Sampling-Based Roadmap Methods for a Visual Reconnaissance UAV*

Karl J. Obermeyer†
University of California at Santa Barbara, CA, 93106

Paul Oberlin‡and Swaroop Darbha§
Texas A&M University, College Station, TX 77843

This article considers a path planning problem for a single fixed-wing aircraft performing a reconnaissance mission using EO (Electro-Optical) camera(s). A mathematical formulation of the general aircraft visual reconnaissance problem for static ground targets in terrain is given and it is shown, under simplifying assumptions, that it can be reduced to what we call the *PVDTSP (Polygon-Visiting Dubins Traveling Salesman Problem)*, a variation of the famous TSP (Traveling Salesman Problem). Two algorithms are developed to solve the PVDTSP. They fall into the class of algorithms known as *sampling-based roadmap methods* because they operate by sampling a finite set of points from a continuous state space in order to reduce a continuous motion planning problem to planning on a finite discrete graph. The first method is *resolution complete*, which means it provably converges to a nonisolated global optimum as the number of samples grows. The second method achieves slightly shorter computation times by using approximate dynamic programming, but consequently is only guaranteed to converge to a nonisolated global optimum modulo target order. Numerical experiments indicate that, for up to about 20 targets, both methods deliver good solutions suitably quickly for online purposes. Additionally, both algorithms allow trade-off of computation time for solution quality and are shown extensible to handle wind, airspace constraints, any vehicle dynamics, and open-path (vs. closed-tour) problems.

I. Introduction

UAVs (Unmanned Air Vehicles) are increasingly being used for both civilian and military applications such as environmental monitoring, geological survey, surveillance, reconnaissance, and search and rescue.[1,2] Good control and planning algorithms are a key component of UAV technology because they can increase operational capabilities while reducing risk, costs, and operator workloads. In this article we present novel path planning algorithms for a single fixed-wing aircraft performing a reconnaissance mission using EO (Electro-Optical) camera(s). Given a set of stationary ground targets in a terrain (natural, urban, or mixed), the objective is to compute a path for the reconnaissance aircraft so that it can photograph all targets in minimum time. That the targets are situated in terrain plays a significant role because terrain features can occlude visibility. As a result, in order for a target to be photographed, the aircraft must be located where both (1) the target is in close enough range to satisfy the photograph's resolution requirements, and (2) the line-of-sight between the aircraft and the target is not blocked by terrain. For a given target, we call the set of all such aircraft positions the target's *visibility region*. An example visibility region is illustrated in Fig. 1. In full generality, the aircraft path planning can be complicated by wind, airspace constraints (e.g. due to enemy threats or collision avoidance), aircraft dynamic constraints, and the aircraft body itself occluding visibility. However, under simplifying assumptions, if we model the aircraft as a Dubins vehicle[a], approximate the targets' visibility regions by polygons, and let the path be a closed tour, then the reconnaissance path planning problem can be reduced to the following.

> For a Dubins vehicle, find a shortest planar closed tour which visits at least one point in each of a set of polygons.

---

†PhD student, Center for Control, Dynamical Systems, and Computation; karl@engr.ucsb.edu. Student Member AIAA.
‡PhD student, Department of Mechanical Engineering; paul.v.oberlin@gmail.com
§Professor, Department of Mechanical Engineering; dswaroop@tamu.edu
[a]A Dubins vehicle is one which moves only forward and has a minimum turning radius.[3,4]

