



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

공학석사 학위논문

2차원 균일 커버리지 경로 계획을 위한
효율적 알고리즘

An Efficient Algorithm for Uniform
Coverage Path Planning on Surfaces

2020년 8월

서울대학교 대학원

기계공학부

최 현 응

ABSTRACT

An Efficient Algorithm for Uniform Coverage Path Planning on Surfaces

by

Hyunwoong Choi

Department of Mechanical Engineering
Seoul National University

Coverage path planning (CPP) is widely used in numerous robotic applications. With progressively complex and extensive applications of CPP, automating the planning process has become increasingly important. This thesis proposes an efficient CPP algorithm based on a random sampling scheme for spray painting applications. We have improved on the conventional CPP algorithm by alternately iterating the path generation and node sampling steps. This method can reduce the computational time by reducing the number of sampled nodes. We also suggest a

new distance metric called “upstream distance” to generate reasonable path following given vector field. This induces the path to be aligned with a desired direction. Additionally, one of the machine learning techniques, support vector regression (SVR) is utilized to identify the paint distribution model. This method accurately predict the paint distribution model as a function of the painting parameters. We demonstrate our algorithm on several types of analytic surfaces and compare the results with those of conventional methods. Experiments are conducted to assess the performance of our approach compared to the traditional method.

Keywords: Uniform coverage path planning, Spray painting, Traveling salesman problem, Upstream distance, Support vector regression, Paint distribution model

Student Number: 2018-24497

Contents

Abstract	i
List of Tables	vi
List of Figures	vii
1 Introduction	1
1.1 Related Work	3
1.2 Contribution of Our Work	7
1.3 Organization of This Thesis	8
2 Preliminary Background	9
2.1 Elementary Differential Geometry of Surfaces in \mathbb{R}^3	10
2.1.1 Representation of Surfaces	10
2.1.2 Normal Curvature	10
2.1.3 Shape Operator	12
2.2 Traveling Salesman Problem	15
2.2.1 Definition	15

2.2.2	Variations of the TSP	17
2.2.3	Approximation Algorithm for TSP	19
2.3	Path Planning on Vector Fields	20
2.3.1	Randomized Path Planning	20
2.3.2	Upstream Criterion	20
2.4	Support Vector Regression	21
2.4.1	Single-Output SVR	21
2.4.2	Dual Problem of SVR	23
2.4.3	Kernel for Nonlinear System	25
2.4.4	Multi-Output SVR	26
3	Methods	29
3.1	Efficient Coverage Path Planning on Vector Fields	29
3.1.1	Efficient Node Sampling	31
3.1.2	Divide and Conquer Strategy	32
3.1.3	Upstream Distance	34
3.2	Uniform Coverage Path Planning in Spray Painting Applications .	35
3.2.1	Minimum Curvature Direction	35
3.2.2	Learning Paint Deposition Model	36
4	Results	38
4.1	Experimental Setup	38
4.2	Simulation Result	41
4.3	Discussion	41
5	Conclusion	45

Bibliography	47
--------------	----

국문초록	52
------	----

List of Tables

2.1	Five expressions to describe surfaces.	11
2.2	Description of vectors and matrices used in multi-output SVR . . .	27

List of Figures

1.1	Applications of Coverage Path Planning (a) spray painting, (b) farming, (c) cleaning, (d) surveillance, (e) inspection, (f) exploration . .	2
2.1	Representation of Traveling Salesman Problem	16
2.2	Representation of single-out Support Vector Regression	23
3.1	Scheme of sampling exclusion	32
3.2	Scheme of k -nearest- $TSP()$ algorithm	33
3.3	Three points on a vector field $F(q)$	34
3.4	Preferred direction for painting robot	36
3.5	Concept of 1D paint distribution learning	37
4.1	Four types of surface used in simulation (a) flat plane (b) quadratic plane (c) half ellipsoid (d) half torus	39
4.2	Representation of minimum curvature direction on each surface (a) flat plane (b) quadratic plane (c) half ellipsoid (d) half torus . . .	40

4.3	Simulation result of our Coverage Path Planning method on each surface (a) flat plane (b) quadratic plane (c) half ellipsoid (d) half torus	42
4.4	Simulation result of the optimized paint deposition on each surface (a) flat plane (b) quadratic plane (c) half ellipsoid (d) half torus .	43
4.5	Performance of the conventional method and our method on each surface (a) upstream cost (b) simulation time (c) standard deviation of paint thickness	44

1

Introduction

Coverage path planning (CPP) is the task of determining an efficient path that covers all points in the workspace under the prescribed environments and constraints [1]. The field of CPP has attracted considerable attention in recent decades due to its numerous applications in robotics that include spray painting [2], farming [3], cleaning [4], mining [5], surveillance [6], inspection [7], and exploration [8].

In early works on CPP applications, agents manually planned a path and programmed a robot based on their experiences. This hand-operated method is not only time-consuming but also renders the output subject to the daily conditions or an individual's proficiency. With progressively complex and extensive applications of CPP, automating the planning process has become increasingly important. An automatic planning algorithm reduces the required programming time and creates a standard form of path by providing guidelines for effective paths.

Studies on CPP with vector fields have yet to receive the same attention as the classical CPP problem even though such studies may prove useful. For example, an exploration rover may need to navigate tough, hilly terrain in the shifting winds, or

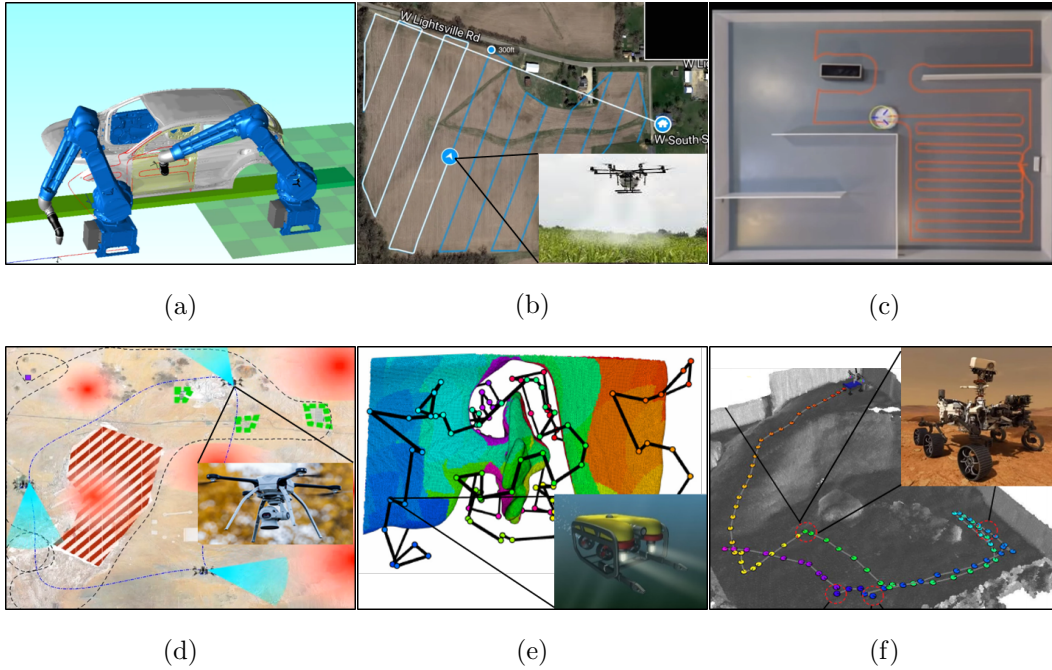


Figure 1.1: Applications of Coverage Path Planning (a) spray painting, (b) farming, (c) cleaning, (d) surveillance, (e) inspection, (f) exploration

an underwater inspection robot may be required to swim in the shifting currents of water. In such cases, the shortest path may no longer be the optimal path, as moving against the flow may worsen certain objectives [9]. A few studies have tried to reflect the preference for path to be aligned with the desired direction by specifying the robot dynamics determined by the power consumption model. However, because these studies utilized the concept of mechanical work, it was still difficult to identify the path that accurately followed the given vector field. Therefore, it is essential to devise a path planning method where the generated path exactly follows given vector field.

In CPP, some tasks are more challenging than others depending on the additional output requirements necessary for the application. For example, in automotive spray painting applications, agents must consider uniform deposition of the paint materials while creating a path on automotive surface [10]. The material deposition is typically multi-objective process that minimizes the cycle time while achieving an acceptable uniformity of deposition pattern. Uniformity is generally measured as a standard deviation of thickness of the deposited material over the surface. This new type of path planning is called uniform coverage path planning (UCPP), in which the complicated problem of the conflict of criteria for path generation arises and must be solved to balance the overall cost trade-offs with the requirement of paint uniformity [2].

In studies on UCPP, a fixed paint distribution model is assumed and then solved as an optimization problem. However, due to the complex physical phenomenon of the paint deposition process, a numerical model cannot accurately reflect the real behaviors of the robotic system. Therefore, it is essential to apply an alternative method for model prediction to create uniform coverage path with reasonable precision. Since numerous parameters associated with the environment affect the distribution model, it can be a solution to identify the relation between these parameters and deposition pattern.

1.1 Related Work

In the past few decades, a variety of CPP methods has been proposed with various robotic applications. Early works on CPP found in the literature typically address the problem of covering two-dimensional (2D) workspaces. Numerous studies in 2D

environments utilize various approaches such as artificial potential field [11], approximate cellular decomposition [12], exact cellular decomposition [13], template-based model [14], sampling-based model [15], neural networks [16], and fuzzy logic [17]. One of the most frequently used CPP methods in a flat plane is cellular decomposition, which divides the workspace into several non-overlapping regions without obstacles. Each cell is easily covered because it requires only simple back-and-forth motions to sweep the area [18, 19]. Another common method for CPP is template-based approaches. Hofner et al. [14] introduced a template-based method using motion templates and motion mosaic, where the obstacles are modified by problem environments. Since this method requires prescribed map and templates, it is difficult to adapt to environmental changes. In addition, it has difficulty investigating which area is uncovered by the created path. Thus, uncovered area appears near obstacles even after supplementary templates are utilized.

More recent studies have addressed the CPP problems in 3D environments. In 3D workspace, CPP typically requires a full sweep of 2D surfaces boundary of 3D structure [7]. One of the recently used CPP methods in 3D space is a sampling-based approach. Englot et al. [20] proposed a sampling-based algorithm to cover a boundary of complex 3D structures for ship hull inspections. Additionally, Papadopoulos et al. [21] built upon the idea of a sampling-based planning algorithm, which inspects complex structures using systems with differential constraints. Here, the CCP problem is separated into two subproblems, which are then solved sequentially. The first subproblem is a set cover problem (SCP) that determines a set of nodes for which the intersection of the areas centered at the nodes covers all workspaces. The second subproblem is traveling salesman problem (TSP) that determines the shortest path to visit all nodes generated by the previous SCP

solver. Because the TSP is NP-hard [22], the computational complexity of the algorithms increases exponentially as the number of nodes increases. Therefore, the solution is approximated using heuristic approaches to obtain less time complexity of the algorithm. Englot et al. [23] suggested polynomial-time approximation algorithm that improves the speed of the solver within guaranteed factors of optimality.

