11-2-2019

A Co-optimal Coverage Path Planning Method for Aerial Scanning of Complex Structures

Zhexiong Shang
University of Nebraska - Lincoln, zshang@unomaha.edu

justin Bradley
University of Nebraska-Lincoln, justin.bradley@unl.edu

Zhigang Shen
University of Nebraska - Lincoln, shen@unl.edu

A Co-optimal Coverage Path Planning Method for Aerial Scanning of Complex Structures

Zhexiong Shang, Ph.D. Candidate, Durham School of Architectural Engineering and Construction, University of Nebraska-Lincoln, Lincoln, NE 68588, USA, szx0112@huskers.unl.edu

Justin Bradley, Ph.D., Assistant Professor, Department of Computer Science and Engineering, University of Nebraska-Lincoln, Lincoln, NE 68588, USA, justin.bradley@unl.edu

Zhigang Shen, Ph.D. Associate Professor, Durham School of Architectural Engineering and Construction, University of Nebraska-Lincoln, Lincoln, NE 68588, USA, shen@unl.edu

Abstract

The utilization of unmanned aerial vehicles (UAVs) in survey and inspection of civil infrastructure has been growing rapidly. However, computationally efficient solvers that find optimal flight paths while ensuring high-quality data acquisition of the complete 3D structure remains a difficult problem. Existing solvers typically prioritize efficient flight paths, or coverage, or reducing computational complexity of the algorithm — but these objectives are not co-optimized holistically. In this work we introduce a co-optimal coverage path planning (CCPP) method that simultaneously co-optimizes the UAV path, the **quality** of the captured images, and reducing computational complexity of the solver all while adhering to safety and inspection requirements. The result is a highly parallelizable algorithm that produces more efficient paths where quality of the useful image data is improved. The path optimization algorithm utilizes a particle swarm optimization (PSO) framework which iteratively optimizes the coverage paths without needing to discretize the motion space or simplify the sensing models as is done in similar methods. The core of the method consists of a cost function that measures both the **quality** and **efficiency** of a coverage inspection path, and a greedy heuristic for the optimization enhancement by aggressively exploring the viewpoints search spaces. To assess the proposed method, a coverage path quality evaluation method is also presented in this research, which can be utilized as the benchmark for assessing other CPP methods for structural inspection purpose. The effectiveness of the proposed method is demonstrated by comparing the quality and efficiency of the proposed approach with the state-of-art through both synthetic and real-world scenes. The experiments show that our method enables significant performance improvement in coverage inspection quality while preserving the path efficiency on different test geometries.

Keywords—Coverage Path Planning, Unmanned Aerial Vehicle, Particle Swarm Optimization, Viewpoint Quality

Nomenclature

Notation	Definition
Ω	The triangular mesh model
$\mathcal{M}$	The number of triangular surface planes on the model
Ω_i	The ith surface of model
c_i	The centroid at the ith surface plane
n_i	The normal vector of ith surface plane
q_i	The camera view at ith model surface
FOV	Camera field of view
DOV	Camera depth of view
$\mathcal{D}_r$	Maximal distance with minimal acceptable spatial resolution, defined by camera intrinsic parameters and inspection requirements
$(\mathcal{D}_{min}, \mathcal{D}_{max})$	Acceptable distance between camera to target surface, determined by DOV and safety concerns

η_{max}	Maximal observation angle between camera ray and the surface normal, determined by FOV and inspection requirements
$(\theta_{min}, \theta_{max})$	Acceptable pitch rotation range of onboard gimbal system
$\mathcal{P}$	The population of particles
$\mathcal{N}$	The number of particles in population
$\mathcal{G}$	The number of iterations
$\mathscr{p}_j$	The jth particle in population with each particle denoted as a collection of camera views and an associated shortest path index
τ	The shortest path index vector
$\mathcal{Q}$	The metric evaluates the quality of inspection
E	The metric evaluates the path efficiency

1. INTRODUCTION

Designing paths for a UAV to explore a structure or environment while avoiding obstacles is called coverage path planning (CPP) (Galceran & Carreras, 2013). Over the last decade, different CPP methods have been developed for performing a variety of real-world inspection applications including structural health monitoring (Bircher et al., 2016), augmented reality (Papachristos & Alexis, 2016), robot manipulation (Krainin, Curless, & Fox, 2011), and stereo reconstruction (Jing, Polden, Lin, & Shimada, 2016). Visual structural inspections typically require high-quality images where hairline cracks can be detected. The majority of existing CPP optimization approaches focus on minimizing path distance while ensuring coverage and flight safety (Englot & Hover, 2010; Englot & Hover, 2012a). Few studies were found to incorporate inspection images' quality into the path optimization. Without the consideration of coverage-image quality, these methods may result in collecting a set of suboptimal images that fail to capture enough structural defects features for image analysis. Additionally, CPP problems are computationally complex since they encompass the classical art gallery problem (AGP) (O'rourke, 1987) and the traveling salesman problem (TSP) (Flood, 1956), making the problem intrinsically non-deterministic polynomial time hard (NP-hard). Existing algorithms either discretize the motion spaces of the UAVs or restrict the degrees of freedom (DoF) of onboard sensors in the continuous space. These strategies allow the problem to be formulated into a convex optimization schema – greatly simplifying the solution. However, without incorporating the full sensing capability and the UAV movability into the path generation, these simplified models may fail to provide the full coverage or the global optimal when the inspecting structure geometry is complex.

To address these issues, we propose a new CPP approach, a co-optimal coverage path planner (CCPP), for aerial structural inspection. We present a relationship between the inspection quality and the operational cost of a coverage path and generate flight trajectories that balance image quality and path distance simultaneously. We achieve this through a novel cost function that measures both the quality and the efficiency of inspection with the concerns of task urgency. We wrap the CPP problem into a particle swarm optimization (PSO) framework that incorporates the full sensing models (e.g., camera field of view, 6d camera pose) and the physical constraints (e.g., site obstacles) into the 3D path generation without the need to convexify and discretize the model. The proposed PSO framework embeds a greedy heuristic to enhance the convergence speed as well as supporting the particles escape from the local optima. An evaluation method focused on the coverage inspection quality is presented that can be utilized as the benchmark for evaluation of other CPP methods where the inspection quality matters. The effectiveness of the proposed method is evaluated by comparing the proposed method with the state-of-art through multiple test cases. We also evaluate performance under different parameter configurations to explore the limitations of our proposed algorithm.

Our contributions in this paper are:

- Creation of a computationally efficient PSO co-optimization framework to compute the coverage path for the purpose of inspecting a scene/structure with arbitrary 3D topography using a camera-equipped multicopter UAV.
- Development of a novel cost metric that measures the quality-efficiency of a coverage inspection path, incorporation of this new metric into our co-optimization framework.
- Design of a set of metrics and benchmark method for evaluating the quality of the coverage path and a thorough evaluation of the proposed method through multiple test cases under various configuration parameters.

The paper is organized as follows. Section 2 presents work related to coverage path planning and the background of the PSO for path optimization. Section 3 introduces the principle steps of the proposed method. Section 4 presents the benchmark method for the coverage path assessment as well as the thorough evaluation of the proposed method. Section 5 summarizes our contributions and discusses future work.

2. BACKGROUND

In this section we first provide a brief summary of work related to coverage inspection planning, then discuss the theoretical background of the particle swarm optimization (PSO) as well as challenges applying it to robotic path optimization.

2.1 Coverage Inspection Planning

For purposes of structural inspection, current CPP methods can be classified as offline model-based or online sensor-based depending on whether the environment is given as prior information. Assuming the 3D geometry of the scene is known, model-based CPP uses geometrical information to design the optimal trajectory that covers the regions of interest within the scene for complete surface inspection. The sensor-based CPP, on the other hand, only relies on onboard sensors to explore the environment and greedily cover the scene based on an incrementally built map (Almadhoun, Taha, Seneviratne, Dias, & Cai, 2016). Because 3D geometries of civil structures are often known in advance (e.g. BIM, 3D CAD) we only focus on these in this paper.

In 2D planar workspaces, the majority of CPP methods utilize cell decomposition, such as the cellular decomposition (Acar, Choset, Rizzi, Atkar, & Hull, 2002; Choset & Pignon, 1998), or the grid-based method (Gonzalez, Alvarez, Diaz, Parra, & Bustacara, 2005; Zelinsky, Jarvis, Byrne, & Yuta, 1993) to formulate the free motion spaces. Then, chosen path such as the naïve zigzag path, the heuristic topological-based path (Lin & Goodrich, 2014; Wong & MacDonald, 2003), or the randomized approximate path (Danner & Kavraki, 2000) were executed within the free spaces to provide the full sweep of the plane. For 3D workspace coverage, there are a variety of modular (i.e., 2.5D) approaches that divide the 3D workspace into many 2D components and solve each component with 2D CPP methods. For example, Atkar, Choset, Rizzi, and Acar (2001) combined 3D topological segmentation with continuous back-and-forth patterns for uniform vehicle parts spraying. Cheng, Keller, and Kumar (2008) performed UAV-based urban inspection through a spiral trajectory with minimized coverage time and consideration of aerodynamics. Mansouri, Kanellakis, Wuthier, Fresk, and Nikolakopoulos (2016) fused a slicing algorithm with the modular approach and extended the coverage planning into a multi-agent schema for cooperative, aerial coverage inspection. In (Galceran et al., 2015), a 2.5D coverage algorithm was implemented with a real-time re-planning ability to enable autonomous underwater vehicle (AUV) bathymetric mapping in an uncertain environment. While these approaches have the advantage of fast computation and easy implementation, these regular 2D branch-and-bound and state-space based methods become computationally intractable in very large search areas.

Approximate algorithms were widely employed for 3D coverage problems because of their efficiency and the completeness. In Englot and Hover (2012b), a sampling-based CPP method was developed for inspecting 3D complex structures. The method asymptotically computed the optimal coverage path by iteratively solving two subproblems: (1) a coverage sampling problem (CSP) and (2) a multi-goal planning problem (MPP). Combined with SLAM-based navigation and control, this method has been used to perform ship hull inspection using a hovering autonomous underwater vehicle (HAUV) (Hover et al., 2012). To incorporate the differential constraints into the path generation, Papadopoulos, Kurniawati, and Patrikalakis (2013) proposed a random inspection tree algorithm (RITA) for non-holonomic robot inspection and planning in cluttered environments. Contrary to handling each subproblem individually, the algorithm simultaneously deals with the views and paths that guarantee global optimality of the designated paths. Ellefsen, Lepikson, and Albiez (2016) enabled the capability to move while sensing for sampling-based coverage inspection by simultaneously feeding the views and paths into an evolutionary optimization framework. The method also accepts imperfect coverage when the cost of 100% covering is sufficiently high. Recently, Bircher, Alexis, Burri, Oettershagen, et al. (2015) provided a new coverage path planning algorithm called structural inspection planner (SIP), with a novel view sampling strategy for aerial coverage inspection. Unlike other coverage planning methods that formulated the view planning into a submodular function, SIP samples the feasible camera views within a set of pre-defined search spaces based on the geometry of the model, and iteratively resamples these views through optimization. Our method is inspired by the viewpoint sampling strategy in SIP, however, our method provides for an additional degree-of-freedom for the onboard camera (i.e., gimbal pitch) which more efficiently provides full coverage inspection for complex structures. Moreover, the objective of our method is to compute a **quality-efficiency co-optimal coverage path** rather than only minimizing the flight distance as in SIP. Below we use the SIP algorithm to compare with and evaluate our co-optimization algorithm.