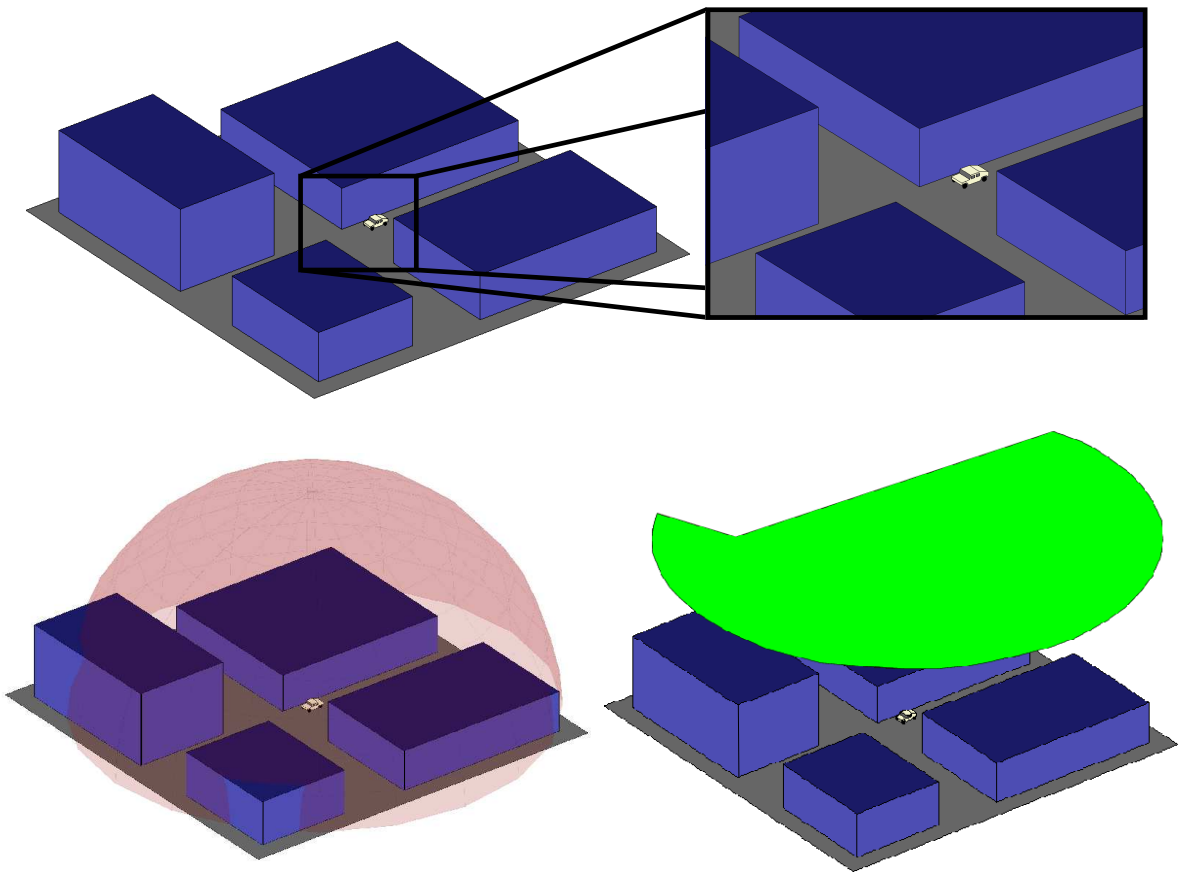

Figure 1. Top is shown an example target, a ground vehicle parked next to a building in urban terrain. The set of all points which are close enough to the target to satisfy photograph resolution requirements is a solid sphere (bottom left). The green two-dimensional region in the sky (bottom right) shows the subset of the sphere, at a reconnaissance aircraft's altitude h, where target visibility is not occluded by terrain. Assuming the aircraft body itself doesn't occlude visibility, then flying the aircraft through the green region is sufficient for the target to be photographed, hence we call it the target's *visibility region* for fixed aircraft altitude h.

We refer to this henceforth as the *PVDTSP (Polygon-Visiting Dubins Traveling Salesman Problem)* since it is a variation of the famous *TSP (Traveling Salesman Problem)*.[b] A graphical illustration of the PVDTSP is shown in Fig. 2.

To our knowledge the PVDTSP has not previously been studied aside from Ref. 6 where we designed a genetic algorithm. Athough the genetic algorithm performed well in a Monte-Carlo numerical study, there unforntunately are no proven performance guarantees. Because the PVDTSP has embedded in it the combinatorial problem of choosing the order to visit the polygons, the solution space is very large and discontinuous. This precludes direct application of numerical optimal control techniques traditionally used in trajectory optimization, surveyed, e.g., in Ref. 7. However, several related variations of the TSP are of interest. The *ETSP (Euclidean TSP)* is a TSP where the vertices of the graph are points in the Euclidean plane $\mathbb{R}^2$ and the edges are weighted with Euclidean distances. In the *ETSPN (Euclidean TSP with Neighborhoods)* one seeks a shortest closed Euclidean path passing through n subsets of the plane. The ETSP is NP-hard[8] and so is the ETSPN by virtue of being a generalization of the ETSP. The *DTSP (Dubins TSP)*, where a Dubins vehicle must follow a shortest tour through n single point targets in the plane, is known to be NP-hard in n.[9] Various heuristics for both single and multi-vehicle versions of the DTSP can be found, e.g., in Ref. 10, 11, and 12. The PVDTSP reduces to the ETSPN in the limit as the vehicle's minimum turning radius becomes small compared to the distances between polygons. Similarly, as the area of the polygons goes to zero, the PVDTSP reduces to the DTSP, hence the PVDTSP is NP-hard. There

[b]The TSP, one of the most famous NP-hard problems of combinatorial optimization, is to find a minimum cost tour (cyclic path) through a weighted graph such that every vertex is visited exactly once. If the graph is directed, it is called the ATSP (Asymmetric TSP). See, e.g., Ref.[5]