Earlier studies on randomized path planning on vector fields were mostly confined to application-specific contexts. The work of Jaillet et al. [24] addresses the planning problem on general vector fields. In this case, the preference for the direction of motion is determined by the negative gradient of the predefined potential function in the configuration space. Given a path created in configuration space, mechanical work can be calculated along the path, where only segments along which the potential increases contribute to the mechanical work. This criterion is utilized as the extent to which a certain path follows the desired direction as specified by the vector field in randomized motion planning problems. Ko et al. [9] formulated a more advanced mathematical criterion by which a certain path follows the desired direction as specified by the vector field, assuming premise that more control effort is required to move against the vector field.

Among the various CPP problems in 3D systems, Atkar et al. [2] present a planning algorithm specifically focused on the spray painting of automotive parts, which guarantees uniform paint distribution over a surface. They introduced a slicing plane method, which is an efficient UCPP method that renders the optimization problem of coverage path tractable. Their approach decomposed the main problem into two subproblems that can be solved separately. The method first generates a seed curve and then repeatedly offsets it sideways determining the

spacing between adjacent curves. In the first step, determining a seed curve affects the spatial location and orientation of the whole path, which, in turn, affects the uniformity of the paint thickness. In the second step, the method optimizes the spacing between the curves, called the pitch, to overlap the paint profiles of two adjacent routes in the direction perpendicular to the curves. This enables the uniform deposition of the paint. However, this method is useful only in a simple environment because a typical model for a paint distribution was applied. For an automotive body, the complex geometry of a non-planar surface may induce the deterioration of the painting quality because the paint distribution pattern depends on the geometric properties of surface.

Researchers have used simple planar deposition models to establish geometric projective models such as the rotation model, which is based on the rotated parabolic thickness profile [25], the bivariate Gaussian model [26], the Cauchy or Gaussian distributions model [27], and the beta distribution model [28]. Atkar et al. [29] proposed a highly accurate dual Gaussian paint deposition model that provides a significant improvement in painting pattern prediction over earlier models while retaining sufficient tractability for use in path planning tools. The deposition model consists of an offset 1D Gaussian revolved about the axis of a spray gun, and a 2D Gaussian aligned with the origin of the distribution model plane to capture the shape of the paint distribution. In this article, a new practical cumulative rate model for painting was derived, and the experiments on spray painting were performed to determine the 1D thickness profile function of the spatial painting distribution.

1.2 Contribution of Our Work

In this study, we propose a new method for automated UCPP, which resolves the aforementioned disadvantages. Also, we apply our efficient UCPP method especially in spray painting applications. Our first contribution is a new framework that can reduce the computational complexity. Conventional sampling-based CPP methods incrementally sample nodes until all points on a target surface is fully covered, and then generate a path that passes through all sampled nodes by solving the TSP. On the contrary, in our algorithm, we alternately iterate the path generation and node sampling steps. In the node sampling step, sampling is excluded for the area covered by the generated path. It leads to creating fewer nodes and reducing the computation time. The second contribution of our work is introducing a new distance metric for solving the TSP. Most TSP algorithms generally adopt Euclidean distance metric. However, this is not proper in some situations where there is a system drift because Euclidean distance cannot reflect the preference for path to be aligned in a direction parallel to the vector field. As an alternative to this, we regard that the distance against the given vector field is greater than Euclidean distance. This metric help determine the optimal path to be generated aligning with a desired direction. Finally, we apply an advanced method for model prediction to improve a painting quality. Researches so far have utilized fixed distribution model, but it can not reflect complex effect in physical environment. Here, we construct a machine learning network to predict an unknown distribution model by investigating the relationship between the system parameters and the paint distribution pattern.

1.3 Organization of This Thesis

The remainder of this thesis is organized as follows. In chapter 2, we review the necessary background for our study. Chapter 3 describes our experimental method used in our research. We enumerate the process while explaining our contributions. Chapter 4 presents the experimental results of our algorithm. Also, appropriate discussions are proposed to explain the experimental result. We compare our new method with the existing planning algorithm. Chapter 5 concludes our thesis with summary of our research and future work.

2

Preliminary Background

In this chapter, we focus on the theoretical background that is the basis of our study. First, we briefly review the elementary differential geometry of surfaces in \mathbb{R}^3 to understand the basic properties of them. Our ultimate goal is to understand the concept of shape operator stated in the last subsection. Second, we explain the TSP related to CPP problem. We define the original TSP and introduce its variations, and then we present heuristic algorithms to solve that problem efficiently. Third, we introduce the randomized path planning method and the upstream criterion that can quantify how much the path follows the given vector fields. Lastly, we review one of the machine-learning techniques, support vector regression (SVR), to predict the nonlinear relation between the input and output data. We begin with a single-output case and ultimately end with multi-output one.

2.1 Elementary Differential Geometry of Surfaces in \mathbb{R}^3

It is necessary to consider the geometry of the target surface when planning a uniform coverage path. For example, the curvature of a surface is an important and influential factor in UCPP since the coverage area greatly depends on the curvature at each point. Moreover, it may be difficult to achieve uniform coverage if the curvature is drastically changed along the path. The following subsections explain how the curvature is analyzed in mathematical form.

2.1.1 Representation of Surfaces

There are various ways to express 2D surfaces, and these expressions are complementary to each other. Table 2.1 represents the widely used expressions for common examples of surfaces. The most common method to express surfaces is with the parametric expression listed in the third row of the Table 1. This expression is also used in this thesis.

2.1.2 Normal Curvature

Assuming there are surface S and point p on S , one way to analyze the local shape of a surface S at a point p is to draw curves on S through p and find out the unique vector perpendicular to principal curvature vector of the curves. For a curve in \mathbb{R}^3 , to evaluate how much bent the curve is, we calculate the curvature that is determined by only position of the point p . However, for a surface, it is difficult to understand in which direction and how much the surface is bent because the curvature may vary based on the direction. More specifically, the position of a point p as well as the direction determine the curvature of a surface. We begin with definition of normal curvature:

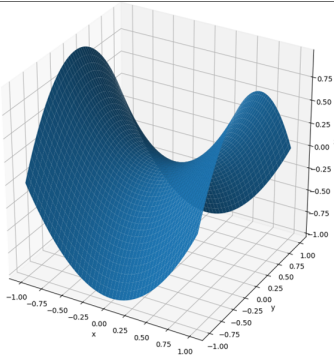
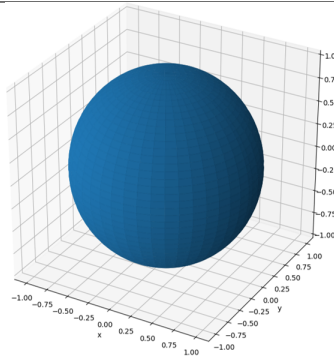
	hyperbolic paraboloid	unit sphere
		
Parametric expression	$(u, v, u^2 - v^2)$	$(\sin u \cos v, \sin u \sin v, \cos u)$
Implicit expression	$x^2 - y^2 - z = 0$	$x^2 + y^2 + z^2 = 1$
Explicit expression	$z(x, y) = x^2 - y^2$	$z(x, y) = \pm\sqrt{1 - x^2 - y^2}$
Solution set expression	$\{(x, y, z) \in \mathbb{R}^3 x^2 - y^2 - z = 0\}$	$\{(x, y, z) \in \mathbb{R}^3 x^2 + y^2 + z^2 = 1\}$
Contour expression	$F^{-1}(0)$ s.t. $F(x, y, z) = x^2 - y^2 - z$	$F^{-1}(1)$ s.t. $F(x, y, z) = x^2 + y^2 + z^2$

Table 2.1: Five expressions to describe surfaces.

Definition 2.1.1. Let S be a regular surface, and let $X : D \rightarrow \mathbb{R}^3$ be a parametrization of coordinate neighborhood V of S . Let $\alpha : I \rightarrow \mathbb{R}^3$ be a parametrization for a curve C that lies on S in V . The normal curvature of S along C is the function

$$\kappa_n(t) = \frac{1}{s'} \vec{T}' \cdot \vec{N} = k(\vec{P} \cdot \vec{N}) = k \cos \theta, \quad (2.1.1)$$

where \vec{T} and \vec{P} is the unit tangent vector and the principal normal vector of the curve \vec{C} , respectively, \vec{N} is the normal vector of the surface S , and θ is the angle between the \vec{P} and \vec{N} [30].

2.1.3 Shape Operator

Suppose that Z is a Euclidean vector field on a surface S in \mathbb{R}^3 . The derivative $\nabla_{\mathbf{v}} Z$ implies the rate of change of Z in the \mathbf{v} direction. Let α be a curve in S that has initial velocity $\alpha'(0) = \mathbf{v}$. Let Z_α be the restriction of Z to α , that is, the vector field $t \rightarrow Z(\alpha(t))$ on α . Then

$$\nabla_{\mathbf{v}} Z = (Z_\alpha)'(0). \quad (2.1.2)$$

Now, we can evaluate a mathematical measurement of the shape of a surface in \mathbb{R}^3 .

Definition 2.1.2. If p is a point on S , then for each tangent vector \mathbf{v} to S at p , let

$$S_p(\mathbf{v}) = -\nabla_{\mathbf{v}} U, \quad (2.1.3)$$

where U is a unit normal vector field on a neighborhood of p in S . S_p is called the shape operator of S at p derived from U [31].

The tangent plane of S at any point p , $T_p(S)$, is composed of all Euclidean vectors orthogonal to $U(p)$. Thus the derivative $\nabla_{\mathbf{v}}U$ in the \mathbf{v} direction implies how much the tangent planes are changing in the \mathbf{v} direction.

Theorem 2.1. *For each point p of $S \subset \mathbb{R}^3$, the shape operator is a linear operator*

$$S_p : T_p(S) \rightarrow T_p(S), \quad (2.1.4)$$

on the tangent plane of S at p .

Proof. Letting U be a unit vector, $U \cdot U = 1$. Thus by a property of derivatives,

$$0 = \nabla_{\mathbf{v}}(U \cdot U) = 2\nabla_{\mathbf{v}}U \cdot U(p) = -2S_p(\mathbf{v}) \cdot U(p), \quad (2.1.5)$$

where vector \mathbf{v} is tangent to S at p . Since U is also a normal vector, it follows that $S_p(\mathbf{v})$ is tangent to S at p . Thus S_p is a function from $T_p(S)$ to $T_p(S)$. The linearity of S_p is a consequence of a linearity property of derivatives:

$$S_p(a\mathbf{v} + b\mathbf{w}) = -\nabla_{a\mathbf{v}+b\mathbf{w}}U = -(a\nabla_{\mathbf{v}}U + b\nabla_{\mathbf{w}}U) = aS_p(\mathbf{v}) + bS_p(\mathbf{w}). \quad (2.1.6)$$

□

Since we have proved that the shape operator is a linear operator, it is possible to express the shape operator as a matrix form. First, we define some variables for the matrix expression. We can define three real-valued functions on D using X stated in Definition 2.1.1

$$\begin{aligned} E &= X_u \cdot X_u \\ F &= X_u \cdot X_v = X_v \cdot X_u \\ G &= X_v \cdot X_v, \end{aligned} \quad (2.1.7)$$

where $X_u = \frac{\partial X}{\partial u}$ and $X_v = \frac{\partial X}{\partial v}$

If S_p is the shape operator derived from U , we define three more real-valued functions on D :

$$\begin{aligned} L &= S_p(X_u) \cdot X_u \\ M &= S_p(X_u) \cdot X_v = S_p(X_v) \cdot X_u \\ N &= S_p(X_v) \cdot X_v. \end{aligned} \tag{2.1.8}$$

Because X_u, X_v are bases for the tangent space of S at each point p , it is clear that these functions uniquely determine the shape operator.