2.2 Particle Swarm Optimization

Particle swarm optimization (PSO) is a population-based stochastic optimization approach within the domain of swarm intelligence (Kennedy, 2011). The algorithm was developed by Kennedy and Eberhart in 1995, inspired by the social foraging behavior of some animals such as bird flocking and fish schooling. Comparing to the classic optimization method that requires a specific data structure, PSO is an evolutionary algorithm and can work with a wide range of optimization schemes. Compared to other evolutionary algorithms (e.g., genetic algorithm), PSO is fast and easy to implement. PSO starts by randomly distributing a certain number of individuals, called particles, in a multi-dimensional search space. Each particle in the search space is representative of a candidate solution in the problem space where the performance of each particle is evaluated by a fitness function. At each iteration, PSO adjusts the particles velocities and positions based on the local best position of each particle’s own records and the global best position of the population. The update function (Shi & Eberhart, 1998) of the particles’ velocities and positions is:

$$v_i \leftarrow \alpha v_i + \chi_i(0,\beta)\,(\mathcal{L}_i - x_i) + \chi_i(0,\beta)\,(\mathcal{G} - x_i), \tag{1}$$

$$x_i \leftarrow x_i + v_i, \tag{2}$$

where x_i is the ith particle's position, v_i is the velocity, $\mathcal{L}_i$ is the best position of particle i, and $\mathcal{G}$ is the best position in the population $(\mathcal{P})$. α is an inertia weight that controls the search behavior of particles in the search space. $\chi(\cdot)$ is a random number generator that preserves the particles' random motion capability to avoid premature convergence (Karatzas & Shreve, 1998). β is the acceleration coefficient that controls the speed of the particle moves towards the local/global best position. Over a number of iterations, particles gradually move towards better positions in the search space where the fitness value converges to one optimum, or several optima in the problem space. These steps are repeated until the termination condition is met, and the improved function results are obtained.

While there is research employing PSO in UAV inspection path planning (Foo, Knutzon, Kalivarapu, Oliver, & Winer, 2009; Phung, Quach, Dinh, & Ha, 2017), the motions of particles are confined to the discrete, two-dimensional search space. Computing the optimal coverage path directly in the 3D continuous space remains challenging because particles may get "stuck" in local optima without enforcing discretization or assuming a particular function shape (e.g., convex). The optimization framework proposed in this study overcomes this problem by (1) revising the particle update mechanism, and (2) introducing a greedy heuristic to enhance the particles exploration in the high dimensional search spaces. We describe this methodology next.

3. PROPOSED METHODOLOGY

The objective of our proposed co-optimal coverage path planning (CCPP) method (Figure 1) is to generate an aerial inspection path that co-optimizes visual coverage of an arbitrary 3D structure **and** the quality of inspection imagery. We assume the geometrical model of the scene at the region of interest (ROI) is given either through scanning (e.g., structure-from-motion), or modeled (e.g., building information modeling). The provided model is cleaned and coarsened in a pre-processing step to convert the model into a uniformly distributed surface mesh similar to related methods (Valette, Chassery, & Prost, 2008).

First our method finds the feasible paths that provide complete visual coverage of the scene. Specifically, for each path, we create a viewpoint configuration space at every mesh surface based on the camera and UAV constraints. We then iteratively sample one admissible camera view within each space. At last, the sampled views are connected through a shortest path.

Next, we feed a collection of the sampled paths into a PSO-based optimization framework for continuous path optimization. At the core of this method is the development of an objective function that integrates the quality and efficiency of a coverage path. We also provide a coefficient that allows users to control the optimized path based on the urgency of the task – prioritizing speed over quality. A greedy heuristic is embedded in this framework to efficiently increase the convergence speed as well as the overall optimization results.

Once the optimal path is found, the final step is to transfer the calculated path into a flight trajectory. In this study, we employ two path refinement steps: 1) using the rapidly exploring random tree (RRT) (LaValle, 1998) to avoid obstacles between two adjacent viewpoints, 2) applying the global B-spline curve interpolation (De Boor, De Boor, Mathématicien, De Boor, & De Boor, 1978) to smooth the path. Generally we have found that we prefer closeness to smoothness in the parameter configuration to guarantee that the trajectory reaches every designated viewpoint. The produced trajectory can then be safely executed for visual quality-based inspections of arbitrary 3D structures by a multicopter UAV.

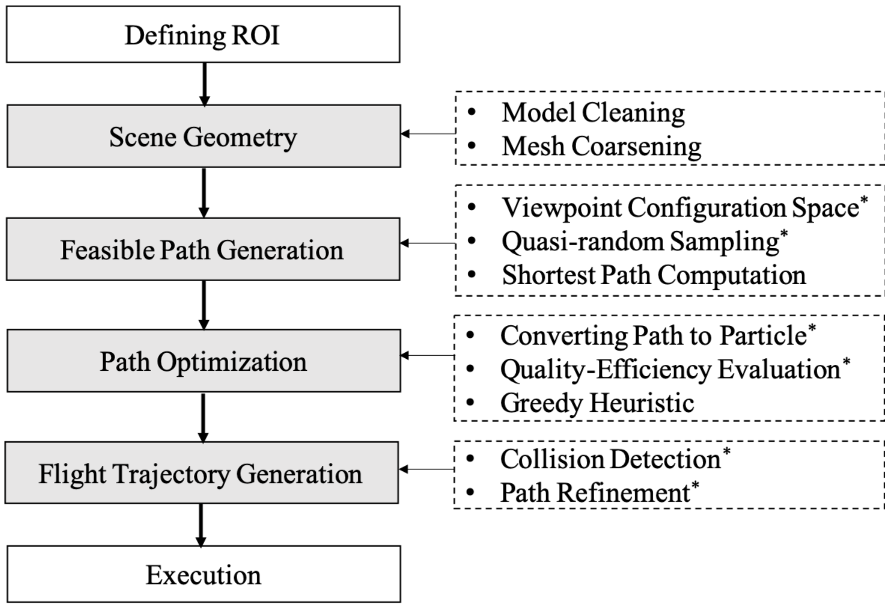


Figure 1 General workflow of the proposed method (detailed steps are presented in the dashed table, $*$ denotes the steps that can support parallel computing)

3.1 Feasible Path Generation

Here we introduce the viewpoint configuration space of each mesh surface, then discuss the strategy to find the admissible views and the feasible path.

3.1.1 Viewpoint Configuration Space

We pair each mesh surface (i.e., a triangular plane connected with other planes on the surface mesh with their common edges and corners) with a camera view that provides complete model coverage. The configuration space of each viewpoint is adapted from (Bircher et al., 2016) wherein a gimballed camera is used to satisfy remote sensing requirements. Specifically, the configuration space is defined by the required distance for acceptable spatial resolution ($\mathcal{D}_r$), the camera depth of view (DOV), safety concerns ($\mathcal{D}_{min}, \mathcal{D}_{max}$), the camera field of view (FOV), inspection requirement (η_{max}), and the gimbal motions ($\theta_{min}, \theta_{max}$). Mathematically, we let $\mathcal{q}$ be the set of admissible views that fully cover a triangular mesh model, Ω, and $\mathcal{q}_i$ denotes a viewpoint that observes the ith model surface Ω_i $(i = 1, \ldots, \mathcal{M})$ where $\mathcal{M}$ is the number of mesh surfaces. Thus, the configuration space of $\mathcal{q}_i$ is presented as:

$$\begin{gathered} d(\mathcal{q}_i, c_i) \leq \mathcal{D}_r, \\ \dot{d}(\mathcal{q}_i, \Omega_i) \in [\mathcal{D}_{min}, \mathcal{D}_{max}], \\ \varphi(\mathcal{q}_i, \Omega_i) \leq \frac{\pi}{2} - \eta_{max}, \end{gathered} \tag{3}$$

$$\theta(q_i, c_i) \in [\theta_{min}, \theta_{max}],$$

where $d(\cdot)$ denotes the Euclidean distance between two points, $\dot{d}(\cdot)$ is the orthogonal distance from a point to a surface, and c is the centroid of a triangular surface. $\varphi(\cdot)$ measures the incidence angle between the camera and the surface plane. The viewpoint orientation is computed as a ray casting from the camera principle point to the centroid of the triangular plane. This setup is intuitive for visual inspection applications where the target of inspection is often located at the center of the image. In addition, because the surfaces are uniformly distributed, our method encourages the model to be evenly inspected, thereby avoiding some points being over-observed and others being missed. These constraints formulate the viewpoint configuration space at each mesh surface (as shown in Fig. 2 (a)).

The viewpoint orientation is also restricted by the rotation range of the onboard gimbal system. Most existing aerial surveillance systems (Cheng et al., 2008; Geng et al., 2014) limit gimbal yaw, and therefore we likewise assume gimbal yaw between $[-\pi, \pi]$ and use the cone-shape camera model to simulate the camera FOV. Because this camera model is invariant to camera roll motions we lock the gimbal roll movement to 0 for computational simplicity, removing one degree-of-freedom without impacting flexibility. As a result, the camera orientations are only restricted by the gimbal pitch motions (as shown in Fig. 2 (b)).

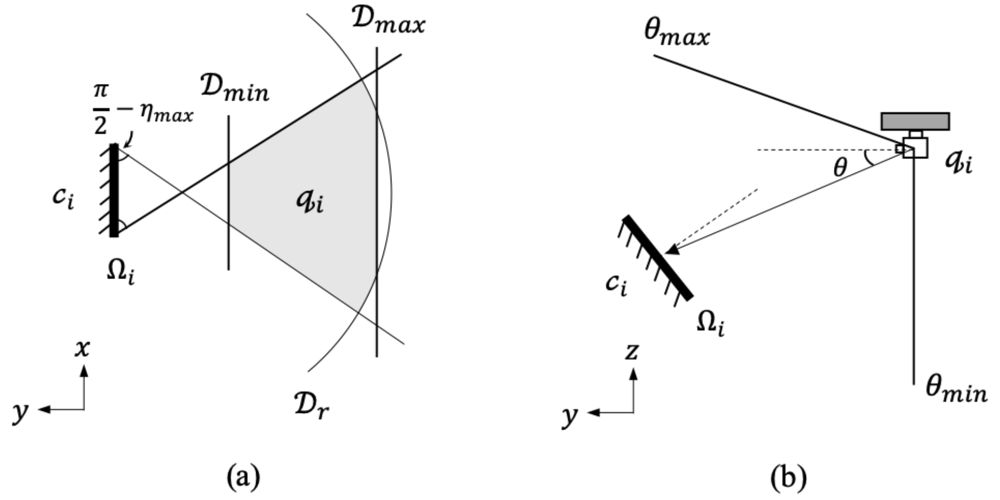


Figure 2 (a) 2D view of the configuration space (shaded region); (b) The camera ray and the gimbal pitch limits

3.1.2 Views and Path Generation

After the configuration spaces are determined, quasi-random sampling is employed to sample the admissible views at each surface. OBBTree (Gottschalk, Lin, & Manocha, 1996) is utilized at each sampled view to detect collisions and to perform the visibility test by casting a ray from the view to the surface centroid. The sampled views are then connected via a shortest path. In this study, we employ the LKH-TSP (Helsgaun, 2000) as the shortest path solver for the feasible path generation. Compared to other symmetric TSP solvers, LKH employs the k-opt heuristic that finds the shortest path by incrementally swapping the k (e.g. 2 or 3) pairs of sub-tours in the path. This strategy showed good trade-off between accuracy and efficiency for finding shortest paths. Such efficiency allows us to iteratively compute the shortest path, making our method computationally tractable even for large-scale scenes.

3.2 Cost Evaluation

We now introduce our objective function that evaluates feasible paths and guides path optimization. First we define the score for each viewpoint, then integrate it over the views to calculate the cost of the path. We define the score of each view as the weighted sum of two objectives: the *inspection quality* and the *inspection efficiency*. We utilize the weighted sum because the goal of this study is to efficiently design a single executable coverage path (i.e., a prior articulation of preference) rather than yielding a complete Pareto optimal set of a multi-objective problem (MOP) (i.e., a posterior articulation of preference). Eq. 4 and 5 present the proposed cost function of a path and each view within the path:

$$\mathcal{F} = \frac{\sum_{i=1}^{\mathcal{M}} \mathcal{F}_i}{\mathcal{M}}, \tag{4}$$

$$\mathcal{F}_i = (1 - \varepsilon)\mathcal{Q}_i + \varepsilon E_i, \tag{5}$$

where $\mathcal{F}_i$ is the score of the ith camera view and $\mathcal{F}$ is the cost of a coverage path. $\mathcal{Q}$ and E respectively represents the metrics utilized for measuring the inspection quality and the inspection efficiency contributed by each designated view. ε ($\varepsilon \in [0,1]$) is a user selected input weight coefficient that trades quality for efficiency to speed up performance. In order for the weight to reflect the prior articulation of preference more accurately, function transformation is performed before the weighted summation so that either objective does not naturally dominate the other. We now discuss the details of each objective.