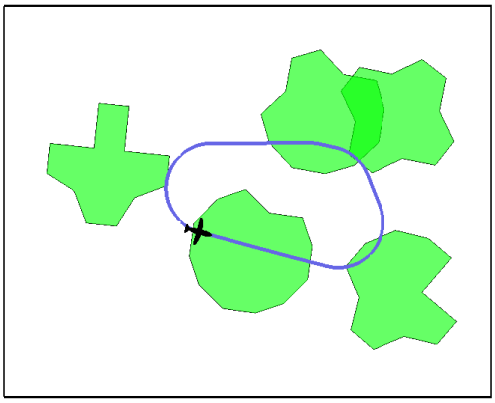

Figure 2. Example problem instance and candidate solution path for the *PVDTSP* **(***Polygon-Visiting Dubins Traveling Salesman Problem***). In order to photograph all targets, the aircraft must fly through at least one point in each target's visibility region (green), cf. Fig. 1.**

exist a number of algorithms with approximation guarantees for both the DTSP[13–15] and ETSPN,[16–18] but it appears that extending any of these algorithms to the PVDTSP would put undesirable restrictions on the problem instances which could be handled, e.g., the polygons would not be allowed to overlap. The *FOTSP (Finite One-in-set TSP)*[c] is the problem of finding a closed path of minimum cost which passes through at least one vertex in each of a finite collection of *clusters*, the clusters being mutually exclusive finite vertex sets. The FOTSP is NP-hard because it has as a special case the *ATSP (Asymmetric TSP)*.[5] An FOTSP instance can be solved exactly by transforming it into an ATSP instance using the *Noon-Bean transformation* from Ref. 19, then invoking an ATSP solver. Alternatively, an FOTSP can be solved using an approximate dynamic programming technique as in Ref. 20. In the robotics literature,[21,22] a *sampling-based roadmap method*[d] refers to any algorithm which operates by sampling a finite set of points from a continuous state space in order to reduce a continuous motion planning problem to planning on a finite discrete graph. Sampling-based roadmap methods have traditionally only been used for collision-free point-to-point path planning amongst obstacles, however, in Ref. 23 approximate solutions to the DTSP are found by sampling discrete sets of orientations that the Dubins vehicle can have over each target, essentially approximating a DTSP instance by an FOTSP instance. The Noon-Bean transformation is then used to convert the FOTSP instance into an ATSP instance so that a standard ATSP solver can be applied. Discretization of the vehicle state space in order to approximate the original problem by an FOTSP is a key idea which we build upon in designing sampling-based roadmap methods for the PVDTSP in the present work.

There are three main contributions in this article. First, we precisely formulate the general aircraft visual reconnaissance problem for static ground targets in terrain. Under simplifying assumptions, we reduce our general formulation to the PVDTSP. Although the PVDTSP reduces to the well-studied DTSP and ETSP in the *sparse limit* as targets are very far apart and minimum turning radius is small, we provide a worst-case analysis demonstrating the importance of developing specialized algorithms for the PVDTSP in the *dense limit* as targets are close together and polygons may overlap significantly. An early version of the PVDTSP formulation appeared in our previous work Ref. 6, but that did not include the worst-case analysis. Our second contribution is the design and numerical study of two sampling-based roadmap methods for the PVDTSP. These methods operate by sampling finite discrete sets of vehicle states to approximate a PVDTSP instance by an FOTSP instance, then applying existing FOTSP solving techniques. One of our sampling-

[c]What we have chosen to call the FOTSP is known variously in the literature as "Group-TSP", "Generalized-TSP", "One-of-a-Set TSP", "Errand Scheduling Problem", "Multiple Choice TSP", "Covering Salesman Problem", or "International TSP".

[d]In this usage, "method" means a high level algorithm having multiple components, each of which may be considered an algorithm in its own right.

based roadmap methods uses the Noon-Bean transformation from Ref. 19 and is *resolution complete*, which means it provably converges to a nonisolated global optimum as the number of samples grows. Our other sampling-based roadmap method achieves faster computation times by using the approximate dynamic programming technique from Ref. 20, but consequently only converges to a nonisolated global optimum modulo target order. While we have borrowed the idea of approximation by an FOTSP from Ref. 23, the present work goes beyond a simple extension in that we (1) illustrate the connection with sampling-based roadmap methods used for path planning in the robotics literature[e], (2) use a novel sampling technique to reduce computational time complexity, and (3) provide proof of convergence to nonisolated global optima. Numerical experiments indicate that both sampling-based roadmap methods deliver good solutions suitably quickly for online purposes when applied to PVDTSP instances having up to about 20 targets. For problem instances with greater than 5 targets the sampling-based roadmap methods significantly outperformed the genetic algorithm in Ref. 6. Additionally, both methods have a means for a user to trade off computation time for solution quality. Our third contribution is to describe how the modular nature of both the algorithms allows them to easily be extended to handle wind, airspace constraints, any vehicle dynamics, and open-path (vs. closed-tour) problems.

