Conference on Robotics and Automation*. IEEE, 2015, pp. 6423–6430.

[53] K. Shimada and D. C. Gossard, "Bubble mesh: automated triangular meshing of non-manifold geometry by sphere packing," in *Proceedings of the third ACM symposium on Solid modeling and applications*. ACM, 1995, pp. 409–419.

[54] R. M. Haralick, S. R. Sternberg, and X. Zhuang, "Image analysis using mathematical mor-

phology," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 4, pp. 532–550, 1987.

[55] T. Möller and B. Trumbore, "Fast, minimum storage ray-triangle intersection," *Journal of graphics tools*, vol. 2, no. 1, pp. 21–28, 1997.

[56] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[57] V. Chvatal, "A greedy heuristic for the set-covering problem," *Mathematics of operations research*, vol. 4, no. 3, pp. 233–235, 1979.

[58] Q.-S. Hua, Y. Wang, D. Yu, and F. Lau, "Dynamic programming based algorithms for set multicover and multiset multicover problems," *Theoretical Computer Science*, vol. 411, no. 26, pp. 2467–2474, 2010.

[59] W.-C. Huang, C.-Y. Kao, and J.-T. Horng, "A genetic algorithm approach for set covering problems," in *IEEE World Congress on Computational Intelligence*. IEEE, 1994, pp. 569–574.

[60] C. Wu, "Visualsfm: A visual structure from motion system," *URL http://ccwu. me/vsfm/*, 2011.

[61] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz, "Multicore bundle adjustment," in *2011 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 3057–3064.

[62] M. Jancosek and T. Pajdla, "Multi-view reconstruction preserving weakly-supported surfaces," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 3121–3128.

[63] K. Tarabanis, R. Y. Tsai, and A. Kaul, "Computing occlusion-free viewpoints," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 3, pp. 279–292, 1996.

[64] S. Son and K. H. Lee, "Automated scan plan generation using stl meshes for 3d stripe-type

laser scanner," in *Computational Science and Its ApplicationsICCSA*. Springer, 2003, pp. 741–750.

[65] H. Blum and R. N. Nagel, "Shape description using weighted symmetric axis features," *Pattern recognition*, vol. 10, no. 3, pp. 167–180, 1978.

[66] H.-Y. C. Yeh, J. Denny, A. Lindsey, S. Thomas, and N. M. Amato, "Umaprm: Uniformly sampling the medial axis," in *IEEE International Conference on Robotics and Automation*. IEEE, 2014, pp. 5798–5803.

[67] D. Reniers, J. J. van Wijk, and A. Telea, "Computing multiscale curve and surface skeletons of genus 0 shapes using a global importance measure," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 2, pp. 355–368, 2008.

[68] V. Boor, M. H. Overmars, and A. F. van der Stappen, "The gaussian sampling strategy for probabilistic roadmap planners," in *IEEE International Conference on Robotics and Automation*, vol. 2. IEEE, 1999, pp. 1018–1023.

[69] J. C. Bean, "Genetic algorithms and random keys for sequencing and optimization," *ORSA journal on computing*, vol. 6, no. 2, pp. 154–160, 1994.

[70] J. F. Gonçalves and M. G. Resende, "Biased random-key genetic algorithms for combinatorial optimization," *Journal of Heuristics*, vol. 17, no. 5, pp. 487–525, 2011.

[71] Mathworks, *MATLab Genetic Algorithm*. The MathWorks Inc., 2017. [Online]. Available: http://www.mathworks.com/help/gads/genetic-algorithm.html

[72] B. Shumaker and R. Sinnott, "Astronomical computing: 1. computing under the open sky. 2. virtues of the haversine." *Sky and telescope*, vol. 68, pp. 158–159, 1984.

[73] DJI, "DJI phantom 3 advanced specs," *URL http://www.dji.com/product/phantom-3-adv/info#specs*, 2016.

[74] 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.

[75] C. Wu, "Siftgpu: A gpu implementation of scale invariant feature transform," *URL http://cs. unc. edu/~ ccwu/siftgpu*, 2011.

[76] S. Fuhrmann, F. Langguth, and M. Goesele, "Mve-a multiview reconstruction environment," in *Proceedings of the Eurographics Workshop on Graphics and Cultural Heritage (GCH)*, vol. 6, no. 7. Citeseer, 2014, p. 8.

[77] I.-M. Chen, G. Yang, C. T. Tan, and S. H. Yeo, "Local POE model for robot kinematic calibration," *Mechanism and Machine Theory*, vol. 36, no. 11, pp. 1215–1239, 2001.

[78] P. Y. Tao and G. Yang, "Calibration of industrial robots with product-of-exponential (poe) model and adaptive neural networks," in *IEEE International Conference on Robotics and Automation*. IEEE, 2015, pp. 1448–1454.

[79] F. C. Park, "Computational aspects of the product-of-exponentials formula for robot kinematics," *IEEE Transactions on Automatic Control*, vol. 39, no. 3, pp. 643–647, 1994.

[80] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: the MIT Press, 2006.

[81] S. Park, S. K. Mustafa, and K. Shimada, "Learning-based robot control with localized sparse online Gaussian process," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1202–1207.

[82] C. M. Bishop, *Pattern Recognition and Machine Learning*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[83] P. Y. Tao, G. Yang, and M. Tomizuka, "A calibration framework for industrial robotic work cells," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. IEEE, 2013, pp. 1637–1642.

[84] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python,"

Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[85] Q. Wu, J. Lu, W. Zou, and D. Xu, "Path planning for surface inspection on a robot-based scanning system," in *IEEE International Conference on Mechatronics and Automation*. IEEE, 2015, pp. 2284–2289.

[86] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem," *Journal of the operations research society of America*, vol. 2, no. 4, pp. 393–410, 1954.

[87] G. Laporte, "The traveling salesman problem: An overview of exact and approximate algorithms," *European Journal of Operational Research*, vol. 59, no. 2, pp. 231–247, 1992.

[88] R. Diankov and J. Kuffner, "Openrave: A planning architecture for autonomous robotics," *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34*, vol. 79, 2008.

[89] J. Pan, S. Chitta, and D. Manocha, "Fcl: A general purpose library for collision and proximity queries," in *IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 3859–3866.

[90] I. A. Sucan and S. Chitta, "Moveit!" *Online Available: http://moveit. ros. org*, 2013.

[91] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[92] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, December 2012, http://ompl.kavrakilab.org.

[93] L. V. Snyder and M. S. Daskin, "A random-key genetic algorithm for the generalized traveling salesman problem," *European Journal of Operational Research*, vol. 174, no. 1, pp. 38–53, 2006.

[94] S. Edwards and C. Lewis, "Ros-industrial: applying the robot operating system (ros) to

industrial applications," in *IEEE International Conference on Robotics and Automation, ECHORD Workshop*, 2012.