3.2.1 Inspection Quality

For image-based aerial inspection, images taken with an onboard camera at some locations are more informative than the others due to lighting, glare, white balance, incidence angle, etc. (Ham, Han, Lin, & Golparvar-Fard, 2016). Only one camera view is allowed for each surface plane so it is imperative to identify which viewpoint provides the most information. We develop a confidence measure, inspired by (Shen, Zhang, & Fels, 2007), to measure the view quality based on two terms: (1) the view-to-surface distortion and (2) the view-to-surface resolution.

View-to-surface distortion denotes the level of distorted effects of the captured images. We use the view angle between the ray shooting from camera center and the norm at the triangular planar centroid to demonstrate the distortion (as shown in Fig. 4 (a)). The closer the view angle is to the norm, the less distortion is capture in the image. As a result, we measure the view-to-surface distortion as η/η_{max} where η is the computed observation angle, and η_{max} is the maximal acceptable observation angle.

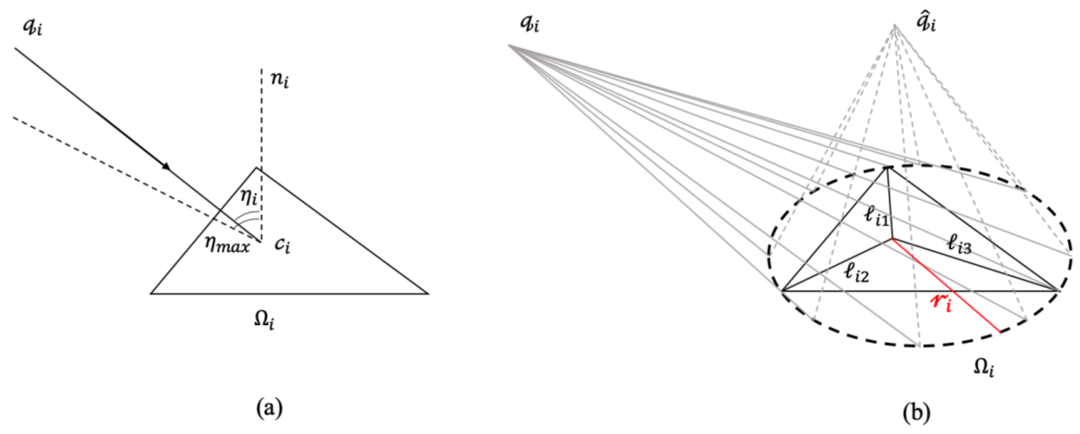


Figure 3 (a) View-to-surface distortion at viewpoint q_i: measured as the ratio between η and η_{max}; (b) Viewpoint distance at viewpoint q_i: measured as the normalized difference between the radius (r) of the projected viewpoint ($\hat{q}_i$) and the longest medium $\ell_{i\cdot max}$ at surface plane Ω_i.

View-to-surface resolution defines how clearly the camera sees the features on the target surface. Images captured too far from the object have insufficient spatial resolution, while images captured too close may not cover the entire surface plane and fail to guarantee the complete coverage. This is handled by finding the optimal camera position where the projected image best fits the surface plane. As shown in Fig. 4 (b), the optimal view-to-surface resolution is identified when the radius of the projected cone-shape camera model approaches the longest medium (i.e., the longest distance from the centroid to the triangle corner) of the triangular surface. In this model, the longest medium is chosen so that a larger penalty is given to viewpoints that do not fully cover the plane.

Based on the above definitions, we formulate inspection quality of each camera in Eq. 6 below:

$$Q_i = s_i \cap \delta\left(\frac{\eta_i}{\eta_{max}}, 1 - \left|\frac{r_i - \ell_{i\cdot max}}{r_{max} - \ell_{i\cdot max}}\right|\right), \tag{6}$$

where $\ell_{i\cdot max}$ is the medium with the maximum distance at Ω_i, r_i is radius of the projected image plane at the transformed camera view, $\hat{q}_i$, from q_i, and r_{max} is the maximal projected radius defined by the camera's intrinsic parameter and the upper bound of the search space (Eq. 3). s_i is a binary term that measures if Ω_i is fully visible from q_i, and $\delta(\cdot)$ fuses the two terms. In this study, we use the mean as the default function for the integration.

3.2.2 *Inspection Efficiency*

Inspection efficiency measures the traveling cost of the designated path. For a coverage path, the distance of the entire path is equal to the sum of the distance at each view segment. Thus, we compute the path efficiency at view q_i as the average of the connecting distance from the preceding view to the current view and from the current view to the successive view:

$$e_i = \mathcal{C} \cdot \frac{d(q_{i-1}, q_i) + d(q_i, q_{i+1})}{2}, \tag{7}$$

where $\mathcal{C}$ is a scaling vector. By assigning different values to $\mathcal{C}$, e can be used to measure other inflight costs such as energy consumption, time-of-travel, risk level, etc. Note that e_i is an unbounded distance function that has to be normalized in order to be aggregated with the normalized quality metric. Thus, we provide the function-transformation that rescales the distance function e and generates the proposed efficiency metric (as shown in Eq. 8):

$$E_i = \frac{e_i - \gamma {e_i}^*}{(1-\gamma){e_i}^*}, \tag{8}$$

where ${e_i}^*$ is the base distance at view $\mathcal{q}_i$. We use the distance computed at the initial iteration (i.e., random sampling) to produce the base cost of each view segment. γ is a reduction factor that measures the magnitude of distance reduction. Based on the empirical analysis, γ in most cases is within the range of $[0.2, 0.4]$. In the following studies, we let $\gamma = 0.2$ for all test cases.

3.3 Path Optimization

After feasible paths are generated and the path evaluation method is provided, the next step is to optimize the paths. To inspire our design, imagine there exists one optimal path, where each camera pose along the path is admissible within the viewpoint configuration space. If we know the camera pose at each configuration space, we can find the corresponding path. Thus, we can convert the problem of finding the optimal path to compute the optimal camera poses. Because the camera configuration spaces are known (Section 3.1), we can sample a set of admissible camera views to explore the search space, and greedily compute the best camera pose within each space using PSO (Trelea, 2003). However, those camera views may not lead to the optimal path because the path is also determined by the spatial relationship between the camera poses. Our PSO-based optimization framework overcomes this limitation by both optimizing the camera views and the path. In the following paragraphs, we discuss the proposed optimization framework in detail.

We start by defining each particle $\mathcal{p}$ as a feasible coverage path (as presented in Eq. 9) which is composed of a set of admissible views and a shortest path index:

$$\mathcal{p} = \{\mathcal{q}_i, \tau \mid i = 1,2,3,\dots,\mathcal{M}\}, \tag{9}$$

where τ is the shortest path of the view set $\mathcal{q}$. We repeat the sampling-based strategy to compute a population of particles. Theoretically, the shortest path should be computed for each particle. However, assuming the particles, which are guided by the cost function, would gradually converge to the optimal path, the TSP results would also converge. Thus, we could just compute a TSP for one particle at each iteration. To avoid the TSP getting trapped by a 'bad' particle, we randomly select the particle at each iteration. The computed TSP result is then utilized to evaluate all particles in the population:

$$\mathcal{P} = \left\{\mathcal{p}_j, \tau_\rho \mid j = 1,2,3,\dots,\mathcal{N}\right\}, \tag{10}$$

where $\mathcal{P}$ is a population of particles, τ_ρ is the shortest path index of the randomly selected seed with $\rho = rand(1, \mathcal{N})$. The strategy significantly reduces the processing workload as the computational complexity of the proposed method is dominated by the TSP (detailed in Section 4.4.3).

We now revise the existing PSO update mechanism (as in Eq. 1 and Eq. 2) so that it can handle a list of camera views (shown in Eq. 11 and Eq. 12):

$$\mathcal{v}_{ik} \leftarrow \alpha \mathcal{v}_{ik} + \chi_{ik}(0,\beta)\,(\mathcal{L}_{ik} - \mathcal{q}_{ik}) + \chi_{ik}(0,\beta)\,(\mathcal{G} - \mathcal{q}_{ik}) \tag{11}$$

$$\mathcal{q}_i \leftarrow pose(\mathcal{q}_i, \mathcal{v}_i), \tag{12}$$

where $k = 1{,}2{,}3$ denotes the translations (i.e., x, y, z) of each view in the world coordinate, and the $pose(\cdot)$ computes the new 6d pose (i.e., roll, pitch, yaw) based on the existing viewpoint position and the updated velocity. Note that each newly updated viewpoint pose must still stay within the configuration space to guarantee its validity. Views outside of the feasible spaces are rejected. For particles representing rejected views, we re-compute the update function such that a fresh view is generated. However, this method become ineffective as the optimization converges where the camera search spaces become small. To overcome this problem, a decaying function is developed to support the adaptive viewpoints update:

$$\mathcal{v}_i \leftarrow (\varepsilon^{\mu})\mathcal{v}_i, \tag{13}$$

where ε is the decay coefficient, and μ is the number of attempts to update each view. We empirically set ε equals to 0.5 and let $\mu \leq 6$ to limit the number of attempts. Compared to linear decay, this exponential function takes fewer attempts and performs better by continuously exploiting the search spaces.

3.4 Greedy Heuristic

As presented in Eq. 1, each particle is updated through its own best record so far and the global best record in the population. This mechanism assumes the particles are sufficient to visit the entire search space, and the true optimal solution can be visited by at least one particle through iterations (Trelea, 2003). However, in the proposed framework, each particle denotes a coverage path that encapsulates a set of 6d camera poses with the motion of each camera restricted in specific 3d configuration spaces. The current PSO update mechanism is insufficient to explore those search spaces thoroughly, resulting in premature convergence. We seek a method to overcome this limitation.

Inspired by the crossover technique in the evolutionary algorithm, we introduce an effective and efficient greedy heuristic to avoid particles getting trapped in local optima. The basic idea is that we can enhance the particles exploration capability by recursively updating the global best particle with the suboptimal views. The pseudocode of the proposed heuristic is presented in Algorithm 1. Specifically, for each mesh surface Ω_i, we find the camera view that has the minimal cost $\tilde{\mathcal{q}}_i$ in the population, and store the poses of the connected views $\langle \mathcal{q}_{i-1}, \mathcal{q}_{i+1} \rangle$. Then, we re-compute the new costs of the view that links the stored connecting views, select the view $\tilde{q}_i{}'$ with the minimum cost and utilize it to update the views in the global best particle. We iterate this two-step process through all mesh surfaces to update the global best particle $\mathcal{G}$. The updated $\mathcal{G}$ is then utilized to support the new particle generation in the next iteration. Fig. 4 shows a graphic illustration of the proposed heuristic

Algorithm 1 Pseudocode of the greedy heuristic

```
Require: $\mathcal{P}$
Ensure: $\mathcal{G} \neq \emptyset$
1: for $i = 1$ to $\mathcal{M}$ do:
2:     find $\tilde{q}_i = \underset{j}{\operatorname{argmin}} \mathcal{F}_i$
3:     store $\langle q_{i-1}, q_{i+1} \rangle$ that connects $\tilde{q}_i$
4:     compute the new cost at $i$ using $\langle q_{i-1}, q_{i+1} \rangle$ as $\mathcal{F}_i{}'$
5:     find $\tilde{q}_i{}' = \underset{j}{\operatorname{argmin}} \mathcal{F}_i{}'$
6:     if $\tilde{q}_i{}' < \tilde{q}_i$ do
7:         $\mathcal{G}(i) = \tilde{q}_i{}'$
8:     end if
9: end for
```

Compared to exhaustive searching (i.e., iteratively finding the minimal cost view at each index) requiring $\mathcal{O}(\mathcal{M}\mathcal{N}^3)$ complexity to compute the best view combinations through the entire population, the proposed greedy heuristic is a two-step linear search, which only takes $\mathcal{O}(\mathcal{M}\mathcal{N})$, making the optimization tractable even for large-scale scenes.