L, M , and N can be calculated by more simple way. For example, since $U \cdot X_u = 0$, partial differentiation with respect to v yields

$$0 = \frac{\partial}{\partial v}(U \cdot X_u) = U_v \cdot X_u + U \cdot X_{uv}. \tag{2.1.9}$$

Since $U_v = -S_p(X_v)$, the preceding equation becomes

$$S_p(X_v) \cdot X_u = U \cdot X_{uv}. \tag{2.1.10}$$

Therefore,

$$\begin{aligned} L &= S_p(X_u) \cdot X_u = U \cdot X_{uu} \\ M &= S_p(X_u) \cdot X_v = S_p(X_v) \cdot X_u = U \cdot X_{uv} \\ N &= S_p(X_v) \cdot X_v = U \cdot X_{vv}. \end{aligned} \tag{2.1.11}$$

For each point p of $S \subset \mathbb{R}^3$, and bases X_u, X_v of tangent space $T_p S$, the shape operator finally can be formulated as matrix form:

$$[S_p]_{\{X_u, X_v\}} = \begin{pmatrix} E & F \\ F & G \end{pmatrix}^{-1} \begin{pmatrix} L & M \\ M & N \end{pmatrix}. \quad (2.1.12)$$

2.2 Traveling Salesman Problem

Sampling-based CPP method is similar to solving the covering salesman problem, which is a variation of the original TSP. The following subsections provide an overview of the TSP to help clarify the process of CPP. We first describe the original TSP and introduce its variations, and then we present approximated algorithms to solve that problem with less computational complexity.

2.2.1 Definition

A basic interpretation of TSP is the situation of finding the shortest path of a salesperson who starts from an initial city, visits the prescribed n cities, and returns to the initial position in a way that each city is visited exactly once (Figure 2.1). Although this interpretation appears quite simple, this problem is known as one of the most challenging problems because it is a type of combinatorial optimization problem. The problem can be defined as a mathematical form by graph theory [32].

Let $G = (V, E)$ be a graph where V and E are a set of vertices and edges, respectively. For each edge e in E , a cost c_e is allocated. The objective is to find a Hamiltonian cycle such that sum of the costs of the edges on the route is as small as possible and that the path passes through each vertex only once. Without loss of generality, G is typically considered as a complete graph, otherwise, an

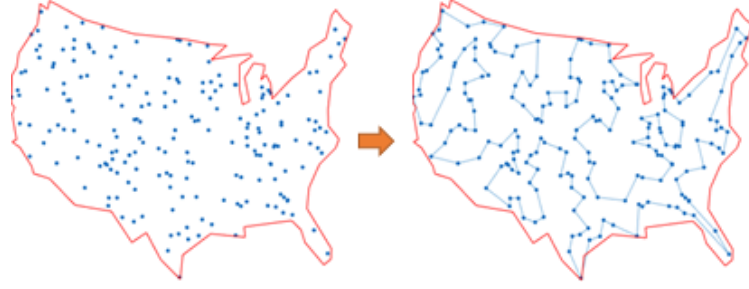


Figure 2.1: Representation of Traveling Salesman Problem

infinite cost is assigned to the missing edges. Letting V be $\{v_1, v_2, \dots, v_n\}$, E is expressed as $\{e_{ij} | i, j = 1, 2, \dots, n\}$ where i and j are indexes of two vertices. The matrix $C = (c_{ij})_{n \times n}$ is defined as a cost matrix, where the entry c_{ij} corresponds to the cost of the edge joining vertices v_i and v_j such that all diagonal entries are zeros. For TSP in real-world, V and E correspond to a set of cities and routes, respectively. The cost c_{ij} is typically set to the Euclidean distance between cities i and j .

Given a set of nodes and a distance metric between the nodes, one can generate a complete path by connecting all nodes while considering multiple objectives. The typical objective is to minimize the total path length, i.e., the goal is to find an order of nodes with the minimum cost. Many algorithms have been proposed to determine the optimal solution of the TSP and can be understood in the context of integer linear programming (ILP). Thus, the problem can be formulated as the following optimization problem:

$$\begin{aligned}
\min \quad & \sum_{i,j=1}^n c_{ij}x_{ij} \\
\text{s.t.} \quad & \sum_{i=1}^n x_{ij} = 1 \quad j = 1, 2, \dots, n \\
& \sum_{j=1}^n x_{ij} = 1 \quad i = 1, 2, \dots, n \\
& x_{ij} \in \{0, 1\} \\
& u_1 = 1 \\
& 2 \leq u_{ij} \leq n \quad i = 2, \dots, n \\
& u_i - u_j + 1 \leq n(1 - x_{ij}) \quad i, j = 2, \dots, n
\end{aligned} \tag{2.2.13}$$

In above optimization problem, the first three constraints specify that the passes through every node exactly once. Here, x_{ij} is a binary variable associated with every edge e_{ij} , which is equal to 1 if the edge e_{ij} is used in the optimal solution, otherwise, it is equal to 0. The other three constraints are MTZ constraints that prevent the solution from creating sub-tours. In other words, these sub-tour elimination constraints ensure that the solution contains no sub-tour involving less than n vertices. Variable u_i indicates the vertex i 's order of visit and this is proposed here to make the optimization problem most compact.

2.2.2 Variations of the TSP

There are many variations of the basic TSP originating from real applications. One of these variations, the path-TSP, is a problem to find a Hamiltonian path which has the minimum cost, between two nodes instead of returning to the initial node.

In the manner of graph theory, given two predefined vertices v_i and v_j in graph G , this problem can be solved as a TSP by substituting a cost c_{ij} with $-M$ where M is a large number. If no nodes are designated, the path-TSP becomes identical to the original TSP. In the path-TSP, this problem can be further classified according to whether the two nodes are fixed or free.

Unlike the classical TSP, the max-TSP is a problem to find a Hamiltonian cycle where the total cost of the edges in the path is a maximum. For example, in the reinforcement learning approach, a cost is replaced by the concept of reward, which increases the objective function. The max-TSP can be solved by replacing each edge cost with its minus value [33].

Letting the set of nodes V partition into clusters V_1, \dots, V_k , a clustered-TSP finds a minimum cost tour in G subject to the constraint that nodes in the same cluster must be visited sequentially. In other words, the tour can not go over another cluster until it visits all nodes in the present cluster. This problem converges to a basic TSP by allocating very large cost M to the each of inter-cluster edges.

With partitioned clusters V_1, \dots, V_k defined in the previous setting, covering-TSP finds the shortest cycle in graph G , which visits exactly one node from each cluster. This problem converges to the original TSP when each cluster has only one node, that is to say when every node makes its own cluster.

2.2.3 Approximation Algorithm for TSP

Because the TSP is known as an NP-hard problem, there is no exact algorithm to solve it with a polynomial time complexity. Therefore, many researchers have utilized a near-optimal algorithm to solve the TSP efficiently [32]. We, therefore, concentrate on a number of heuristic approaches known to yield reasonable TSP solutions in an empirical sense. Broadly speaking, heuristic algorithms for the TSP apply a tour construction approach which gradually constructs a solution by adding a new node at each step and tour improvement approach which improves a feasible solution by performing exchanges among nodes. The best method is a composite algorithm combining these two approaches.

Best insertion algorithm is one of the construction approach. It builds up a feasible tour by adding a new node into the existing path satisfying. Specifically, it construct a primary path consisting of arbitrary two nodes, and then repeat the procedure of inserting another node into the path satisfying least distance increment. The computational complexity of this type of algorithm varies between $O(n^2)$ and $O(n \log n)$ depending on the condition of problem. This method is not much different from greedy algorithm, but one can get a better solution than greedy method.

k -opt algorithm is one of the improvement approaches. It first finds out any initial tour. And then It optimizes the path by eliminating k arcs from the tour and immediately reconnecting the incomplete nodes in all possible ways. If shorter path is created, replace the initial tour with new one. Repeat this reconnection step until no significant performance enhancement occurs. In this case, the number of possible solutions is of the order of n^k . In general, since k is chosen as 2 or 3, the computational complexity of this type of algorithm is $O(n^2)$ or $O(n^3)$

2.3 Path Planning on Vector Fields

A given vector field defined in a configuration space can be used to represent a system flow such as a wind velocity field, water current, or gradient field for some potential function. Sometimes, the preference for the path to be aligned with the desired direction must be considered. The following subsections explain randomized path planning method based on sampling approach and then introduce a mathematical criterion that measures how much the created path deviates from the preferred direction as specified by the vector field.

2.3.1 Randomized Path Planning

One of the randomized path planning methods is the rapidly-exploring random tree (RRT) algorithm. It is one of the powerful path planning algorithms in the way that it quickly searches for a reasonable path in high-dimensional space. The main idea of RRT algorithm is that the tree expands as it stretches into the unexplored area. It is widely used in path planning because it ensures probabilistic completeness.

2.3.2 Upstream Criterion

Given a configuration space, a tangent vector representing the desired direction of movement is attached to each point on configuration space. The collection of vectors attached to the configuration space constitutes a vector field. Naturally, the most efficient thing to do is to always move along the direction of the vector field. Moving against the vector field requires more control effort. On the premise that minimizing this control effort is desirable, an integral functional, called an upstream criterion, is proposed to measure the extent to which the path goes in

the opposite direction of a given vector field. This path integral functional, the upstream criterion, can be expressed as:

$$\mathcal{U}(q) = \int_0^L (||f(q(s))|| - \langle f(q(s)), q'(s) \rangle) ds \quad (2.3.14)$$

2.4 Support Vector Regression

In UCPP, due to the complicated physical phenomenon of the paint deposition process, it is difficult to accurately predict the paint distribution pattern by applying a simple model. Hence, one of the machine learning techniques, SVR, is proposed as an alternative method for model prediction with reasonable accuracy. The following subsections present an overview of the SVR.