This article is organized as follows. In Sec. II we introduce notation, mathematically formulate the minimum time reconnaissance aircraft path planning problem, show how to reduce the problem to a PVDTSP, and provide the worst-case analysis motivating the development of specialized PVDTSP algorithms. In Sec. III we present, analyze, and numerically validate the sampling-based roadmap methods. Finally, we describe how our algorithms can be extended in Sec. IV and conclude in Section V.

II. Mathematical Formulation

We begin with some preliminary notation. The s-dimensional Euclidean space is $\mathbb{R}^s$ and $\mathbb{S}$ is the circle parameterized by angle radians ranging from 0 to 2π, 0 and 2π identified. Let $\mathcal{T} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_n\}$ be the set of n targets which must be photographed by our aircraft. Given a set A, we denote its cardinality by $|A|$, its interior by A°, and its power set, i.e., the set of all subsets of A, by 2^A. Given two sets A and B, $A \times B$ is the Cartesian product of these sets. The complete state of our reconnaissance aircraft is encoded in a vector $\mathbf{x}$, which takes a value in the aircraft's state space X. We can segregate $\mathbf{x}$ into internal and external states so that

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_{\text{internal}} \\ \mathbf{x}_{\text{external}} \end{bmatrix} \in X = X_{\text{internal}} \times X_{\text{external}}. \tag{1}$$

The internal state $\mathbf{x}_{\text{internal}}$ accounts for control surface states, and more importantly, if the aircraft has gimbaled camera(s), then also for the camera state(s). The external state $\mathbf{x}_{\text{external}}$ accounts for the aircraft body position and velocity in the full six degrees of freedom.

We now define a map $\mathcal{V} : \mathcal{T} \to 2^X$ from the set of targets to subsets of the aircraft state space. Under this map, $\mathcal{V}(\mathcal{T}_i) \subset X$, called the ith target's *visibility region*, is precisely the set of all aircraft states such that $\mathcal{T}_i$ can be photographed whenever the aircraft is in that state. Later, in Sec. II.A, we discuss how to calculate visibility regions from a terrain model, but let us assume for now we can make this calculation. We also assume a BVP (Boundary Value Problem) solver is available which calculates the minimum time aircraft trajectory between any two states $\mathbf{x}$ and $\mathbf{x}'$, provided a trajectory exists. We treat this minimum time between states as a "black box" distance function denoted by $d(\mathbf{x}, \mathbf{x}')$. Now our **minimum-time reconnaissance path planning problem** can be stated as

$$\begin{aligned} \text{Minimize :} \quad & C(\mathbf{x}_1, \ldots, \mathbf{x}_n) \;=\; \textstyle\sum_{i=1}^{n-1} d(\mathbf{x}_i, \mathbf{x}_{i+1}) + d(\mathbf{x}_n, \mathbf{x}_1) \\ \text{Subject To :} \quad & \text{for each } i \in \{1, \ldots, n\} \text{ there exists } j \in \{1, \ldots, n\} \\ & \text{such that } \mathbf{x}_j \in \mathcal{V}(\mathcal{T}_i), \end{aligned} \tag{2}$$

where the decision variables are the states $\mathbf{x}_i$ $(i = 1, \ldots, n)$. Once an optimal sequence of states $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ has been chosen, then the minimum time state-to-state trajectory planner can be used to connect each pair of consecutive states, thus we obtain a minimum time closed reconnaissance tour. Since the complete state

[e]Although Ref. 23 appears to be the first application of a sampling-based roadmap method to a TSP-type problem, they do not use the term "sampling-based roadmap method", nor is there any mention of the connection with sampling-based roadmap methods in the robotics literature.

space of an aircraft can be very complicated, we simplify the discussion by making the following **main assumptions**.

(i) The aircraft is modeled as a Dubins vehicle with minimum turning radius $r_{\min}$, fixed altitude h, and constant airspeed V_{a}.
Comments: Common for small low-power UAVs.

(ii) Regardless of state, the aircraft body never occludes visibility between the camera and a target.
Comments: Holds when either there are multiple cameras covering all angles from the aircraft, or there is a sufficiently flexible gimbaled camera with dynamics faster than the aircraft body dynamics.[f]

(iii) There are no airspace constraints nor wind.
Comments: As to be discussed in Sec. IV, our results can easily be extended to handle wind and no-fly zones.