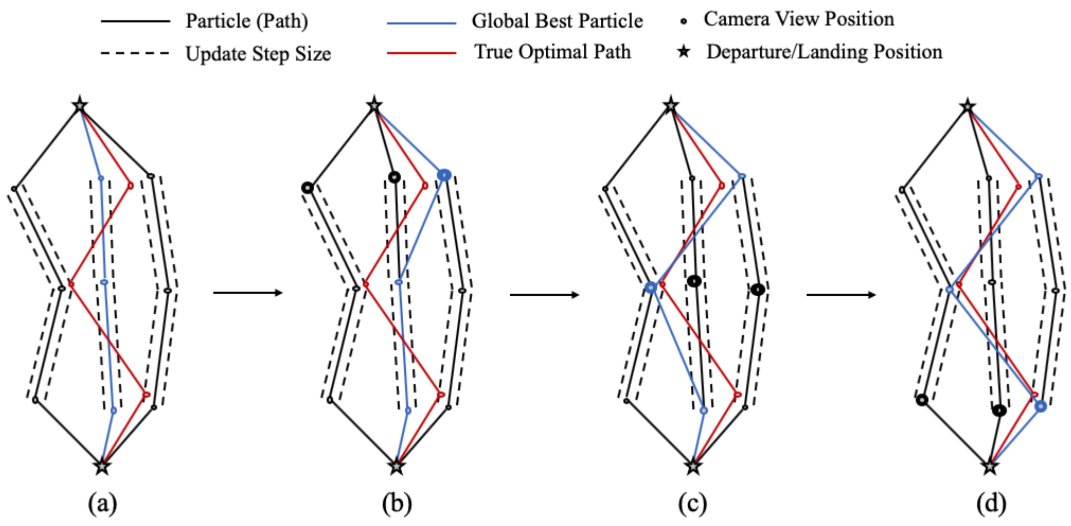


Figure 4 An example of the proposed greedy heuristic: (a) Before the update mechanism, the global best particle $\mathcal{G}$ (blue solid line) is selected from the whole population and is stuck in the

local optima (dashed line); (b-d) iteratively updates $\mathcal{G}$ by updating the camera view through the two-step greedy heuristic; (d) The resulting $\mathcal{G}$ is much closer to the true optimal solution compared to the $\mathcal{G}$ chosen from the population.

4. SIMULATION AND EVALUATION

4.1 Experimental Setup

We test the proposed CCPP against the state-of-art SIP (Bircher, Alexis, Burri, Oettershagen, et al., 2015) on three different target geometries: (1) a 2D-planar scene: the *solarPlant*, (2) a simple, small-scale 3D object: the *hoaHakanaia status*, and (3) a geometrically complicated, large-scale 3D target: the *church*. The first two models are downloaded from the SIP open-source dataset (Bircher, Alexis, Burri, Kamel, et al., 2015) without post-processing. The third model is reconstructed through a set of aerial images we crop to the interested structure (i.e., church) and downsample the model to contain 200 triangles based on the required level of details (LoD) for aerial inspection. These three geometric models were chosen to evaluate the method on objects with different geometries. For UAV safety, we set the minimal height for all three models as 3 meters above ground. Table 1 shows the graphic and parameters of the selected models.

Table 1 Selected test models and the parameters

Model			
Name	solarPlant	hoaHakanaia	church
Triangular Planes	278	225	200
Dimensions	$86.7 \times 56.3 \times 2.4$	$8.4 \times 5.2 \times 19.5$	$63.7 \times 62.4 \times 19.4$
Geometry	2D planar	3D small	3D large

Due to the different constraints used in designing the camera search spaces (i.e., camera direction and gimbal control), the optimal solution of these two methods (i.e., SIP and CCPP) may require distinct search spaces. There is a need to provide the uniform configuration spaces so that both methods can provide the optimal solution. Thus, we use the default parameters as in (Bircher, Alexis, Burri, Kamel, et al., 2015) but amplify the safe distance and minimal observation angle such that both methods have sufficient space for the optimal path generation. The details of parameters used in the comparison study are presented in Table 2∼4.

Table 2 Parameters for the generation of the configuration space of *solarPlant* (distance unit in meters)

Method	$[\mathcal{D}_{min}, \mathcal{D}_{max}]$	η_{max}	FOV	$\mathcal{D}_r$	$[\theta_{min}, \theta_{max}]$
SIP	$[5, 10]$	60°	$[120^\circ, 90^\circ]$	\	-25°
CCPP				15	$[-90^\circ, 30^\circ]$

Table 3 Parameters for the generation of the configuration space of *hoaHakanaia* (distance unit in meters)

Method	$[\mathcal{D}_{min}, \mathcal{D}_{max}]$	η_{max}	FOV	$\mathcal{D}_r$	$[\theta_{min}, \theta_{max}]$
SIP	$[4, 12]$	65°	$[108^\circ, 92^\circ]$	\	-25°
CCPP				20	$[-90^\circ, 30^\circ]$

Table 4 Parameters for the generation of the configuration space of *church* (distance unit in meters)

Method	$[\mathcal{D}_{min}, \mathcal{D}_{max}]$	η_{max}	FOV	$\mathcal{D}_r$	$[\theta_{min}, \theta_{max}]$
SIP	$[6, 35]$	70°	$[108^\circ, 92^\circ]$	\	-25°
CCPP				35	$[-90^\circ, 30^\circ]$

4.2 Evaluation Methodology

To have a comprehensive evaluation of the proposed method, we individually compare the efficiency and the quality of the generated paths. For the efficiency evaluation, we use the Euclidean distance as the primary indicator to compute the efficiency of the designated paths (i.e., $\mathcal{C}$ is the unit vector). Because SIP doesn't employ the motion dynamics for multicopter UAVs we bypass the flight controller implementation and assume perfect motion by the UAV in the comparison study.

For the quality evaluation, our goal is to evaluate how good the designated cameras observe the models. We first evaluate the quality of the designed views observed at each point ($\mathscr{k}$) on the model surface, then compute the mean quality over all surface points ($\mathcal{K}$) to evaluate the camera set. We employ the anisotropic model developed in (Wang, Qi, Shi, & Wang, 2013) that jointly consider the occlusion, focus, resolution and geometrical distortion aspects as the strategy for our quality evaluation. However, this model is sensitive to camera intrinsic and extrinsic parameters which is not effective at evaluating the inspection quality when different camera models or inspecting criteria is utilized. Thus, we revise this model to represent our proposed scenarios more closely. Specifically, we convert each measurement into the normalized forms before the product summation. As shown in Eq. 14, the quality $\mathscr{h}$ of a coverage path $\mathscr{q}$ is the average quality of all surface points contributed by each camera in the path. Unlike (Wang et al., 2013) that recommends computing the quality of each surface point as the sum of all visible cameras, our metric only selects the single best camera from the path because coverage inspection prefers a single best view instead of a set of mediocre views. The metrics are:

$$\mathscr{h}(\mathscr{q}) = \frac{1}{\mathcal{K}} \sum_{\mathscr{k} \in \mathcal{K}} \max_i \left[\mathscr{w}_o(\mathscr{q}_i, \mathscr{k}) \mathscr{w}_r(\mathscr{q}_i, \mathscr{k}) \mathscr{w}_g(\mathscr{q}_i, \mathscr{k}) \right] \tag{14}$$

$$\mathscr{w}_o(\mathscr{q}_i, \mathscr{k}) = \begin{cases} 1, & if\ visible \\ 0, & otherwise \end{cases}$$

$$\mathscr{w}_r(\mathscr{q}_i, \mathscr{k}) = \exp\left(-\frac{1}{\sigma_r} \cdot \left(\frac{\dot{d}(\mathscr{q}_i, \mathscr{k}) - \rho}{\mathcal{D}_{max}} \right)^2 \right)$$

$$\mathcal{w}_g(q_i, \mathcal{k}) = \exp\left(-\frac{1}{\sigma_g}\cdot\left(\frac{\eta(q_i,\mathcal{k})}{\eta_{max}}\right)^3\right),$$

where $\mathcal{k}$ is a sampled point on the model surface, $\mathcal{w}_o(\cdot)$, $\mathcal{w}_r(\cdot)$, $\mathcal{w}_g(\cdot)$ are the revised occlusion, resolution and distortion component. Specifically, $\mathcal{w}_o(\mathcal{p}_i, \mathcal{k})$ measures the visibility of a surface point $\mathcal{k}$ from a camera $\mathcal{p}_i$. $\mathcal{w}_o = 1$ when $\mathcal{k}$ is within the camera frustum and there is no occlusion. $\mathcal{w}_r(\mathcal{p}_i, \mathcal{k})$ is determined by the normalized orthogonal distance between the camera and the point, and $\mathcal{w}_g(\mathcal{p}_i, \mathcal{k})$ indicates the normalized observation angle. σ_r and σ_g are the decay coefficients for the effects of resolution and distortion. We empirically set $\sigma_r = 0.15$ and $\sigma_g = 0.2$ through all the experiments in this study.

Because the surface model is uniformly distributed we provide the uniform sampling at each surface plane to achieve the model surface sampling. For each camera view, a camera frustum is created to simulate the 'snapshot' at each designated view. The width and length of the frustum is defined based on the camera FOV. We set the far plane of the frustum as $\mathcal{D}_r$, and the near plane as a small number ρ ($\rho \approx 0.1$). Fig. 5 is a visualization of the observation quality of the church model contributed by a single camera view. The color bar illustrates the quality of each surface point. Based on the evaluation strategy, the quality of inspection of each coverage path can be assessed both quantitatively (through the metric) and qualitatively (through the color-coded model).