2.4.1 Single-Output SVR

The SVR is a type of supervised machine learning technique that establishes the mapping between input and output data. This technique presents one of the most robust prediction methods based on a statistical learning framework. In the SVR, the simplest case is that the output is 1D (scalar) data for the so-called single-output SVR. Suppose we have training data set $\{(x_1, y_1), \dots, (x_n, y_n)\} \subset \mathbb{R}^d \times \mathbb{R}$ where d is the input dimension. The ε -SVR, a well-known single-output SVR, builds a model $f(x)$ that has at most ε -deviation from the actual target value y_i for all the training data [34]. At the same time, the function is also as flat as possible. We begin the discussion by expressing the hypothesis as a linear function f :

$$f(x) = \langle x, w \rangle + b \quad w \in \mathbb{R}^d, \quad b \in \mathbb{R}. \quad (2.4.15)$$

where $\langle \cdot, \cdot \rangle$ denote inner product in \mathbb{R}^d

Flatness implies a small w in the case of equation 2.4.15. One way to guarantee this requirement is minimizing the norm, $\|w\|^2 = \langle w, w \rangle$. We can write this problem as the following convex optimization problem:

$$\begin{aligned} \min \quad & \frac{1}{2} \|w\|^2 \\ \text{s.t.} \quad & y_i - \langle x_i, w \rangle - b \leq \varepsilon \\ & \langle x_i, w \rangle + b - y_i \leq \varepsilon \end{aligned} \tag{2.4.16}$$

The implicit assumption in equation 2.4.16 is that the function f approximates all training data pairs $(x_i$ and $y_i)$ with ε -precision.

In some cases, however, some errors are allowed, in which case the slack variables ξ, ξ^* are introduced, similar to the soft margin loss function used in support vector machine (SVM). New optimization problem is proposed as follows:

$$\begin{aligned} \min \quad & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n (\xi_i + \xi_i^*) \\ \text{s.t.} \quad & y_i - \langle x_i, w \rangle - b \leq \varepsilon + \xi_i^* \\ & \langle x_i, w \rangle + b - y_i \leq \varepsilon + \xi_i \\ & \xi_i, \xi_i^* \geq 0. \end{aligned} \tag{2.4.17}$$

The constraint C in equation 2.4.17 reflects the ratio of importance between the flatness of function f and the total amount of deviations larger than ε . Figure 2.2 depicts the overall situation graphically. Because the deviations are penalized in a linear fashion, only the points outside the shaded region contribute to the cost in so far.

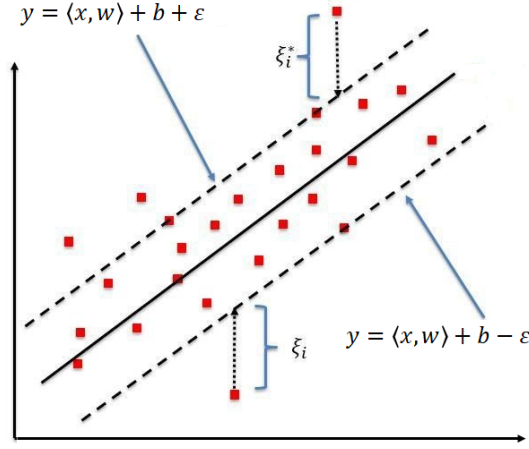


Figure 2.2: Representation of single-out Support Vector Regression

2.4.2 Dual Problem of SVR

The previous description can be solved more easily by means of dual function. Moreover, as we will see in the next subsection, the dual formulation provides the key for extending the SVR to a nonlinear function. Hence, we use a standard dualization method which formulate a Lagrange function from the primal objective function and the corresponding constraints by introducing Lagrange multipliers [35].

$$\begin{aligned}
 L = & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n (\xi_i + \xi_i^*) - \sum_{i=1}^n (\eta_i \xi_i + \eta_i^* \xi_i^*) \\
 & - \sum_{i=1}^n \alpha_i (\epsilon + \xi_i - y_i + \langle x_i, w \rangle + b) \\
 & - \sum_{i=1}^n \alpha_i^* (\epsilon + \xi_i^* + y_i - \langle x_i, w \rangle - b)
 \end{aligned} \tag{2.4.18}$$

Here L is the Lagrange function and $\alpha_i, \alpha_i^*, \eta_i, \eta_i^*$ are Lagrange multipliers. The

dual variables in equation 2.4.18 have to satisfy positivity constraints, i.e.

$$\alpha_i, \alpha_i^*, \eta_i, \eta_i^* \geq 0 \quad (2.4.19)$$

The saddle point condition is that the partial derivatives of L with respect to the primal variables have to become zero for optimality:

$$\frac{\partial L}{\partial b} = \sum_{i=1}^n (\alpha_i^* - \alpha_i) = 0 \quad (2.4.20)$$

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^n (\alpha_i^* - \alpha_i) x_i = 0 \quad (2.4.21)$$

$$\frac{\partial L}{\partial \xi} = C - \alpha_i - \eta_i = 0 \quad (2.4.22)$$

$$\frac{\partial L}{\partial \xi^*} = C - \alpha_i^* - \eta_i^* = 0 \quad (2.4.23)$$

Above equations are substituted for equation 2.4.18 to yields the dual optimization problem.

$$\begin{aligned} \max_{\alpha_i, \alpha_i^*} \quad & -\frac{1}{2} \sum_{i,j=1}^n (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) \langle x_i, x_j \rangle \\ & - \varepsilon \sum_{i=1}^n (\alpha_i + \alpha_i^*) + \sum_{i=1}^n y_i (\alpha_i - \alpha_i^*) \\ \text{s.t.} \quad & \sum_{i=1}^n (\alpha_i - \alpha_i^*) = 0, \quad 0 \leq \alpha_i, \alpha_i^* \leq C \end{aligned} \quad (2.4.24)$$

Equation 2.4.21 can be rewritten as follows:

$$w = \sum_{i=1}^n (\alpha_i - \alpha_i^*) x_i \quad (2.4.25)$$

Thus,

$$f(x) = \sum_{i=1}^n (\alpha_i - \alpha_i^*) \langle x_i, x \rangle + b \quad (2.4.26)$$

This is a support vector expansion where w can be described as a linear combination of the x_i s. The complexity of the function is independent of dimension of the input data and rather depends on the number of support vectors. Moreover, since $f(x)$ is described as an inner product of two data, we need not calculate w explicitly. These properties will be useful for formulation of nonlinear system.

2.4.3 Kernel for Nonlinear System

Nonlinear SVR algorithm could be achieved by processing the training input data into some feature space F by a mapping $\Phi : X \rightarrow F$, and then applying the standard SVR algorithm. As noted in the previous subsection, the SVR algorithm only depends on inner products between input data. Hence it is enough to know $K(x, x') := \langle \Phi(x), \Phi(x') \rangle$ rather than Φ explicitly. This analysis allows us to restate the SVR optimization problem by substituting $\langle x, x' \rangle$ with $K(x, x')$:

$$\begin{aligned} \max_{\alpha_i, \alpha_i^*} \quad & -\frac{1}{2} \sum_{i,j=1}^n (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) K(x_i, x_j) \\ & - \epsilon \sum_{i=1}^n (\alpha_i + \alpha_i^*) + \sum_{i=1}^n y_i (\alpha_i - \alpha_i^*) \\ \text{s.t.} \quad & \sum_{i=1}^n (\alpha_i - \alpha_i^*) = 0, \quad 0 \leq \alpha_i, \alpha_i^* \leq C \end{aligned} \quad (2.4.27)$$

Also, the equation 2.4.25, 2.4.26 may be reformulated as

$$w = \sum_{i=1}^n (\alpha_i - \alpha_i^*) \Phi(x_i), \quad (2.4.28)$$

and

$$f(x) = \sum_{i=1}^n (\alpha_i - \alpha_i^*) K(x_i, x) + b. \quad (2.4.29)$$

2.4.4 Multi-Output SVR

Notation

The following notations are used in this subsection. Vectors and matrices are denoted by small bold letters and capital letters, respectively. If A is an $n \times m$ matrix, we denote by $A^i \in \mathbb{R}^m$ and $A_j \in \mathbb{R}^n$ the i -th row and the j -th column of A . Let $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$ be a set of training data, where $\mathbf{x}_i \in \mathbb{R}^d$ is an input data and $\mathbf{y}_i \in \mathbb{R}^m$ is an output data. n is the number of data pairs, also d and m are the dimension of input and output space, respectively. Some vectors and matrices, derived from this basic setting, are predefined with their dimension in Table 2.2.

The previous single-output framework is extended into multi-output SVR where output data are in multi-dimensional space. However, linear soft margin loss function is no longer available in the multi-output case. Hence, the least-square SVR is proposed to replace the linear loss with quadratic one. Given training dataset, multi-output regression focuses on predicting an output vector $\mathbf{y} \in \mathbb{R}^m$ from a given input vector $\mathbf{x} \in \mathbb{R}^d$ by identifying a mapping from \mathbb{R}^d to \mathbb{R}^m . The multi-output SVR solves this problem by finding W and B that minimizes the following objective function with some constraints:

$$\begin{aligned} \min_{W, B} \quad & \frac{1}{2} \text{tr}(W^T W) + \gamma \frac{1}{2} \text{tr}(\Xi^T \Xi) \\ \text{s.t.} \quad & Y = X^T W + B + \Xi \end{aligned} \quad (2.4.30)$$

where Ξ is soft margin matrix.

$X = (\mathbf{x}_1, \dots, \mathbf{x}_n) \in \mathbb{R}^{d \times n}$	
$Y = (\mathbf{y}_1, \dots, \mathbf{y}_n)^T \in \mathbb{R}^{n \times m}$	
$W = (\mathbf{w}_1, \dots, \mathbf{w}_m) \in \mathbb{R}^{d \times m}$	
$\Xi = (\xi_1, \dots, \xi_m) \in \mathbb{R}_+^{n \times m}$	
$\mathbf{b} = (b_1, \dots, b_m)^T \in \mathbb{R}^m$	
$B = \underbrace{(\mathbf{b}, \dots, \mathbf{b})^T}_n \in \mathbb{R}^{n \times m}$	
$A = (\alpha_1, \dots, \alpha_m) \in \mathbb{R}^{n \times m}$	
$\tilde{\mathbf{y}} = (\mathbf{Y}_1^T, \dots, \mathbf{Y}_m^T)^T \in \mathbb{R}^{mn}$	
$\tilde{\alpha} = (\alpha_1^T, \dots, \alpha_m^T)^T \in \mathbb{R}^{mn}$	
$\mathbf{0}_n = (0, \dots, 0)^T \in \mathbb{R}^n$	
$\mathbf{1}_n = (1, \dots, 1)^T \in \mathbb{R}^n$	
$\mathbb{I}_n = \begin{pmatrix} 1 & \dots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \dots & 1 \end{pmatrix} \in \mathbb{R}^{n \times n}$	
	$\Omega = \underbrace{\begin{pmatrix} X^T X & \dots & X^T X \\ \vdots & \ddots & \vdots \\ X^T X & \dots & X^T X \end{pmatrix}}_m \in \mathbb{R}^{mn \times mn}$
	$Q = \underbrace{\begin{pmatrix} X^T X & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & X^T X \end{pmatrix}}_m \in \mathbb{R}^{mn \times mn}$
	$P = \underbrace{\begin{pmatrix} \mathbf{1}_n & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \mathbf{1}_n \end{pmatrix}}_m \in \mathbb{R}^{mn \times m}$
	$H = \Omega + \gamma^{-1} \mathbb{I}_{mn} + mQ \in \mathbb{R}^{mn \times mn}$
	$S = P^T H^{-1} P \in \mathbb{R}^{m \times m}$