In accordance with assumption (i), the aircraft dynamics take the form

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} V_{\mathrm{a}} \sin(\psi) \\ V_{\mathrm{a}} \cos(\psi) \\ u \end{bmatrix}, \tag{3}$$

where $(x, y) \in \mathbb{R}^2$ are earth-fixed Cartesian coordinates, $\psi \in \mathbb{S}$ is the azimuth angle, and u is the input to an autopilot system. Assumption (ii) tells us that a target can be photographed independent of aircraft azimuth ψ, therefore we can abstract out $\mathbf{x}_{\mathrm{internal}}$ so that the aircraft state space is reduced to

$$\mathbf{x} = (x, y, \psi) \ \in \ X = \mathbb{R}^2 \times \mathbb{S} = \mathrm{SE}(2), \tag{4}$$

and the Visibility sets $\mathcal{V}(\mathcal{T}_1), \ldots, \mathcal{V}(\mathcal{T}_n)$ are reduced to 2-dimensional regions in $\mathbb{R}^2$ as shown in Fig. 1 and 2 (as opposed to subsets of $X = \mathbb{R}^2 \times \mathbb{S}$). Hereinafter we refer to the state of a Dubins vehicle interchangeably as "state" or "pose" (position with orientation). The minimum time path between two Dubins states $\mathbf{x}$ and $\mathbf{x}'$ can be computed very quickly in constant time.[3,24] This provides us with our "black box" distance function $d(\mathbf{x}, \mathbf{x}')$ as it appears in the optimization problem Eq. 2. Although visibility regions may contain circular arcs due to the camera range constraint, they can be well approximated by polygons. We have now reduced our minimum time reconnaissance path planning problem to a PVDTSP.

In some UAV systems in the field today, target visibility sets are neglected and reconnaissance paths are planned by simply solving the DTSP over the target positions, i.e., the UAV is restricted to pass directly over each target in order to photograph it. However the worst-case analysis in the following Theorem II.1 demonstrates that an arbitrarily large relative cost increase can be incurred by solving the DTSP instead of the PVDTSP. This cost increase is most pronounced in the *dense limit* (left in Fig. 3) as targets become very close together, which motivates our development of specialized PVDTSP algorithms for tight urban scenarios especially. In contrast, in the *sparse limit* (right in Fig. 3) when the minimum turning radius and visibility set diameters are much smaller than the distances between targets, there is no significant advantage to solving the PVDTSP over the DTSP nor over the ETSP.

Theorem II.1 (DTSP vs. PVDTSP Worst-Case Analysis)**.** *In a fixed compact subset of the plane $\mathbb{R}^2$, solving the DTSP over point targets instead of the PVDTSP over those same targets' visibility sets may incur a cost penalty of order $\Omega(n)$ in the worst case.*[g]

Proof. The set of all DTSP tours through n point targets is a subset of all PVDTSP tours through those same targets' visibility sets, therefore the length of a tour that results from solving the PVDTSP to optimality can be no greater than that of solving the DTSP. Now it suffices to prove the theorem by demonstrating a class of visual reconnaissance problem instances, parameterized by the number of targets n, for which the tour cost when solved as a DTSP is order $\Omega(n)$ (lower bounded) yet only order $O(1)$ (upper bounded) when solved as a PVDTSP. One such class of instances is illustrated left in Fig. 3.[h] Given any n noncolinear point targets in the plane, we can linearly scale them until the radius of the circle constructed from any three of

[f] An omnidirectional camera is another possibility, but they typically have poor resolution.

[g] A function $f(n)$ is said to be $\Omega(n)$ if there exist positive constants c and n_0 such that $f(n) \geq cn$ for all $n \geq n_0$.

[h] Such a class of instances has been used previously in Ref. 14 to show DTSP tours in general have worst-case length $\Omega(n)$.

them has radius smaller than the Dubins vehicle minimum turn radius. This scaling ensures that, in order to fly a feasible DTSP tour, the aircraft must travel a distance at least the length of one minimum turn radius circle for every two targets. Solving the DTSP over these points would thus cost $\Omega(n)$, yet letting the intersection of the targets visibility sets' contain all the targets, the PVDTSP could be solved with a single minimum turn radius loop and thus cost only $O(1)$. □