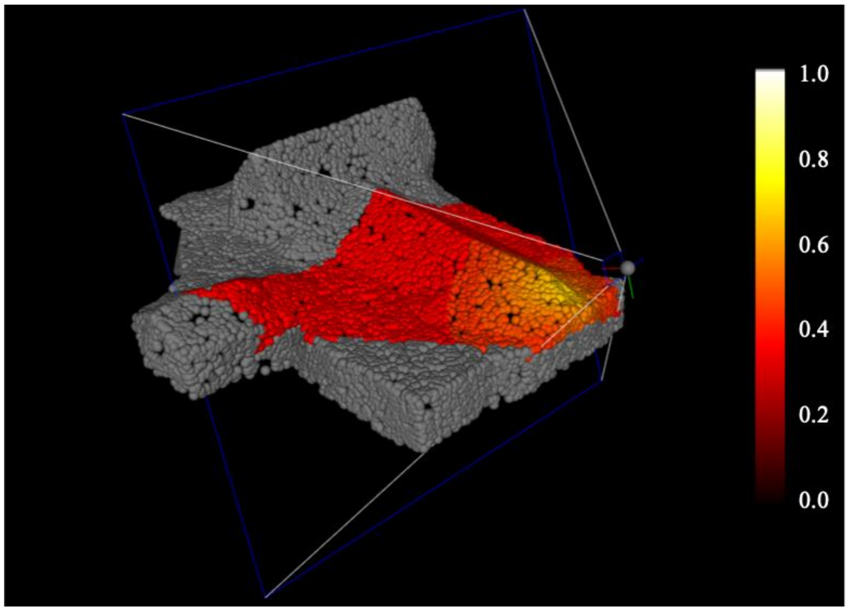
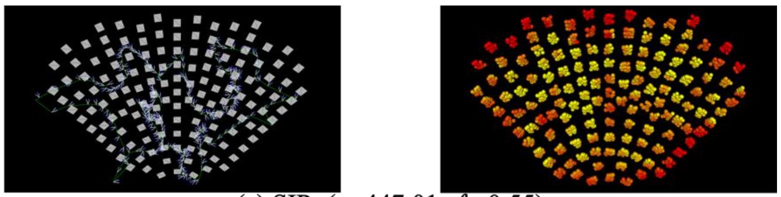
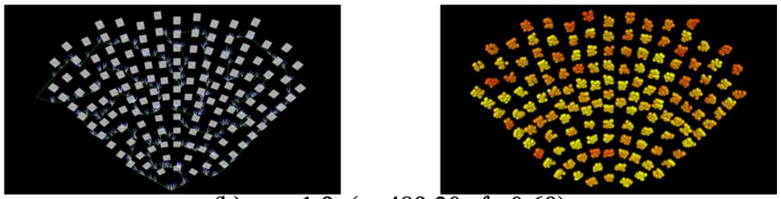


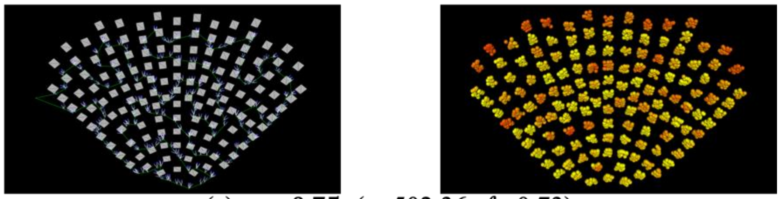
Figure 5 Visualization of the quality of each sampled point on the model when a single camera exists. The color bar colorizes the value of $\mathcal{h}$ at each point. When $\mathcal{h}$ is close to 1, the points become white; while when $\mathcal{h}$ is close to 0, the points become black. Surface points not visible by the camera are gray.

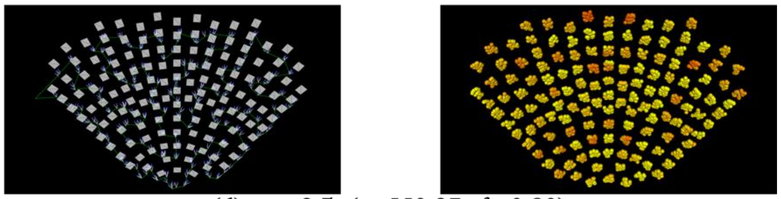
4.3 Results

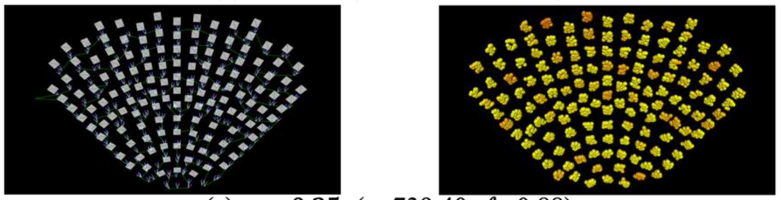
The comparison results for the three test scenes are shown in Figs. 6 to 8. The left column of the figures shows the designated views and paths using the SIP (Figs. 6.a, 7.a, and 8.a) and the proposed CCPP (Figs. 6.b-f, 7.b-f, and 8.b-f) under different ε (i.e., the user selected tradeoff parameter), while the right column shows the color-coded quality inspection results. For all the three cases, we observe that while SIP still provides the shortest distance, it also produces the worst inspection quality results compared to the CCPP under all conditions. This result shows that CCPP is able to provide a baseline quality for coverage inspection with a small sacrifice in efficiency. From the color-coded point cloud model, we observe that the boundary of the inspection geometry (i.e., boundary of solarPlant, bottom of hoaHakanaia and church), and the geometric concavities regions (i.e., middle of hoaHakanaia and inner corners of church) are easily missed by the SIP. With high ε favoring efficiency over quality, CCPP still shows significant quality improvement at each region, although they are more weakly observed due to the choice of ε (Figs. 6.b, 7.b, and 8.b). As ε decreases, the inspection parameters are more heavily weighted in the optimization resulting in an increase in observation quality (Figs. 6.e, 7.e, and 8.e). When $\varepsilon = 0$ (Figs. 6.f, 7.f, and 8.f), only inspection quality is considered in the optimization and the cameras are converged at the orthogonal direction at each surface plane maximizing the quality of the inspection. We observe that compared to the solarPlant and the hoaHakanaia model where the inspection quality is ~1.0 (orthogonal view and closed distance), the observation quality of the church can only reach 0.89. We believe this is caused by the geometrical complexity of the model and the required safety constraints.


(a) SIP: (e: 447.01, $\hbar$: 0.55)


(b) $\varepsilon = 1.0$: (e: 489.20, $\hbar$: 0.69)


(c) $\varepsilon = 0.75$: (e: 502.36, $\hbar$: 0.73)


(d) $\varepsilon = 0.5$: (e: 550.97, $\hbar$: 0.80)


(e) $\varepsilon = 0.25$: (e: 739.40, $\hbar$: 0.88)

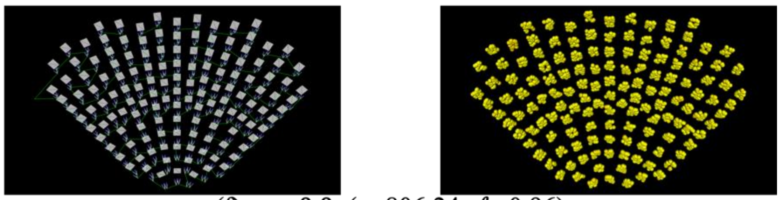
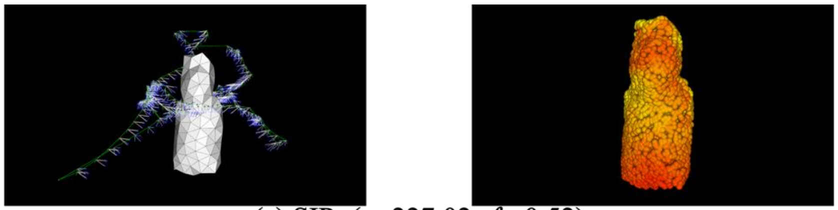
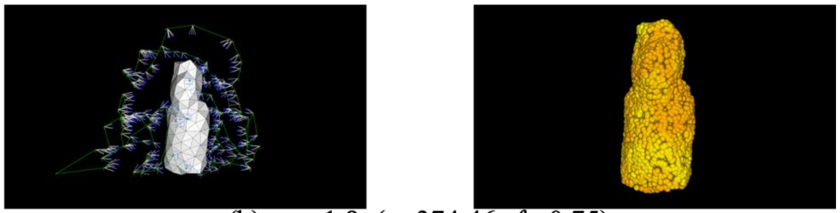
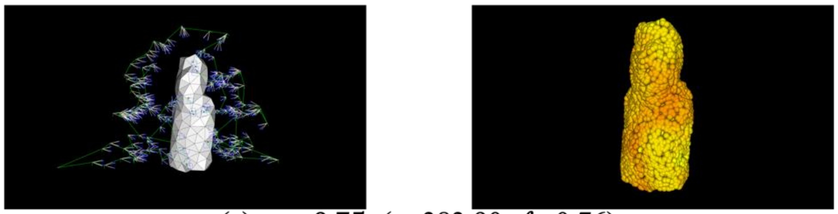
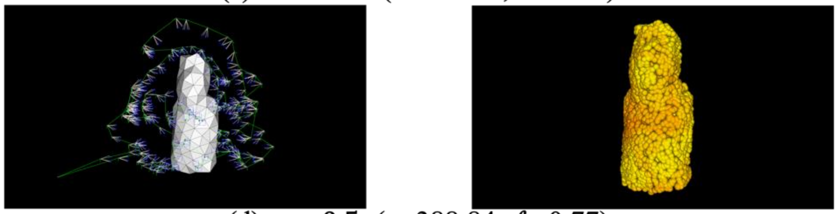

(f) $\varepsilon = 0.0$: (e: 806.24, $\hbar$: 0.96)

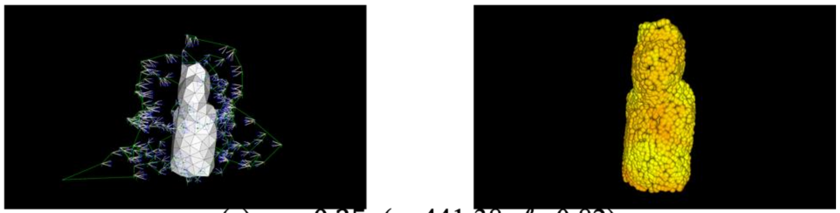
Figure 6 Comparing paths (e) and qualities $(\hbar)$ of *solarPlant*: (a): SIP; (b)-(f): CCPP with decreasing ε


(a) SIP: (e: 227.02, $\hbar$: 0.52)


(b) $\varepsilon = 1.0$: (e: 374.46, $\hbar$: 0.75)


(c) $\varepsilon = 0.75$: (e: 383.90, $\hbar$: 0.76)


(d) $\varepsilon = 0.5$: (e: 388.84, $\hbar$: 0.77)


(e) $\varepsilon = 0.25$: (e: 441.38, $\hbar$: 0.82)

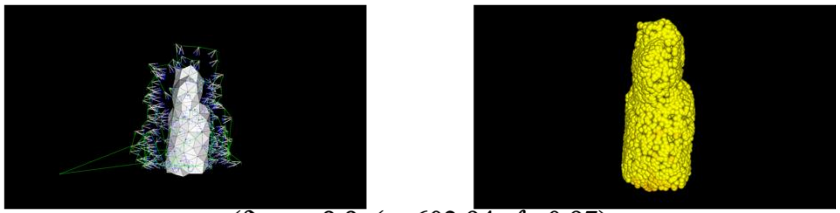
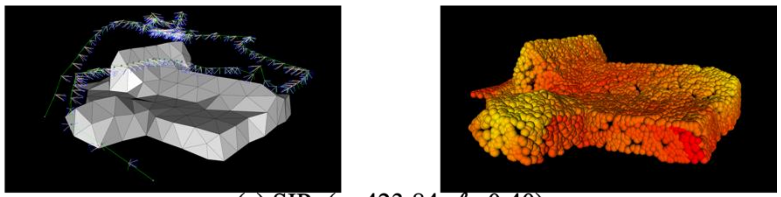
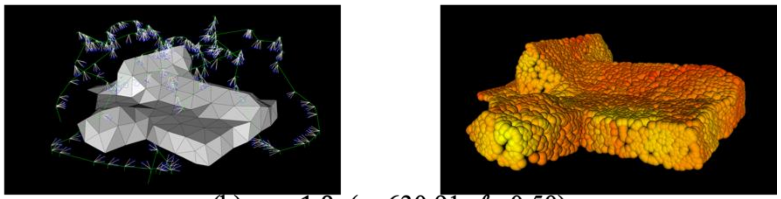
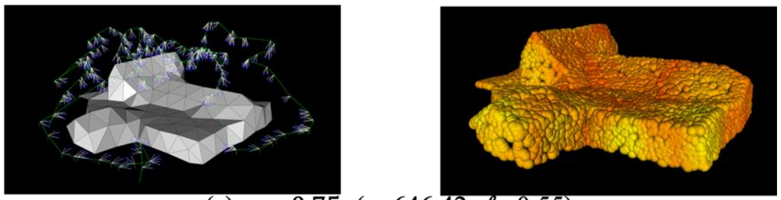
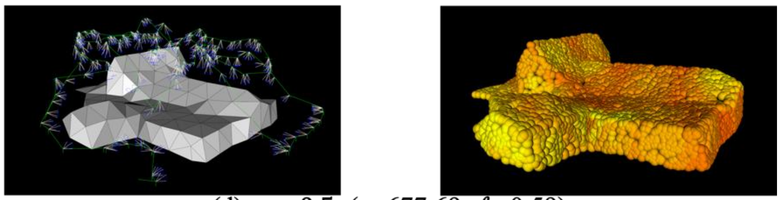