Table 2.2: Description of vectors and matrices used in multi-output SVR

The Lagrange function of the above optimization problem is

$$\begin{aligned}
 L = & \frac{1}{2} \text{tr}(W^T W) + \gamma \frac{1}{2} \text{tr}(\Xi^T \Xi) \\
 & - \text{tr}(A^T (X^T W + B + \Xi - Y))
 \end{aligned} \tag{2.4.31}$$

The following set of linear matrix equations are obtained from KKT conditions:

$$\frac{\partial L}{\partial W} = W - XA = 0 \quad (2.4.32)$$

$$\frac{\partial L}{\partial \mathbf{b}} = A^T \mathbf{1}_n = 0 \quad (2.4.33)$$

$$\frac{\partial L}{\partial \xi} = \gamma \Xi - A = 0 \quad (2.4.34)$$

$$\frac{\partial L}{\partial A} = X^T + B + \Xi - Y = 0 \quad (2.4.35)$$

Eliminating W and Ξ , one can get the following linear matrix equation:

$$\begin{pmatrix} \mathbf{0}_{ml \times m} & P^T \\ P & H \end{pmatrix} \begin{pmatrix} \mathbf{b} \\ \alpha \end{pmatrix} = \begin{pmatrix} 0_m \\ \mathbf{y} \end{pmatrix} \quad (2.4.36)$$

However, it is difficult to find an efficient algorithm to solve this problem since equation 2.4.36 is not positive definite. This can be overcome by reformulating it into as a positive definite form:

$$\begin{pmatrix} S & \mathbf{0}_{mn \times mn} \\ \mathbf{0}_{m \times m} & H \end{pmatrix} \begin{pmatrix} \mathbf{b} \\ H^{-1}Pb + \alpha \end{pmatrix} = \begin{pmatrix} P^T H^{-1} \mathbf{y} \\ \mathbf{y} \end{pmatrix} \quad (2.4.37)$$

This new linear system guarantees a unique solution, and thus can be solved by fast and efficient numerical optimization methods. Finally, letting the solution of above equation be $\tilde{\alpha}^*$ and b^* , the corresponding decision function for the multi-outputs is

$$f(x) = \begin{pmatrix} \sum_{i=1}^m \sum_{j=1}^n \tilde{\alpha}_{i,j}^* K(x, x_j) \\ \vdots \\ \sum_{i=1}^m \sum_{j=1}^n \tilde{\alpha}_{i,j}^* K(x, x_j) \end{pmatrix} + b^{*T} \quad (2.4.38)$$

3

Methods

In this chapter, we describe the proposed efficient UCPP method. In the first section, we explain the way we create a complete coverage path and the way we reduce the computation time with our iterative method. Additionally, we propose the new distance metric to reflect the preference for a path to be aligned with a given vector field. The second section then proposes techniques to achieve uniform coverage in spray painting applications.

3.1 Efficient Coverage Path Planning on Vector Fields

Conventional sampling-based CPP methods incrementally create nodes until all points on a target surface is fully covered, and then generate a path that passes through all the sampled nodes by solving the TSP. Contrary to existing CPP algorithms, we do not separate the problem into two subproblems but address them at once to reduce the computational complexity of the algorithm. We alternately iterate the path generation and node sampling steps. Furthermore, assuming that

moving against the vector field is regarded as traveling longer distances, we formulate a more advanced distance metric and apply this metric to the TSP solver. Algorithm *New Coverage Path Planning()* summarizes our new coverage path planning method.

Algorithm *New Coverage Path Planning*

Input: surface S , initial node q_1 on S

Initialize: $N = \{q_1\}$, $E = \emptyset$, $T = \{N, E\}$

```

1: while True do
2:    $q_{new} = \text{MakeNewNode}(S, T)$ 
3:   if  $C = \emptyset$  then
4:      $N \leftarrow N \cup \{q_{new}\}$ 
5:      $E \leftarrow hTSP(S, N, E)$ 
6:      $T \leftarrow \{N, E\}$ 
7:      $C \leftarrow S - \text{Coverage}(S, T)$ 
8:   else
9:     break
10:  end if
11: end while
12: return  $T$ 

```

We first construct an initial path T of which the root is the predefined initial node q_1 . Then, at each iteration step, we expand the path by executing functions *MakeNewNode()* and *k-nearest-TSP()*. After we obtain a new path as an output of the algorithms, we can compute the region that is covered by the newly created path. If all points on a target surface are fully covered, we terminate the algorithm, otherwise, the iteration process is repeated.

3.1.1 Efficient Node Sampling

When we expand the path, we first sample a new node using the function *MakeNewNode()*. Below algorithm represents the process of this function which takes the current path as an input and returns new node.

Algorithm *MakeNewNode()*

Input: surface S , path T

- 1: **if** q on S is covered by T **then**
 - 2: $D(q) = 0$
 - 3: **else**
 - 4: $D(q) = 1$
 - 5: **end if**
 - 6: Compute probability density function $P(q) \propto D(q)$ s.t. $\int_S P(q) = 1$
 - 7: Sample a node $q_{new} \leftarrow \text{rand}(D(q))$
 - 8: **return** q_{new}
-

Here, we propose our first contribution in which sampling is restricted in the area covered by the generated path. Figure 3.1 describes this procedure. When we sample a new node, point p has much higher probability to be sampled than point q because q is already covered by the path. Fewer nodes will be sampled by utilizing this method. Because the computational complexity of the exact TSP algorithm is proportional to $n!$, where n is the number of sampled nodes, our method can significantly reduce the computation time.

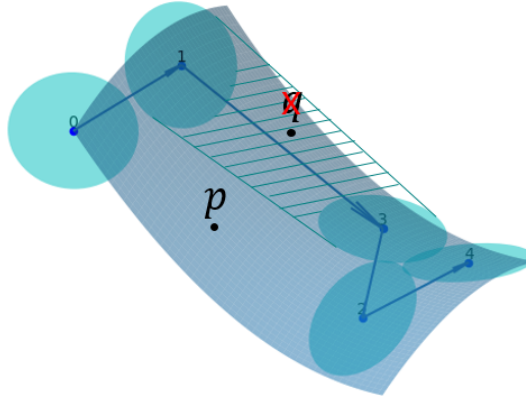


Figure 3.1: Scheme of sampling exclusion

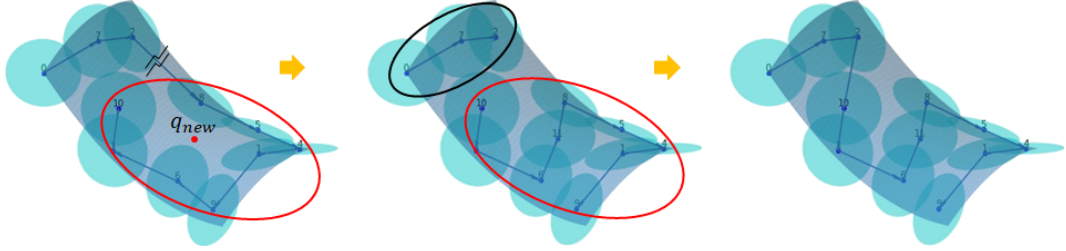
3.1.2 Divide and Conquer Strategy

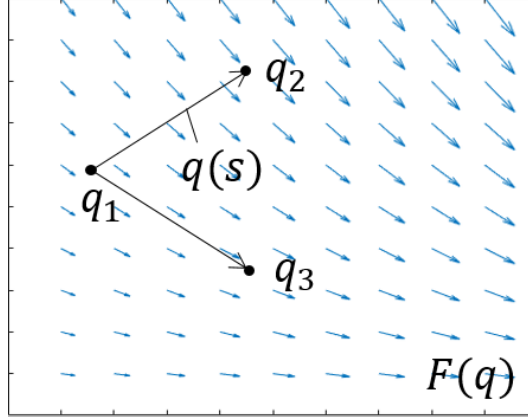
The divide and conquer algorithm works by breaking down a main problem into two or more subproblems of the same or similar type until the subproblems become simple enough to be solved. The solutions of the subproblems are then combined to give a solution of the original problem. Below algorithm describes the process of function, *k-nearest-TSP()* which is one of the heuristic TSP solvers.

Figure 3.2 describes this algorithm graphically. We first find the k -nearest nodes of q_{new} and then solve the TSP using only these k nodes. Finally, we connect this newly generated path with the original sub-path. This algorithm has a complexity that is proportional to $n \times k! \times k^2$. Therefore, if k is set to a much smaller value than n , the computational complexity is greatly reduced.

Algorithm $hTSP()$ **Input:** surface S , sampled node q_{new} , path T **Initialize:** set parameter k

- 1: **if** $n(N) \leq k$ **then**
- 2: $k \leftarrow n(N)$
- 3: **end if**
- 4: Find k -nearest nodes near q_{new}
 $N_k = \{q_1, \dots, q_k\}$, where q_i is one of the k -nearest nodes
- 5: Solve original TSP problem using N_k
 $E_{sub} \leftarrow TSP(N_k)$
- 6: Break tree T removing the node in N_k one by one
 $T_k = \{T_1, T_2, \dots\}$, where T_i is a fragmented tree
- 7: Connect E_{sub} and all elements in T_k
 $E = Connect(E_{sub}, T_k)$
- 8: **return** E

Figure 3.2: Scheme of k -nearest-TSP() algorithm

Figure 3.3: Three points on a vector field $F(q)$

3.1.3 Upstream Distance

In the TSP algorithms, distance metric between two nodes must be defined to solve the optimization problem. And TSP solvers generally adopt the Euclidean distance metric for its convenience. However, this may not adequate in some situations where there is a system drift. Therefore, a new distance matrix should be introduced because a Euclidean distance cannot reflect the preference for the path to be aligned with a direction parallel to the vector field. Here we suggest a new distance metric called the “upstream distance,” which is our second contribution. It can be formulated in a form derived from the upstream criterion previously mentioned in chapter 2.3.2.

Figure 3.3 shows three points on a certain vector field. Although the Euclidean distance between q_1 and q_2 is the same as that between q_1 and q_3 , we deliberately assume that the former distance is greater than the latter one. In other words, we regard that the distance between two points placed in a direction that crosses the stream is larger than the original Euclidean distance. Therefore, we define the

upstream distance as follows:

$$dist_{up}(q_1, q_2) = ||q_1, q_2|| + \int_{path} |f(q(s)) \times q'(s)| \quad (3.1.1)$$

The first term is a Euclidean distance between q_1 and q_2 , and the second term is the additional distance that reflects the fact that moving against the vector field is considered as traveling a farther distance. This new metric enables the optimal path to follow the desired direction.

3.2 Uniform Coverage Path Planning in Spray Painting Applications

In this section, we particularly focus on spray painting applications and propose techniques for UCPP on surface of automotive body. Since spray painting task is usually performed by a robot, additional constraints related to the painting robot must be considered. In addition, geometric properties of the automotive part should also be analyzed because they significantly affect uniform coverage.