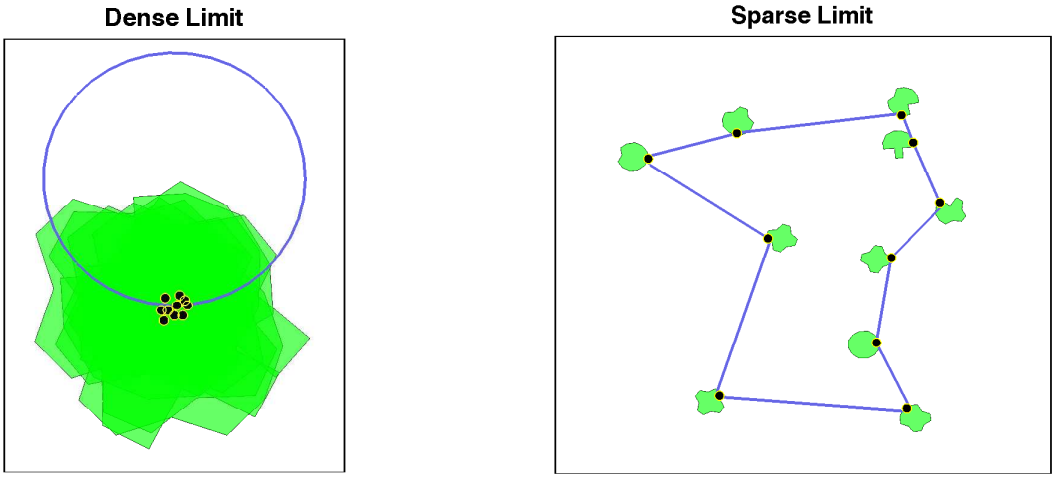

Figure 3. In the *dense limit* (left) as the distances between targets are much smaller than the minimum turning radius, there can be a large penalty incurred ($\Omega(n)$, see Theorem II.1 and proof) by solving the DTSP instead of the PVDTSP. In particular, if the densely packed targets are sufficiently noncolinear, an aircraft solving the PVDTSP can photograph all targets in a single pass (shown as blue circle), but an aircraft solving the DTSP would only be able to photograph two targets per pass, thus requiring a tour at least the length of $\frac{n}{2}$ minimum turn radius circles. In the *sparse limit* (right) when the minimum turning radius and visibility set diameters are much smaller than the distances between targets, there is no significant advantage to solving the PVDTSP over the DTSP nor over the ETSP.

II.A. Calculating Visibility Regions

In order to calculate the visibility region $\mathcal{V}(\mathcal{T}_i)$ of a target, it is necessary to know the target location and to have a computer model/representation of the terrain. This representation may be either a vector format, e.g., a TIN (Triangulated Irregular Network), or a raster format, e.g., a DEM (Digital Elevation Map) such as the military's DTED (Digital Terrain Elevation Data). The necessary data to build a terrain model could be gathered, e.g., by LIDAR (LIght Detection And Ranging), SAR (Synthetic Aperture Radar), or photogrammetry. Once the terrain model has been built, the visibility region of a target may be calculated using a "sweeping algorithm"[25–27] in the vector case, or Bresenham's line algorithm[28] in the raster case.

III. Sampling-Based Roadmap Methods

In this section we present two sampling-based roadmap methods for the PVDTSP. These methods operate by sampling a finite discrete set of poses from the continuous Dubins state space in order to approximate the PVDTSP instance by an FOTSP instance, then applying an FOTSP algorithm. We call the approximating FOTSP instance a *PVDTSP roadmap*. In Sec. III.A we explain in detail how to construct a PVDTSP roadmap. The methods that use the roadmap are then described in Sec. III.B and III.C. We provide a numerical study in Sec. III.D and in Sec. III.E describe the relationship between the sampling-based roadmap methods in the present work and those used in the robotics literature for collision-free path planning. Later in Sec. IV we explain how our methods can be extended to handle wind, airspace constraints, any vehicle dynamics, and open-path problems.

III.A. Roadmap Construction

To define a PVDTSP roadmap we first need definitions of the ATSP (Asymmetric TSP) and FOTSP (Finite One-in-a-set TSP) which are more precise than those given in Sec. I.

Definition III.1 (ATSP)**.** *Given a weighted directed graph $\mathcal{G} = (V, E)$ where V is a finite set of vertices*

$$\{v_1, v_2, v_3, \ldots, v_{n_V}\}$$

and E a set of directed edges with weights

$$\{w_{i,j} | i, j \in \{1, 2, 3, \ldots, n_V\} \text{ and } i \neq j\},$$

the Asymmetric TSP (ATSP) *is to find a directed cycle of minimum cost which visits every vertex in V exactly once.*