(f) $\varepsilon = 0.0$: (e: 602.94, $\hbar$: 0.97)

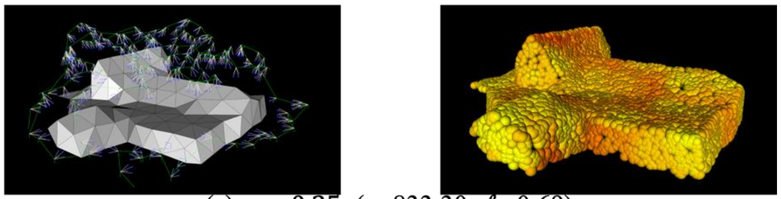
Figure 7 Comparing paths (e) and qualities $(\hbar)$ of *hoaHakanaia*: (a): SIP; (b)-(f): CCPP with decreasing ε


(a) SIP: (e: 423.84, $\hbar$: 0.40)


(b) $\varepsilon = 1.0$: (e: 630.91, $\hbar$: 0.50)


(c) $\varepsilon = 0.75$: (e: 646.42, $\hbar$: 0.55)


(d) $\varepsilon = 0.5$: (e: 677.69, $\hbar$: 0.59)


(e) $\varepsilon = 0.25$: (e: 833.30, $\hbar$: 0.69)

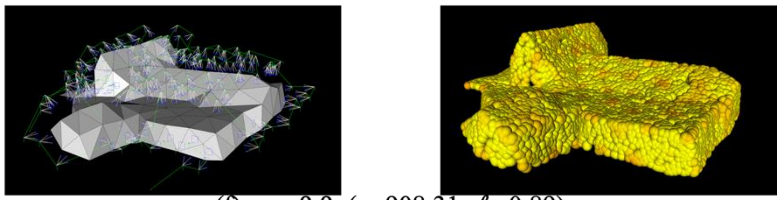

(f) $\varepsilon = 0.0$: (e: 908.31, $\hbar$: 0.89)

Figure 8 Comparing paths (e) and qualities $(\hbar)$ of the church model: (a): SIP; (b)-(f): CCPP with decreasing ε

We further explore the relationship between the inspection quality and efficiency with varying user input parameter, ε, by performing multiple task runs. As shown in Fig. 9, the quality and the efficiency of CCPP decrease/increase as ε increases. We use the rescaled SIP dashed lines as both the lower quality bound and the upper efficiency bound for illustration purposes. We observe that the efficiency and quality are not linearly reflected by the changes in ε and present different patterns with different target geometries. This may be caused by the fact that the weighted sum is a only linear approximation of the multi-objective problem, and each user-defined weight is an approximated preference of the relative magnitude of each objective function, not the function value (Marler & Arora, 2010). Finding optimal weights will be an important part of future work as we seek to fully automate the path planning process.

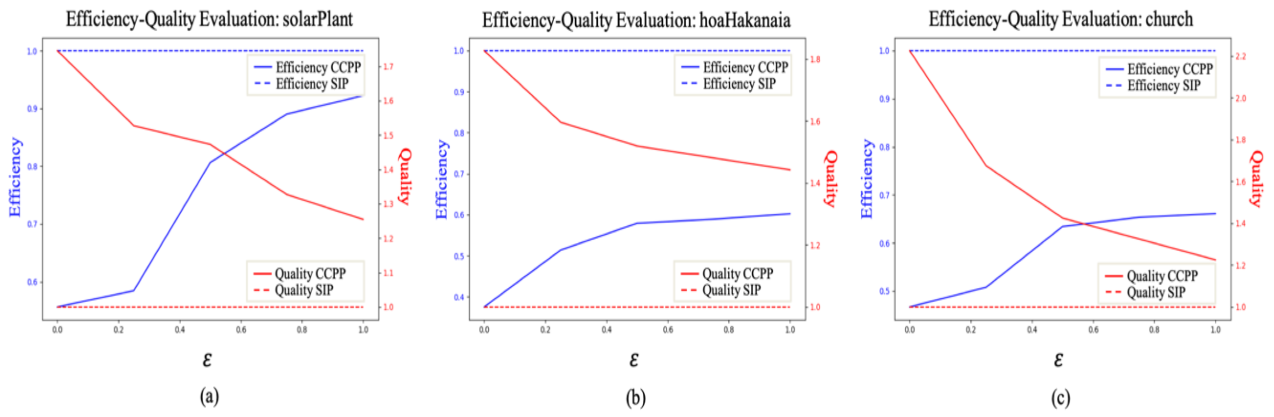


Figure 9 Efficiency vs. quality evaluation under different values of ε (a): solarPlant; (b) hoaHakanaia; (c) church.

Finally, we convert the co-optimal paths into flight trajectories by imposing a path refinement algorithm. Figure 10 shows the computed quality-efficiency balanced trajectory ($\varepsilon = 0.5$) of the three test scenes in the virtual environment. This trajectory can be tightly followed by a multicopter UAV with an appropriate autopilot control system (Mellinger, Michael, & Kumar, 2012).

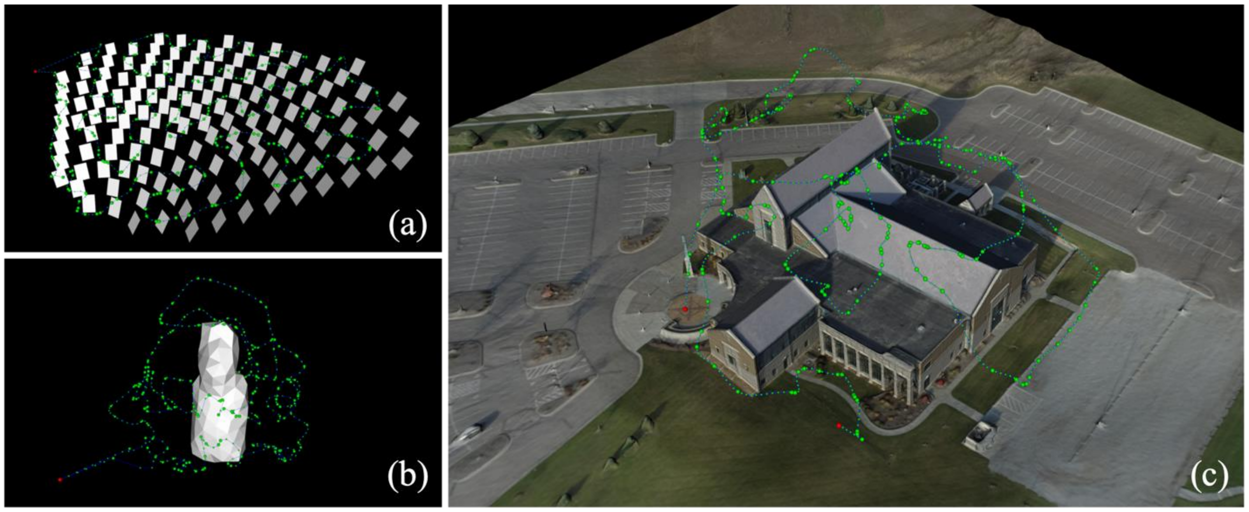


Figure 10 Optimal flight trajectory (blue) of the three scenes: (a) solarPlant, (b) hoaHakanaia, (c) church. The designed viewpoints are shown as large dots (in green) and the interpolated waypoints of the smoothed path are shown as the small dots (in green). The red dots denote the take-off and landing positions of the UAV.

4.4 Evaluation

In this section, a thorough evaluation of the proposed method is provided. First, we evaluate the convergence of the optimization framework for the three test cases and analyze the effects of the PSO parameters as well as the greedy heuristic on the optimization results. Second, we study the effects of the gimbal parameters on the optimization results. Finally, the runtime efficiency of the proposed method is discussed.

4.4.1 Optimization Performance

Table 5 presents the comparison of the optimization performance between the two methods. To make both methods comparable, we first set $\varepsilon = 1.0$ in the CCPP such that the distance is the only objective of optimization and does not consider inspection quality at all. We observe that due to the different camera search space, our method imposes longer distance at both the initial and last iteration compared to the SIP. We use the convergence rate (i.e., optimized distance divided by the initial distance) as the indicator to measure the optimization performance that minimizes such effect. The results showed that while SIP slightly outperforms the CCPP all three cases, the rates of convergence are comparable, especially for the *solarPlant* and the *church* cases, which validates the effectiveness of the proposed optimization framework.

Table 5 Comparison of the optimization performance in distance between SIP and CCPP

	Initial Distance		Optimized Distance		Convergence Rate	
Model	SIP	CCPP	SIP	CCPP	SIP	CCPP
solarPlant	1031.77	1108.42	447.01	484.70	0.43	0.44
hoaHakanaia	482.06	706.49	227.02	376.97	0.47	0.53
church	1190.54	1780.46	423.84	641.32	0.36	0.36

In Fig. 11, we evaluate the effects of the proposed greedy heuristic on all the three test cases. The result verifies our assumption that the particles are easily trapped in local optima without the proposed heuristic. In Fig. 12, we further assess the performance of the CCPP under different PSO population size and iteration numbers. The results show that both population size and iteration number provide positive effects on the optimization results, though their effects vary depending on the geometry being inspected. Generally, we obtain strong results with a population size and iteration number both set at 30.

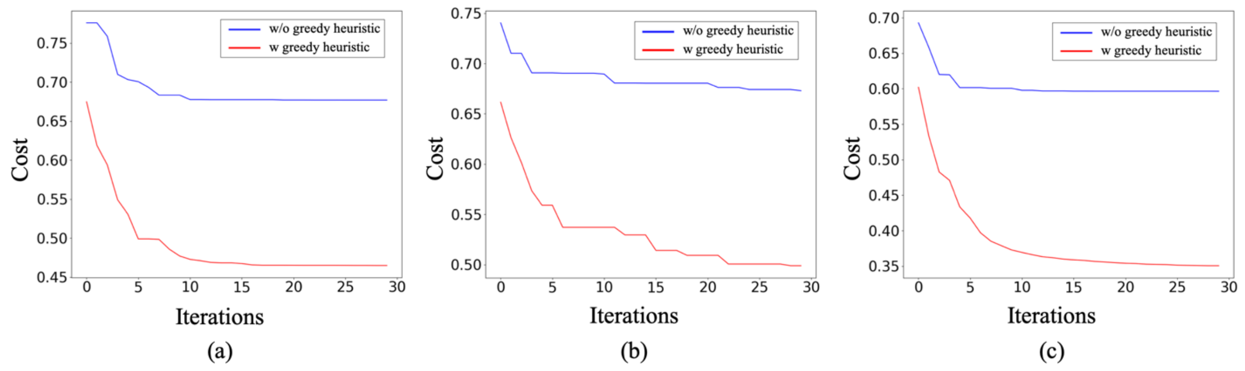


Figure 11 Comparison of the optimizations with and without using the greedy heuristic: (a) solarPlant; (b) hoaHakanaia; (c) church