3.2.1 Minimum Curvature Direction

We have explained how to plan coverage path on a given vector field in the previous chapter, but have not yet discussed how to create a vector field. In spray painting applications, there are preferences for robots to move in a certain direction. For example, given the geometry of an automotive body, it is preferred to move in the direction of minimum curvature (Figure 3.4). This is due to the restriction that industrial robots usually move in a straight line.

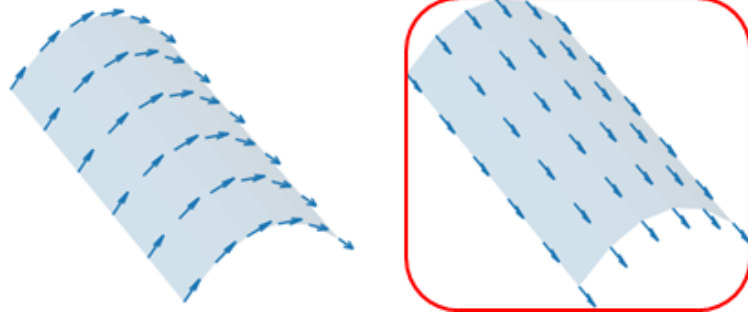


Figure 3.4: Preferred direction for painting robot

The normal curvature of a 2D surface can generally be expressed as a multivariate function of the position and direction; therefore, it is difficult to obtain a solution with analytic form. However, we can easily compute the principal curvature and corresponding direction by means of a shape operator.

Shape operator S_p is formulated as a matrix form in equation 2.1.12. Then, the eigenvalues k_1 and k_2 imply principal curvatures. If we express the corresponding eigenvectors as $v_1 = (a_1, b_1)^T$, $v_2 = (a_2, b_2)^T$, the corresponding principal direction can be formulated as $v_1 = a_1X_u + b_1X_v$ and $v_2 = a_2X_u + b_2X_v$.

So, we can determine the optimal direction as follows:

$$\begin{cases} v_1 & \text{if } |k_1| \leq |k_2| \\ v_2 & \text{if } |k_2| \leq |k_1| \end{cases} \quad (3.2.2)$$

If we have an analytic form of a surface, we can create a vector field on the surface applying the above analysis.

3.2.2 Learning Paint Deposition Model

One particularity of coverage path planning in painting applications, which is different from other forms of coverage path planning, is that uniformity of paint

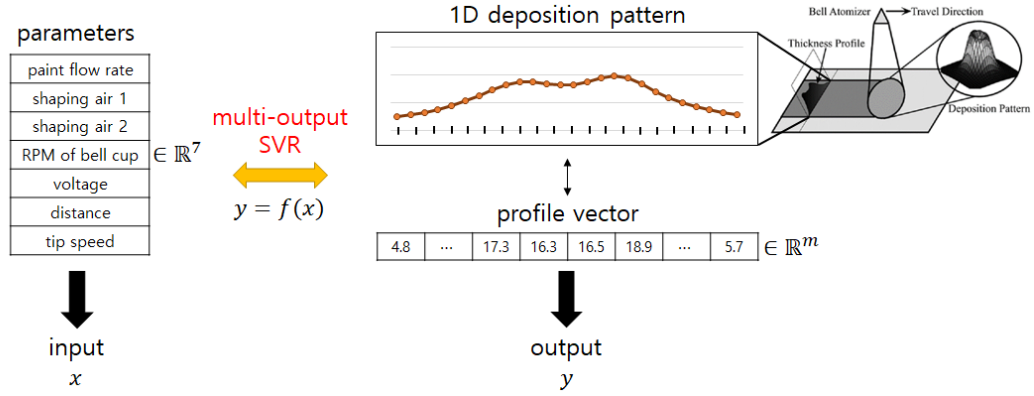


Figure 3.5: Concept of 1D paint distribution learning

distribution must be considered. The paint deposition quality highly depends on the operating parameters. Although many parameters affect the paint distribution model, we consider only seven parameters: paint flow rate, shaping air 1, 2, rotational frequency of bell cup, voltage, distance to the painted parts, and speed of robot's end-effector. Our third contribution is to determine the relation between the system parameters and paint distribution pattern by constructing SVR network. This machine learning technique accurately predict the distribution model as a function of the parameters. In general, the paint profile is represented in 2D domain, but in this thesis, it is reduced to 1D for the sake of easy analysis.

Figure 3.5 shows the schematic of the 1D paint profile learning, summarizing the problem definition. After we identify the paint distribution model, we optimize the painting parameters reducing the standard deviation of paint thickness over the surface.

4

Results

In this chapter, we demonstrate our new UCPP algorithm and present the experimental results. Also, we evaluate the performance of our method by comparing the results with that of the conventional sampling-based method.

4.1 Experimental Setup

Here, we simulate our algorithm on four types of surfaces, flat plane, quadratic plane, half ellipsoid, half torus (Figure 4.1). Training data for the paint distribution pattern was provided by the Doolim-Yaskawa Research Center. All the algorithms are implemented and performed with Python3 on an Intel Core i7-6700 3.40GHz CPU with 16GB of memory.

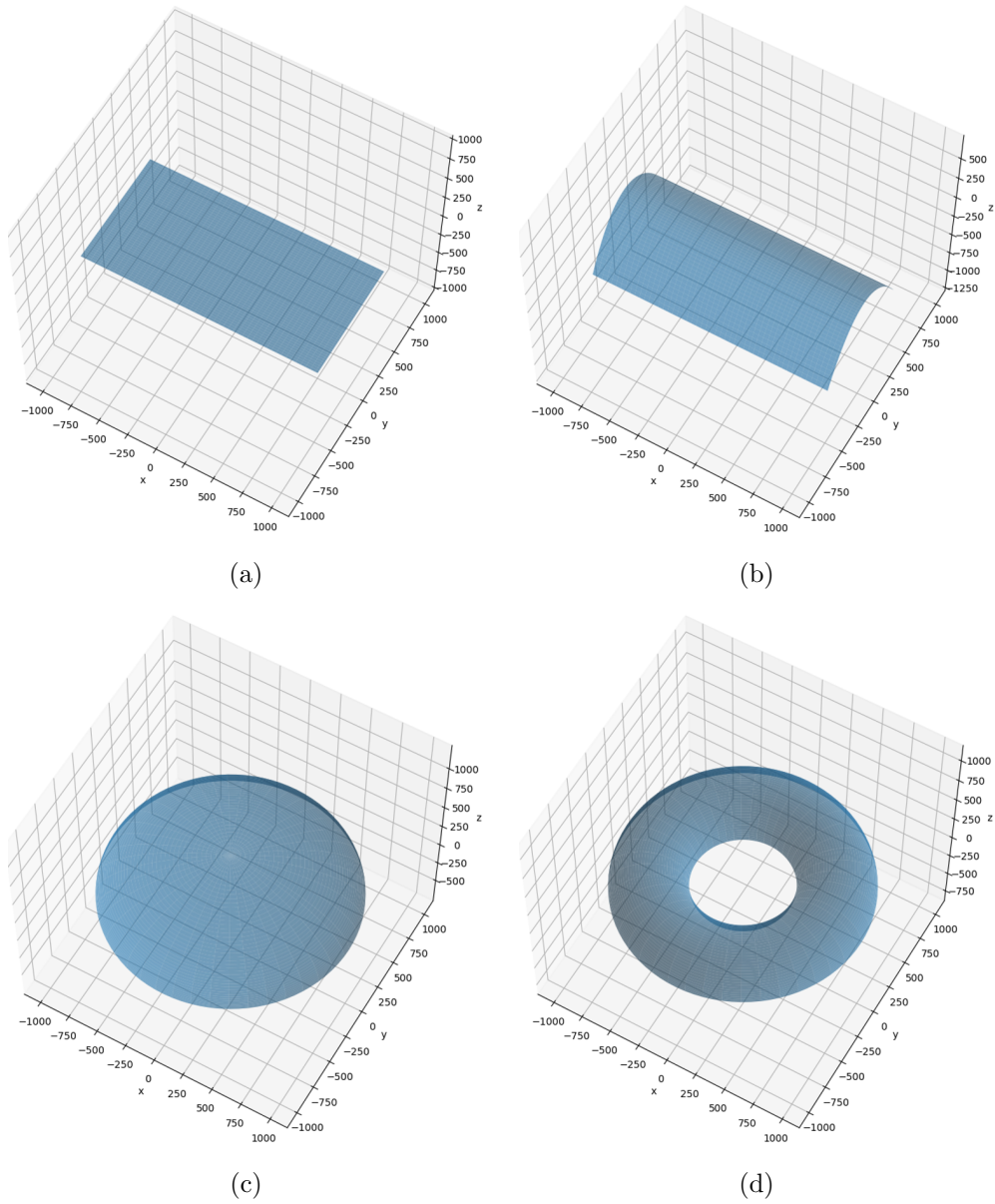


Figure 4.1: Four types of surface used in simulation (a) flat plane (b) quadratic plane (c) half ellipsoid (d) half torus