Definition III.2 (FOTSP)**.** *Suppose we have a weighted directed graph $\mathcal{G} = (V, E)$ as in Def. III.1, but now the vertices are partitioned into finitely many nonempty mutually exclusive vertex sets called* clusters *$S = \{S_1, S_2, S_3, \ldots, S_{n_S}\}$, so that the vertices can be written as*

$$\begin{aligned} V = \Big\{ & v_{(1,1)}, v_{(1,2)}, v_{(1,3)}, \ldots, v_{(1,n_{S_1})}, v_{(2,1)}, v_{(2,2)}, v_{(2,3)}, \ldots, v_{(2,n_{S_2})}, \ldots \\ & \ldots, v_{(n_S,1)}, v_{(n_S,2)}, v_{(n_s,3)}, \ldots, v_{(n_S,n_{S_{n_S}})} \Big\}. \end{aligned} \tag{5}$$

Then the Finite One-in-a-set TSP (FOTSP) *is to Find a directed cycle of minimum cost which visits at least one vertex from each cluster.*

Definition III.3 (PVDTSP Roadmap)**.** *A* roadmap *for a PVDTSP instance is an FOTSP instance, as per Def. III.2, where there is one cluster for each polygon. The vertices V are obtained by sampling a finite set of poses in each polygon and assigning them to the respective cluster. The edges E are obtained by making all possible inter-cluster connections using Dubins minimum time state-to-state distances as weights.*

We perform the pose sampling in Def. III.3 using what is known as a *quasirandom sequence*, although it could also be performed with a random sequence, uniform grid, or some heuristic. A quasirandom sequence is a deterministic sequence which densely fills a space and concurrently optimizes a generalized notion of resolution such as *dispersion*.[i] Given a set of samples in a metric space, the dispersion of that set is the radius of the largest empty ball. Although there are many different quasirandom sequences to choose from in the literature,[21,22,29] we have chosen to use *Halton* sequences for simplicity, efficiency, and because they (1) are asymptotically optimal with respect to dispersion, (2) have, with high probability, better dispersion than uniform random sampling, and (3) allow more flexibility in the number of samples than a regular grid. Halton sequences are defined formally as follows.

Definition III.4 (Halton Sequence[30])**.** *Let $b_1, \ldots, b_s$ be coprime positive integers greater than 1. For each $j \in \{1, \ldots, s\}$, let the base b_j representation of an integer k be given by*

$$k = \sum_i a_{ij} b_j^i \qquad (a_{ij} \in \{0, 1, \ldots, b_j - 1\}).$$

Let

$$\Phi_{b_j}(k) := \sum_i a_{ij} b_j^{-(i+1)}.$$

*Then the s-*dimensional Halton sequence *$h_{(b_1,\ldots,b_{s-1})}(k) : \mathbb{N} \to [0, 1]^s$ is*

$$h_{(b_1,\ldots,b_s)}(k) = (\Phi_{b_1}(k), \Phi_{b_2}(k), \ldots, \Phi_{b_s}(k)).$$

A Halton sequence produces samples on the s-dimensional unit box in $\mathbb{R}^s$, for some s, so in order to use it for sampling tours, we must show how to map samples from a unit box onto poses in the polygons of a PVDTSP instance. The definitions and main convergence result Theorem III.9 to follow show that, without loss of generality, we may construct our PVDTSP roadmaps by sampling Halton points on a 2-dimensional unit box even though the full Dubins state space SE(2) is 3-dimensional. In particular, it suffices to map Halton points on the 2-dimensional unit box to *entry poses* of the polygons as shown in Fig. 4. By *entry pose of a polygon* we mean a pose which is positioned on the polygon's perimeter and either oriented towards the polygon's interior or parallel with its boundary.

[i] There exist other generalized notions of resolution in the literature, e.g., *discrepancy* is usually used in the Monte-Carlo integration context.[29]

Definition III.5 (Metric Space of Entry Poses $(X_{\text{entry}}, \rho_X)$)**.** *Let $X_{\text{entry}} \subset X = \text{SE}(2)$ be the 2-dimensional set of all Dubins states which correspond to an entry pose of some polygon in a PVDTSP instance. We endow X_{entry} with the metric ρ_X as follows. For any two points $\mathbf{x} = (x, y, \psi)$ and $\mathbf{x}' = (x', y', \psi')$ in X_{entry},*

$$\rho_X(\mathbf{x}, \mathbf{x}') := \underbrace{\sqrt{|x - x'|^2 + |y - y'|^2}}_{\text{Euclidean metric on } \mathbb{R}^2} + \underbrace{\min\{|\psi - \psi'|, 2\pi - |\psi - \psi'|\}}_{\text{geodesic distance on } \mathbb{S}}. \tag{6}$$

To achieve a prescribed dispersion on a bounded s-dimensional continuous space can require arbitrarily many more samples than to achieve the same dispersion on a codimension one surface. Constructing a PVDTSP roadmap by sampling on the 2-dimensional X_{entry} instead of the 3-dimensional X should therefore significantly reduce the computational time complexity of any method which uses the roadmap.