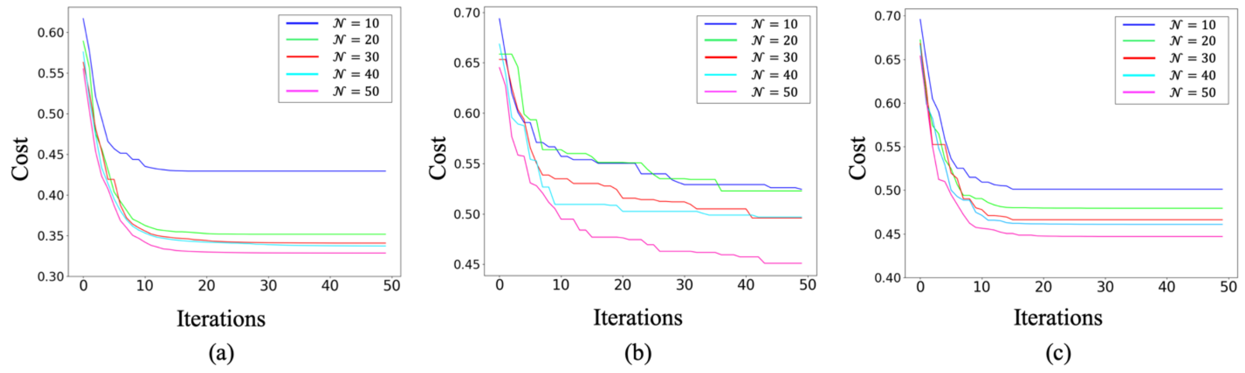


Figure 12 Comparison of the optimizations under varying PSO population size and iteration numbers: (a) solarPlant; (b) hoaHakanaia; (c) church

4.4.2 Effects of Gimbal Parameter

We now evaluate the effects of gimbal pitch angles on the optimization results. For the three test cases, Figs. 13 to15 respectively, show the histograms of the gimbal pitch angles extracted from the optimal paths and the histogram of the transformed surface normal at the geometrical models. The results illustrate that the pitch rotation angles vary significantly based on the model geometry. For example, solarPlant is a planar target tilted at approximate 45 degrees above the ground, thus all the cameras in the optimal paths are pointed downwards to the scene (between $[-60, -30]$). The hoaHakanaia statue, on the other hand, is a cylinder structure so most of the views are formulated in a horizontal direction (between $[-30, 30]$). The church model is geometrically more complicated as it contains both horizontal (e.g., flat roofs), vertical (e.g., walls) and oblique surfaces (e.g., gable roofs), and therefore a wider range of pitch rotations is required for full surface coverage (between $[-90, 5]$). Such observations show the importance of incorporating the gimbal rotation as another degree of freedom in order to improve the aerial inspection quality. Comparing the results under different values of ε, we also observed that the pitch angles of the optimal path gradually converged to the surface normal of the model geometry when ε decreases. We note that other than the solarPlant, the quality-optimal histograms of the hoaHakanaia and the church do not exactly fit the histogram of the model surface. This is caused by the camera poses being restricted by other parameters that formulated the viewpoint configuration space. While this could potentially be a limiting factor our results validate the overall methodology of our algorithm and the importance of considering quality. We project that future work incorporating the additional freedom of rotation in the gimbal could overcome the few limitations we encountered.

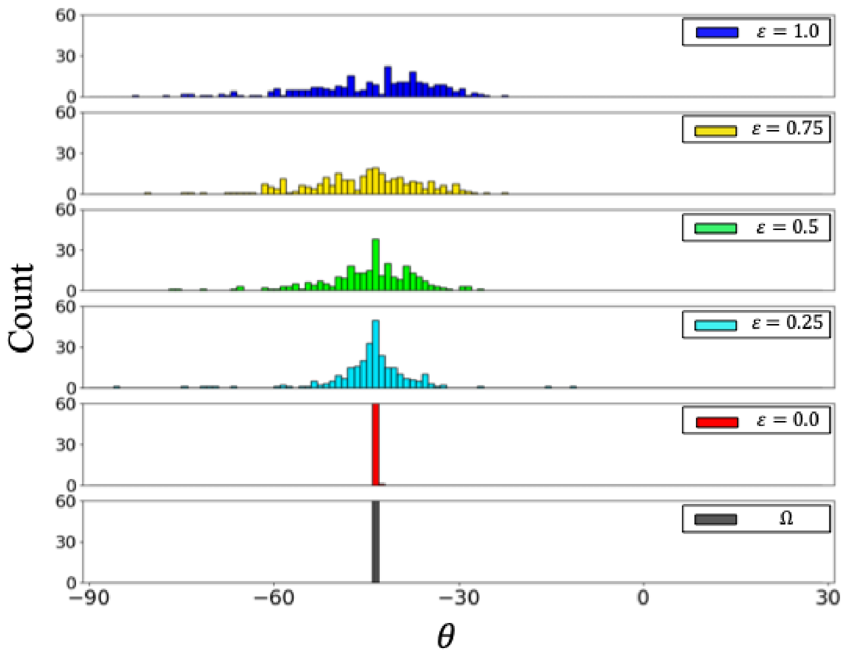


Figure 13 Histograms of gimbal pitch angles (in degrees) of the optimal path under different values of ε and the transformed model surfaces normal of solarPlant

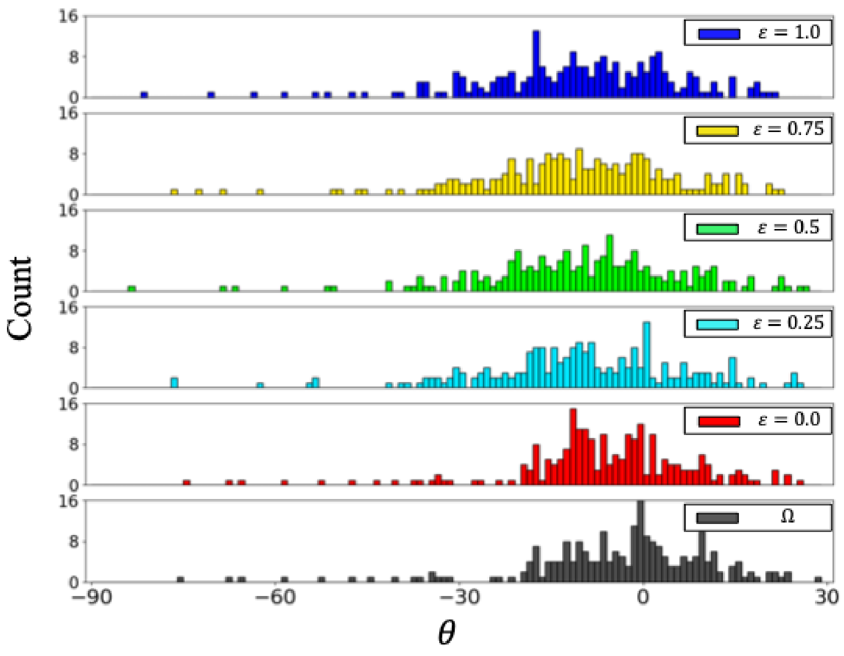


Figure 14 Histograms of gimbal pitch angles (in degrees) of the optimal path under different values of ε and the transformed model surfaces normal of hoaHakanaia

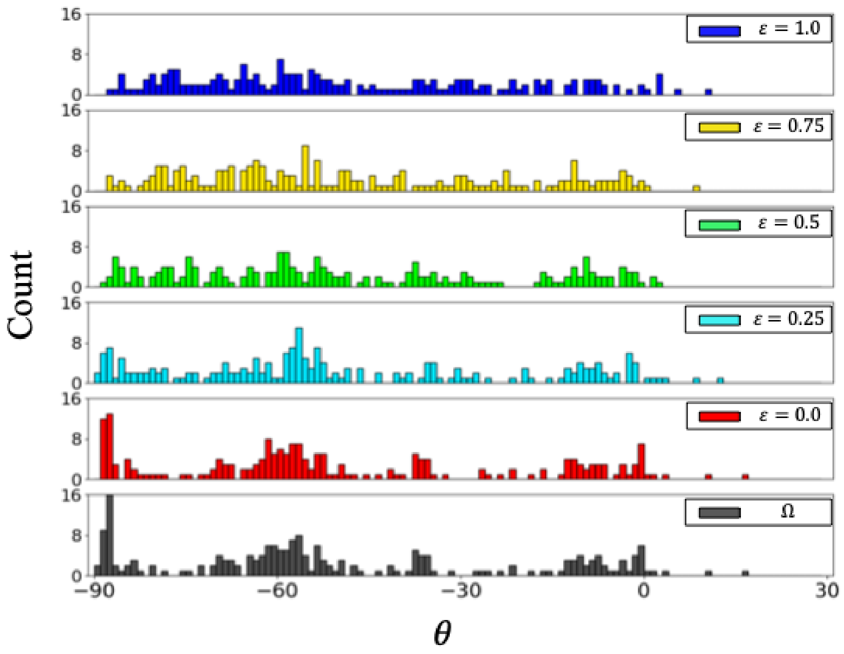


Figure 15 Histograms of gimbal pitch angles (in degrees) of the optimal path under different values of ε and the model surfaces normal of church

4.4.3 Runtime Efficiency

Finally, we approximate the computational efficiency of the proposed method by evaluating runtime of our algorithm on the three test scenes. The method is implemented with python 3.7 and the parallel computing is enabled in the OpenMP style (Chapman, Jost, & Van Der Pas, 2008). We performed the experiments on a PC workstation with the Intel CPU E5-2630, 64GB (DDR4 2133

MHZ) memory, running Ubuntu Linux 18.04. Although computational runtime is highly dependent on operating system, software implementation, computer hardware architectures, and many other sources of uncertainty, we hope this gives a sense of how our algorithm could perform. Table 6 shows the average of the total computational duration (in minutes) as well as the detailed time spent at each step (in seconds) in one iteration. For this test we disabled the path refinement step as the computation is performed only when needed. For all test cases, the computations take no more than 6 minutes demonstrating that our algorithm is easily computed offline, while onboard (the UAV) computation is likely currently infeasible. This is likely an acceptable tradeoff since detailed paths in UAV planning are often computed offline, prior to launch, relying on efficient trajectory generation algorithms to translate high-level paths into controller-followable commands (Hoffmann, Waslander, & Tomlin, 2008). The table also illustrates that the geometrically complicated models (i.e., church) may take longer to find admissible views, while the total duration of computation is mostly affected by effort to find the shortest path which is highly correlated to the size of the inspection target (i.e. $\mathcal{M}$).

Table 6 Computational duration of the three test cases

Model	View sampling (s)	At each iteration (s)				Total (min)
		Cost evaluation	Shortest path	Greedy heuristic	Particles update	
solarPlant	7.36	0.43	9.55	0.11	2.98	5.35
hoaHakanaia	9.28	0.40	5.41	0.09	2.12	4.32
church	11.22	0.34	4.62	0.08	1.66	2.48

Lastly, we provide a more detailed evaluation on the relationship between the computational duration and the model size, as well as the effects of a potential parallel computing implementation. We resample the church model to contain an arbitrary, user-defined number of triangular surfaces. Fig. 16 shows the optimization time of the church model with model sizes ($\mathcal{M}$) varying from 100 to 800 under three conditions: optimization with sequential computing, optimization with parallel computing and solving a pure TSP using the shortest path solver. The results show that for all the tests, the computation takes less than 10 minutes when the model size is less than 400, while it goes up dramatically with model sizes larger than 500. For all the steps in the optimization, the TSP (shown in red) consumes the majority of computing resources compared to other steps, and increases significantly with increased model size. Although these results are highly dependent on the specs of the executing computer and the software implementation, they provide insight into what parts of the algorithm should be focused on for speed up. Compared to the sequential computing (in green), the parallel computing (in blue) can reduce the total computational time, especially when model sizes increase. However, the increased efficiency is minor as the TSP solver $\mathcal{O}(\mathcal{M}^{2.2})$ runtime grows exponentially with model size compared to other steps $\mathcal{O}(\mathcal{M})$ in the proposed method. This increased computation can be potentially mitigated by reducing the number of times needed to compute the shortest paths in the optimization, or incorporating multi-agent UAV inspection scenarios, which could be considered in the future works.