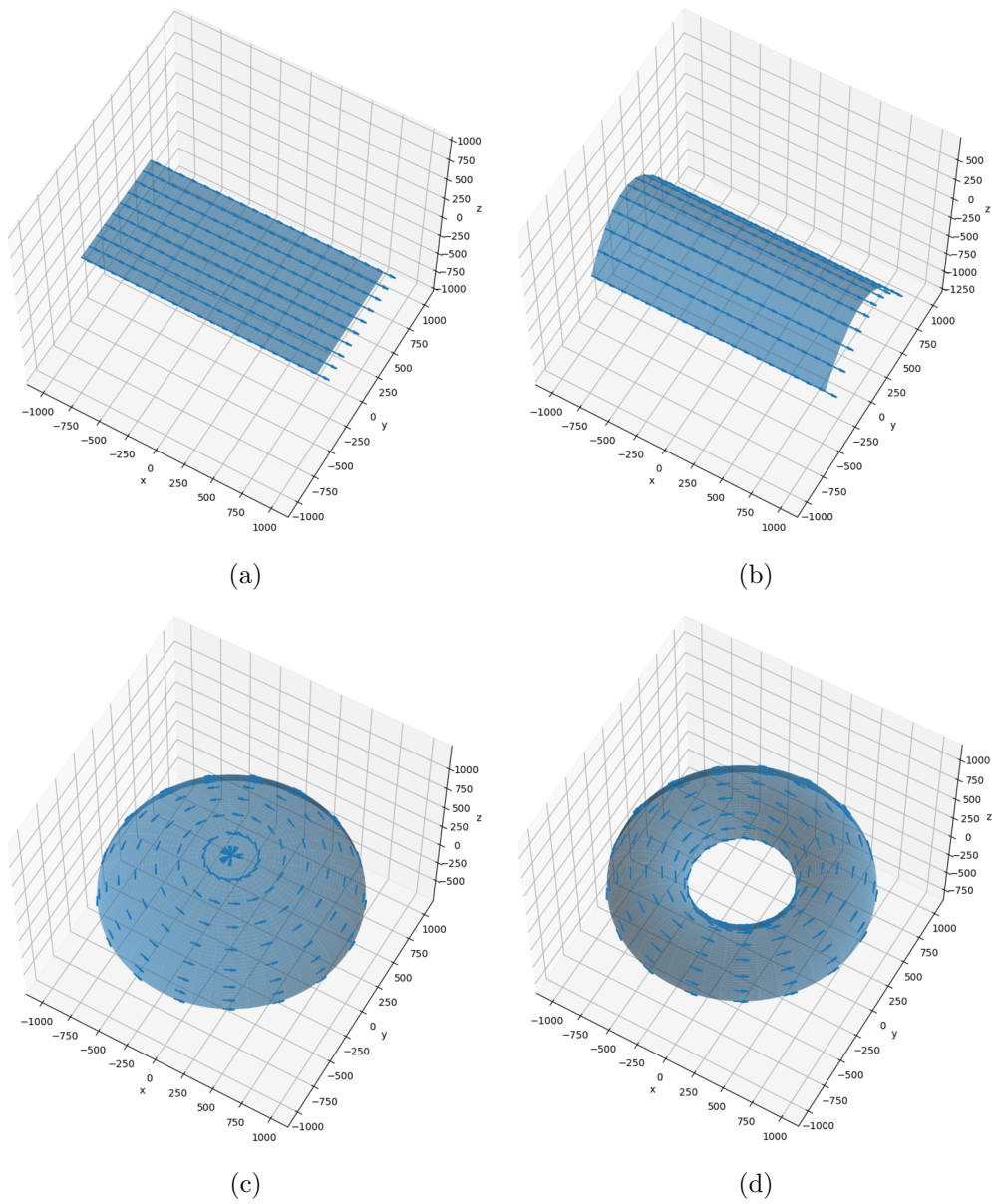


Figure 4.2: Representation of minimum curvature direction on each surface (a) flat plane (b) quadratic plane (c) half ellipsoid (d) half torus

4.2 Simulation Result

We first create the vector field in the direction of a minimum curvature on each surface (Figure 4.2). Next, we create a set of nodes and create a path by connecting the nodes in order.

Figure 4.3 shows the result of our CPP algorithm on each surface. Each node is annotated with a order of visit. The blue disks indicate the paint coverage at each node. It can be observed that the generated path is aligned with the given vector field.

Next, we determine the appropriate distribution model by learning the training data set. Then, we optimized the painting parameters utilizing the network output for uniform coverage. Figure 4.4 presents the results of the optimized paint deposition as viewed from the top.

4.3 Discussion

Figure 4.5 summarizes the performance of our method with respect to three features, upstream cost, standard deviation of the paint thickness, simulation time, and compare it with that of conventional sampling-based method. We can easily verify that the upstream cost associated with our method is much smaller, implying the path is well aligned with the given vector field. Moreover, it can be seen that the simulation time has also been greatly reduced, thanks to the sampling exclusion technique described in the chapter 3.1.1. Reasonable uniformity is also achieved, where the standard deviation of the paint thickness of our method is comparable to that of existing method.

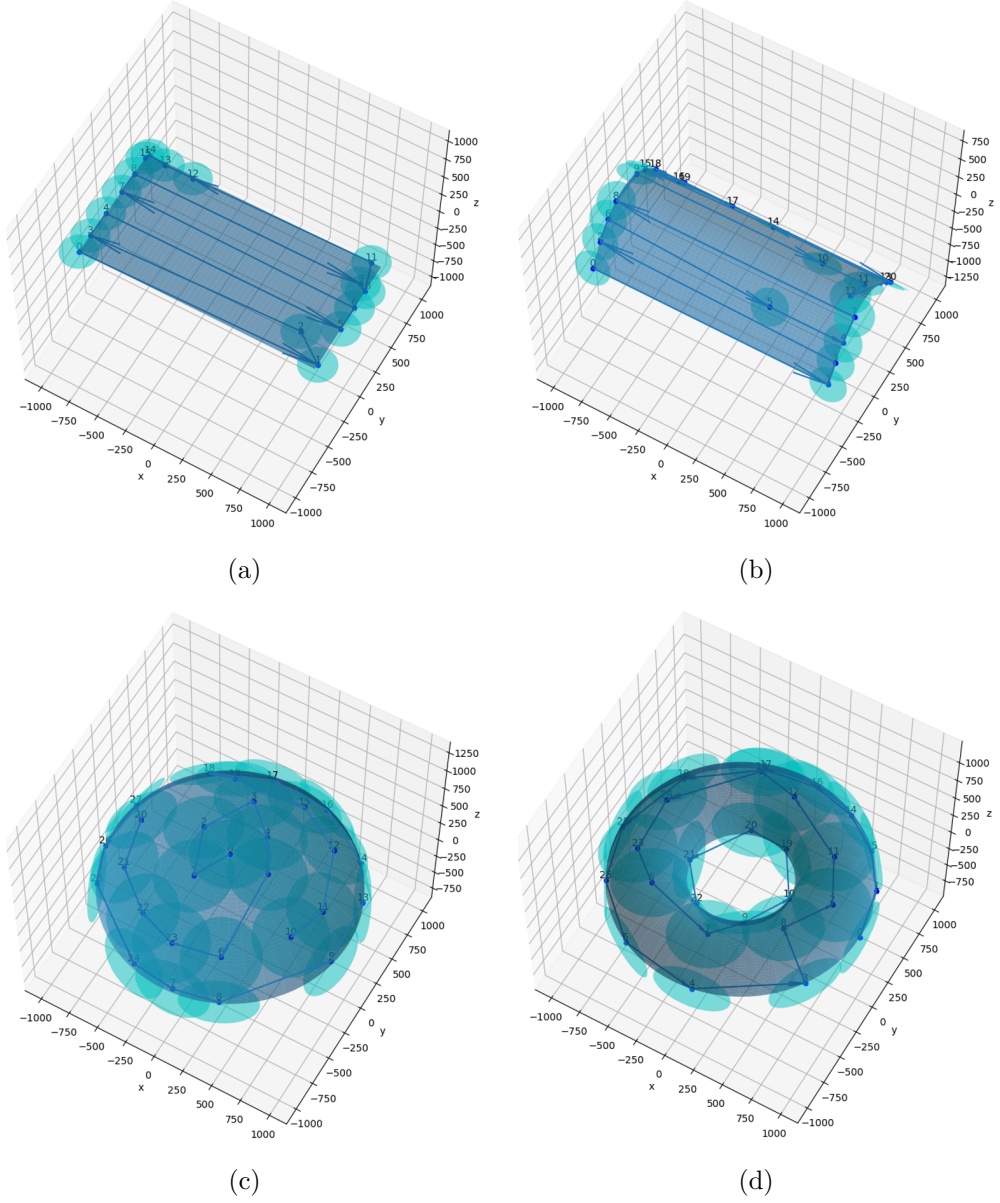


Figure 4.3: Simulation result of our Coverage Path Planning method on each surface (a) flat plane (b) quadratic plane (c) half ellipsoid (d) half torus

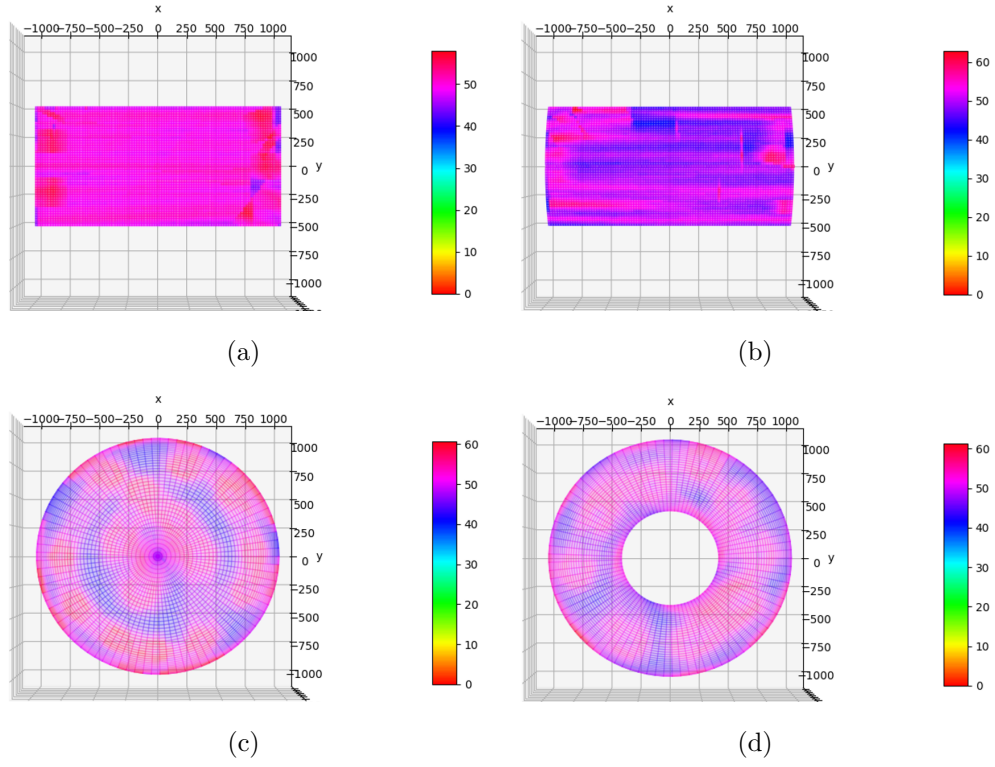
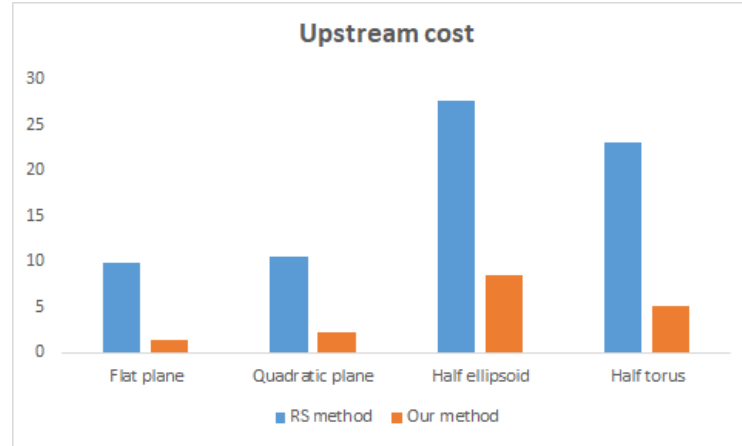
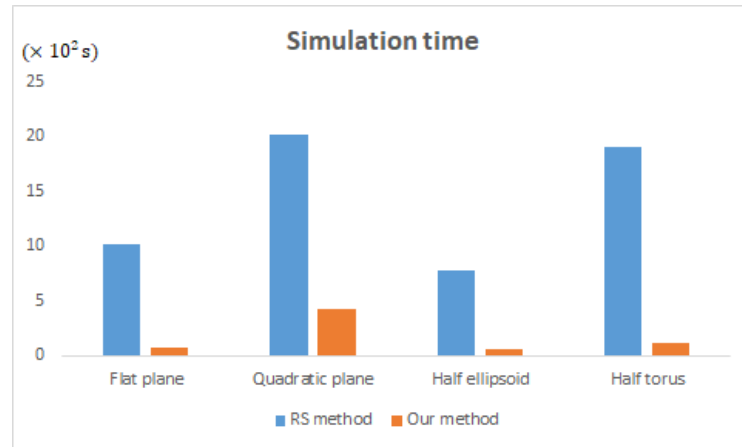