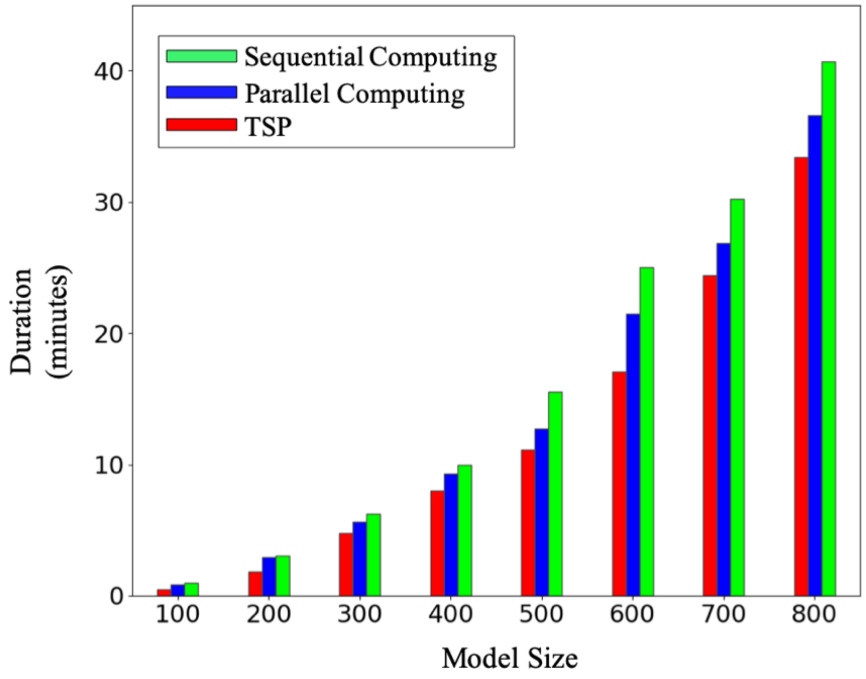


Figure 16 The relationship between the computation duration and the size ($\mathcal{M}$) of the church model with sequential (one thread) and parallel computing (12 threads)

5. CONCLUSIONS

In this study, we have proposed a new and efficient flight path planning method for camera-based aerial inspections of complex 3D structures. The method poses the coverage path planning into a particle swarm optimization (PSO) based framework that has the capability to co-optimize inspection efficiency **and** quality while still considering traditional vehicle constraints in path planning of UAVs. A benchmark method, composed of a quality evaluation metric and a simulated pin-hole camera model, is proposed to assess the inspection performance of the designed paths. The proposed method is assessed by comparing inspection paths against state-of-the-art CPP algorithms in three geometrically different scenes. The results show that our method provides greatly improved inspection quality, comparable path efficiency, and more flexible options compared to algorithms that only consider path length. To further investigate the method's performance, the optimization convergence, the effects of the gimbal system, as well as the runtime efficiency of the proposed method are thoroughly evaluated.

Future work should include the extension of the method to aerial photogrammetry applications. Because complexity grows exponentially with problem size, further work toward parallelizing the algorithm could increase the algorithm's efficiency, ideally leading to online computation on size-weight, and power (SWaP) constrained vehicles that increasingly leverage GPUs to perform onboard machine learning. The primary benefit of our algorithm is the flexibility it allows in designing functions that co-optimize performance. Work incorporating UAV energy consumption, predicted wind, and efficiency of required flight maneuvers into the strategy would likely yield even better results.

REFERENCES

Acar, E. U., Choset, H., Rizzi, A. A., Atkar, P. N., & Hull, D. (2002). Morse decompositions for coverage tasks. *The International Journal of Robotics Research, 21*(4), 331-344.

Almadhoun, R., Taha, T., Seneviratne, L., Dias, J., & Cai, G. (2016). A survey on inspecting structures using robotic systems. *International Journal of Advanced Robotic Systems, 13*(6), 1729881416663664.

Atkar, P. N., Choset, H., Rizzi, A. A., & Acar, E. U. (2001). *Exact cellular decomposition of closed orientable surfaces embedded in/spl Rfr//sup 3.* Paper presented at the Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on.

Bircher, A., Alexis, K., Burri, M., Kamel, M., Oettershagen, P., Omari, S., . . . Siegwart, R. (2015). Structural inspection planner dataset release (Online). Retrieved from https://github.com/ethz-asl/StructuralInspectionPlanner/wiki/Example-Results. Retrieved 2015 https://github.com/ethz-asl/StructuralInspectionPlanner/wiki/Example-Results

Bircher, A., Alexis, K., Burri, M., Oettershagen, P., Omari, S., Mantel, T., & Siegwart, R. (2015). *Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics.* Paper presented at the 2015 IEEE International Conference on Robotics and Automation (ICRA).

Bircher, A., Kamel, M., Alexis, K., Burri, M., Oettershagen, P., Omari, S., . . . Siegwart, R. (2016). Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots. *Autonomous Robots, 40*(6), 1059-1078.

Chapman, B., Jost, G., & Van Der Pas, R. (2008). *Using OpenMP: portable shared memory parallel programming* (Vol. 10): MIT press.

Cheng, P., Keller, J., & Kumar, V. (2008). *Time-optimal UAV trajectory planning for 3D urban structure coverage.* Paper presented at the Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on.

Choset, H., & Pignon, P. (1998). *Coverage path planning: The boustrophedon cellular decomposition.* Paper presented at the Field and Service Robotics.

Danner, T., & Kavraki, L. E. (2000). *Randomized planning for short inspection paths.* Paper presented at the Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on.

De Boor, C., De Boor, C., Mathématicien, E.-U., De Boor, C., & De Boor, C. (1978). *A practical guide to splines* (Vol. 27): Springer-Verlag New York.

Ellefsen, K. O., Lepikson, H. A., & Albiez, J. C. (2016). *Planning inspection paths through evolutionary multi-objective optimization.* Paper presented at the Proceedings of the 2016 on Genetic and Evolutionary Computation Conference.

Englot, B., & Hover, F. (2010). *Inspection planning for sensor coverage of 3D marine structures.* Paper presented at the Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on.

Englot, B., & Hover, F. S. (2012a). Sampling-based coverage path planning for inspection of complex structures.

Englot, B., & Hover, F. S. (2012b). *Sampling-Based Coverage Path Planning for Inspection of Complex Structures.* Paper presented at the Icaps.

Flood, M. M. (1956). The traveling-salesman problem. *Operations Research, 4*(1), 61-75.

Foo, J. L., Knutzon, J., Kalivarapu, V., Oliver, J., & Winer, E. (2009). Path planning of unmanned aerial vehicles using B-splines and particle swarm optimization. *Journal of aerospace computing, information, and communication, 6*(4), 271-290.

Galceran, E., Campos, R., Palomeras, N., Ribas, D., Carreras, M., & Ridao, P. (2015). Coverage path planning with real‐time replanning and surface reconstruction for inspection of three‐dimensional underwater structures using autonomous underwater vehicles. *Journal of Field Robotics, 32*(7), 952-983.

Galceran, E., & Carreras, M. (2013). A survey on coverage path planning for robotics. *Robotics and Autonomous Systems, 61*(12), 1258-1276.

Geng, L., Zhang, Y., Wang, P., Wang, J. J., Fuh, J. Y., & Teo, S. (2014). *UAV surveillance mission planning with gimbaled sensors.* Paper presented at the 11th IEEE International Conference on Control & Automation (ICCA).

Gonzalez, E., Alvarez, O., Diaz, Y., Parra, C., & Bustacara, C. (2005). *BSA: A complete coverage algorithm.* Paper presented at the Proceedings of the 2005 IEEE International Conference on Robotics and Automation.

Gottschalk, S., Lin, M. C., & Manocha, D. (1996). *OBBTree: A hierarchical structure for rapid interference detection.* Paper presented at the Proceedings of the 23rd annual conference on Computer graphics and interactive techniques.

Ham, Y., Han, K. K., Lin, J. J., & Golparvar-Fard, M. (2016). Visual monitoring of civil infrastructure systems via camera-equipped Unmanned Aerial Vehicles (UAVs): a review of related works. *Visualization in Engineering, 4*(1), 1-8.

Helsgaun, K. (2000). An effective implementation of the Lin–Kernighan traveling salesman heuristic. *European Journal of Operational Research, 126*(1), 106-130.

Hoffmann, G., Waslander, S., & Tomlin, C. (2008). *Quadrotor helicopter trajectory tracking control.* Paper presented at the AIAA guidance, navigation and control conference and exhibit.

Hover, F. S., Eustice, R. M., Kim, A., Englot, B., Johannsson, H., Kaess, M., & Leonard, J. J. (2012). Advanced perception, navigation and planning for autonomous in-water ship hull inspection. *The International Journal of Robotics Research, 31*(12), 1445-1464.

Jing, W., Polden, J., Lin, W., & Shimada, K. (2016). *Sampling-based view planning for 3d visual coverage task with unmanned aerial vehicle.* Paper presented at the Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on.

Karatzas, I., & Shreve, S. E. (1998). Brownian motion. In *Brownian Motion and Stochastic Calculus* (pp. 47-127): Springer.

Kennedy, J. (2011). Particle swarm optimization. In *Encyclopedia of machine learning* (pp. 760-766): Springer.

Krainin, M., Curless, B., & Fox, D. (2011). *Autonomous generation of complete 3D object models using next best view manipulation planning.* Paper presented at the Robotics and Automation (ICRA), 2011 IEEE International Conference on.

LaValle, S. M. (1998). Rapidly-exploring random trees: A new tool for path planning.

Lin, L., & Goodrich, M. A. (2014). Hierarchical heuristic search using a Gaussian mixture model for UAV coverage planning. *IEEE Transactions on cybernetics, 44*(12), 2532-2544.

Mansouri, S. S., Kanellakis, C., Wuthier, D., Fresk, E., & Nikolakopoulos, G. (2016). Cooperative Aerial Coverage Path Planning for Visual Inspection of Complex Infrastructures. *arXiv preprint arXiv:1611.05196*.

Marler, R. T., & Arora, J. S. (2010). The weighted sum method for multi-objective optimization: new insights. *Structural and multidisciplinary optimization, 41*(6), 853-862.

Mellinger, D., Michael, N., & Kumar, V. (2012). Trajectory generation and control for precise aggressive maneuvers with quadrotors. *The International Journal of Robotics Research, 31*(5), 664-674.

O'rourke, J. (1987). *Art gallery theorems and algorithms* (Vol. 57): Oxford University Press Oxford.

Papachristos, C., & Alexis, K. (2016). *Augmented reality-enhanced structural inspection using aerial robots.* Paper presented at the 2016 IEEE International Symposium on Intelligent Control (ISIC).

Papadopoulos, G., Kurniawati, H., & Patrikalakis, N. M. (2013). *Asymptotically optimal inspection planning using systems with differential constraints.* Paper presented at the Robotics and Automation (ICRA), 2013 IEEE International Conference on.

Phung, M. D., Quach, C. H., Dinh, T. H., & Ha, Q. (2017). Enhanced discrete particle swarm optimization path planning for UAV vision-based surface inspection. *Automation in Construction, 81*, 25-33.

Shen, C., Zhang, C., & Fels, S. (2007). *A multi-camera surveillance system that estimates quality-of-view measurement.* Paper presented at the Image processing, 2007. ICIP 2007. IEEE international conference on.

Shi, Y., & Eberhart, R. (1998). *A modified particle swarm optimizer.* Paper presented at the Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on.

Trelea, I. C. (2003). The particle swarm optimization algorithm: convergence analysis and parameter selection. *Information processing letters, 85*(6), 317-325.

Valette, S., Chassery, J. M., & Prost, R. (2008). Generic remeshing of 3D triangular meshes with metric-dependent discrete Voronoi diagrams. *IEEE Transactions on Visualization and Computer Graphics, 14*(2), 369-381.

Wang, C., Qi, F., Shi, G., & Wang, X. (2013). A sparse representation-based deployment method for optimizing the observation quality of camera networks. *Sensors, 13*(9), 11453-11475.

Wong, S. C., & MacDonald, B. A. (2003). *A topological coverage algorithm for mobile robots.* Paper presented at the Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453).

Zelinsky, A., Jarvis, R. A., Byrne, J., & Yuta, S. i. (1993). *Planning paths of complete coverage of an unstructured environment by a mobile robot.* Paper presented at the Proceedings of international conference on advanced robotics.