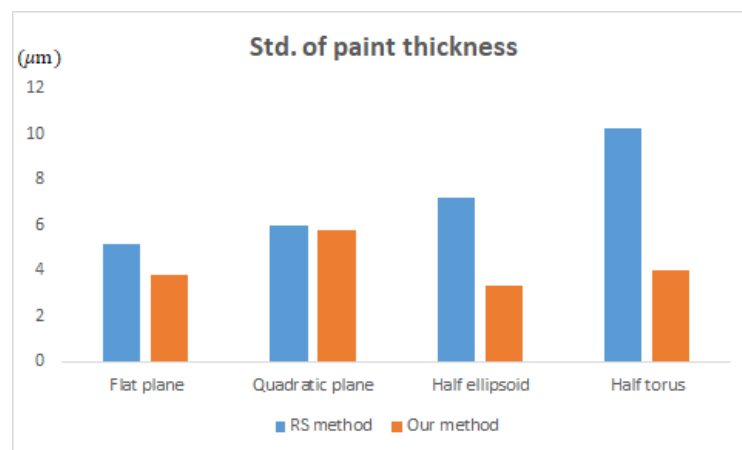
Figure 4.4: Simulation result of the optimized paint deposition on each surface (a) flat plane (b) quadratic plane (c) half ellipsoid (d) half torus



(a)



(b)



(c)

Figure 4.5: Performance of the conventional method and our method on each surface (a) upstream cost (b) simulation time (c) standard deviation of paint thickness

5

Conclusion

In this thesis, we propose an efficient UCPP algorithm based on a random sampling scheme for spray painting applications. We have improved on the previous versions of CPP algorithm by addressing two subproblems at once. We alternately iterate the path generation and node sampling steps and this method reduces the computational time by reducing the number of sampled nodes. In addition, we propose the concept of an upstream distance when solving the TSP. This induces more of the path to be aligned with a desired direction. Finally, we identify the paint distribution model by means of SVR network and optimized the painting parameters to achieve uniformity in the paint deposition. This machine learning technique enabled us to accurately predict the paint distribution model as a function of the painting parameters. We demonstrated our algorithm on several types of analytic surfaces and compared the results with those of conventional methods. The overall results indicate that our algorithm requires less computation time and

yields less upstream cost while achieving reasonable uniformity of the paint thickness. However, it should be noted that our algorithm is not a perfect method. Because our methods adopt a sampling-based approach, it has a disadvantage that the result varies with each simulation. We expect future improvements that resolve this drawback.

Bibliography

- [1] Enric Galceran and Marc Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous systems*, 61(12):1258–1276, 2013.
- [2] Prasad N Atkar, Aaron Greenfield, David C Conner, Howie Choset, and Alfred A Rizzi. Uniform coverage of automotive surface patches. *The International Journal of Robotics Research*, 24(11):883–898, 2005.
- [3] Mark Ollis and Anthony Stentz. Vision-based perception for an automated harvester. In *Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robot and Systems. Innovative Robotics for Real-World Applications. IROS'97*, volume 3, pages 1838–1844. IEEE, 1997.
- [4] M Farsi, K Ratcliff, JP Johnson, CR Allen, KZ Karam, and R Pawson. Robot control system for window cleaning. In *Proceedings of 1994 American Control Conference-ACC'94*, volume 1, pages 994–995. IEEE, 1994.
- [5] H Najjaran and N Kircanski. Path planning for a terrain scanner robot. In *INTERNATIONAL SYMPOSIUM ON ROBOTICS*, volume 31, pages 132–137. unknown, 2000.
- [6] Ali Ahmadzadeh, James Keller, George Pappas, Ali Jadbabaie, and Vijay Kumar. An optimization-based approach to time-critical cooperative surveillance and coverage with uavs. In *Experimental robotics*, pages 491–500. Springer, 2008.
- [7] Brendan J Englot and Franz S Hover. Sampling-based coverage path planning for inspection of complex structures. In *Twenty-Second International Conference on Automated Planning and Scheduling*, 2012.

- [8] Howie Choset, Sean Walker, Kunyayut Eiamsa-Ard, and Joel Burdick. Sensor-based exploration: Incremental construction of the hierarchical generalized voronoi graph. *The International Journal of Robotics Research*, 19(2):126–148, 2000.
- [9] Inyoung Ko, Beobkyoon Kim, and Frank Chongwoo Park. Randomized path planning on vector fields. *The International Journal of Robotics Research*, 33(13):1664–1682, 2014.
- [10] Wei Chen, Hao Liu, Yang Tang, and Junjie Liu. Trajectory optimization of electrostatic spray painting robots on curved surface. *Coatings*, 7(10):155, 2017.
- [11] Amir Pirzadeh and Wesley Snyder. A unified solution to coverage and search in explored and unexplored terrains using indirect control. In *Proceedings., IEEE International Conference on Robotics and Automation*, pages 2113–2119. IEEE, 1990.
- [12] Yoav Gabriely and Elon Rimon. Spanning-tree based coverage of continuous areas by a mobile robot. *Annals of mathematics and artificial intelligence*, 31(1-4):77–98, 2001.
- [13] Howie Choset. Coverage of known spaces: The boustrophedon cellular decomposition. *Autonomous Robots*, 9(3):247–253, 2000.
- [14] Christian Hofner and Günther Schmidt. Path planning and guidance techniques for an autonomous mobile cleaning robot. *Robotics and autonomous systems*, 14(2-3):199–212, 1995.

- [15] Tim Danner and Lydia E Kavraki. Randomized planning for short inspection paths. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 2, pages 971–976. IEEE, 2000.
- [16] PW Tse, S Lang, KC Leung, and HC Sze. Design of a navigation system for a household mobile robot using neural networks. In *1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98CH36227)*, volume 3, pages 2151–2156. IEEE, 1998.
- [17] Yili Fu and Sherman YT Lang. Fuzzy logic based mobile robot area filling with vision system for indoor environments. In *Proceedings 1999 IEEE International Symposium on Computational Intelligence in Robotics and Automation. CIRA'99 (Cat. No. 99EX375)*, pages 326–331. IEEE, 1999.
- [18] Timo Oksanen and Arto Visala. Coverage path planning algorithms for agricultural field machines. *Journal of field robotics*, 26(8):651–668, 2009.
- [19] Howie Choset, Ercan Acar, Alfred A Rizzi, and Jonathan Luntz. Exact cellular decompositions in terms of critical points of morse functions. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 3, pages 2270–2277. IEEE, 2000.
- [20] Brendan Englot and Franz S Hover. Three-dimensional coverage planning for an underwater inspection robot. *The International Journal of Robotics Research*, 32(9-10):1048–1073, 2013.

- [21] Georgios Papadopoulos, Hanna Kurniawati, and Nicholas M Patrikalakis. Asymptotically optimal inspection planning using systems with differential constraints. In *2013 IEEE International Conference on Robotics and Automation*, pages 4126–4133. IEEE, 2013.
- [22] Esther M Arkin, Sándor P Fekete, and Joseph SB Mitchell. Approximation algorithms for lawn mowing and milling. *Computational Geometry*, 17(1-2):25–50, 2000.
- [23] Brendan Englot and Franz Hover. Planning complex inspection tasks using redundant roadmaps. In *Robotics Research*, pages 327–343. Springer, 2017.
- [24] Léonard Jaillet, Juan Cortés, and Thierry Siméon. Sampling-based path planning on configuration-space costmaps. *IEEE Transactions on Robotics*, 26(4):635–646, 2010.
- [25] Weihua Sheng, Ning Xi, Mumin Song, Yifan Chen, and Perry MacNeille. Automated cad-guided robot path planning for spray painting of compound surfaces. In *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)(Cat. No. 00CH37113)*, volume 3, pages 1918–1923. IEEE, 2000.
- [26] Eckhard Freund, Dirk Rokossa, and Jürgen Roßmann. Process-oriented approach to an efficient off-line programming of industrial robots. In *IECON’98. Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society (Cat. No. 98CH36200)*, volume 1, pages 208–213. IEEE, 1998.

- [27] Ramanujam Ramabhadran and John K Antonio. Fast solution techniques for a class of optimal trajectory planning problems with applications to automated spray coating. *IEEE Transactions on Robotics and Automation*, 13(4):519–530, 1997.
- [28] MA Sahir Arıkan and Tuna Balkan. Process modeling, simulation, and paint thickness measurement for robotic spray painting. *Journal of Robotic Systems*, 17(9):479–494, 2000.
- [29] Prasad N Atkar, David C Conner, Aaron Greenfield, Howie Choset, and Alfred A Rizzi. Hierarchical segmentation of piecewise pseudoextruded surfaces for uniform coverage. *IEEE Transactions on Automation Science and Engineering*, 6(1):107–120, 2008.
- [30] Thomas F Banchoff and Stephen T Lovett. *Differential geometry of curves and surfaces*. CRC Press, 2016.
- [31] Barrett O’neill. *Elementary differential geometry*. Elsevier, 2006.
- [32] Gilbert Laporte. The traveling salesman problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(2):231–247, 1992.
- [33] Gregory Gutin and Abraham P Punnen. *The traveling salesman problem and its variations*, volume 12. Springer Science & Business Media, 2006.
- [34] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.
- [35] Alex J Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, 2004.

국문초록

본 논문에서는 2차원 표면의 균일 커버리지 경로 계획을 설명하고 이를 효율적으로 푸는 알고리즘을 제시한다. 우리는 경로 계획 문제를 두 개의 하위 문제로 분리하여 각각 푸는 기존의 방식을 보완하여 두 개의 하위문제를 한 번에 풀면서 계산시간을 줄이는 방법을 제시하였다. 또한 경우에 따라 주어진 벡터 필드와 나란한 방향으로 경로가 생성될 필요가 있는데 이를 위해 거스름 거리(upstream distance)의 개념을 제시하였으며 여행 외판원 문제(Traveling Salesman Problem)를 풀 때 이를 적용하였다. 우리는 차량 도장 응용분야에 균일 커버리지 경로 계획법을 적용하였으며 도장 시스템을 고려하여 균일한 페인트 두께를 보장하는 방법을 같이 제시하였다. 네 가지 타입의 2차원 곡면에 대해 시뮬레이션을 진행하였으며 기존의 방법에 비해 더 적은 계산시간을 요구하면서도 합리적인 수준의 페인트 균일도를 달성함을 검증하였다.

주요어: 균일 커버리지 경로 계획, 스프레이 도장, 여행 외판원 문제, 거스름 척도, 서포트 벡터 회기법, 페인트 분포 모델

학번: 2018-24497