Definition III.6 (Halton Entry Pose Induced PVDTSP Roadmap $\mathcal{R}_i$)**.** *Let $\mathcal{R}_i$ denote a PVDTSP roadmap, as per Def. III.3, where the vertices are obtained from mapping the first i terms of the 2-dimensional Halton quasirandom sequence of bases 2 and 3 onto the space of entry poses as in Fig. 4.*

Definition III.7 (Metric Space of Dubins Tours $(\mathcal{D}, \rho_{\mathcal{D}})$)**.** *Let $\mathcal{D}$ be the set of all finite-length closed Dubins tours in the plane. We endow $\mathcal{D}$ with a metric $\rho_{\mathcal{D}}$ as follows. For any two tours $\tau : \mathbb{S} \to \mathbb{R}^2$ and $\tau' : \mathbb{S} \to \mathbb{R}^2$ in $\mathcal{D}$,*

$$\rho_{\mathcal{D}}(\tau, \tau') := \inf_{f:\mathbb{S}\leftrightarrow\mathbb{S}} \sup_{t\in\mathbb{S}} ||\tau(t) - \tau'(f(t))||_2, \tag{7}$$

where $\inf_{f:\mathbb{S}\leftrightarrow\mathbb{S}}$ is the infimum over all reparameterizations between the tours, $\sup_{t\in\mathbb{S}}$ is the supremum over all positions along the tours, and $||\ ||_2$ is the L_2 norm in $\mathbb{R}^2$.

Definition III.8 (Set of PVDTSP-feasible Dubins Tours $\mathcal{D}_{\text{feas}}$)**.** *Let $\mathcal{D}_{\text{feas}}$ be the set of all tours in $\mathcal{D}$ which pass through every polygon of a PVDTSP instance.*

We are now ready to state the main convergence result which will directly lead to convergence properties of the methods in Sec. III.B and III.C.

Theorem III.9 (Roadmap Convergence)**.** *Let $\{\tau_i\}_{i=1}^{\infty}$ be the sequence of best tours contained in the sequence of roadmaps $\{\mathcal{R}_i\}_{i=1}^{\infty}$, respectively. Then the sequence of costs $\{C(\tau_i)\}_{i=1}^{\infty}$ is nonincreasing and*

$$\lim_{i\to\infty} C(\tau_i) \leq \inf_{\tau\in\mathcal{D}_{\text{feas}}^{\circ}} C(\tau). \tag{8}$$

Proof. From Def. III.6 we know that a roadmap $\mathcal{R}_i$ is contained in another roadmap $\mathcal{R}_j$ whenever $j \geq i$, therefore the best tour in $\mathcal{R}_i$ is also in $\mathcal{R}_j$. This ensures the sequence of costs $\{C(\tau_i)\}_{i=1}^{\infty}$ is nonincreasing. The limit of the sequence of costs on the left hand side of Eq. 8 must exist because it is monotonic and lower bounded by zero. To prove the inequality it suffices to show that for all $\epsilon > 0$ there exists N such that $i > N$ implies

$$C(\tau_i) \leq \inf_{\tau\in\mathcal{D}_{\text{feas}}^{\circ}} C(\tau) + \epsilon. \tag{9}$$

By definition of infimum, there exists a sequence of tours $\{\tau_j'\}_{j=1}^{\infty}$ in $\mathcal{D}_{\text{feas}}^{\circ}$ such that

$$\lim_{j\to\infty} C(\tau_j') = \inf_{\tau\in\mathcal{D}_{\text{feas}}^{\circ}} C(\tau),$$

i.e., for all $\epsilon > 0$ there exists N_1 such that $j > N_1$ implies

$$C(\tau_j') \leq \inf_{\tau\in\mathcal{D}_{\text{feas}}^{\circ}} C(\tau) + \frac{\epsilon}{2}. \tag{10}$$

A tour τ_j' must first enter each polygon at a unique entry pose. Because we are sampling poses densely in X_{entry}, we can always choose i large enough that $\mathcal{R}_i$ has a set of entry poses arbitrarily close to the entry poses of τ_j' (with respect to the metric ρ_X). This together with the fact that τ_j' is in the interior of $\mathcal{D}_{\text{feas}}$ implies that for all $\epsilon > 0$ there exists N_2 such that

$$C(\tau_i) \leq C(\tau_j') + \frac{\epsilon}{2} \tag{11}$$

whenever $i > N_2$. Combining Eq. 10 and 11, we obtain the desired result that for all $\epsilon > 0$ there exists $N = \max\{N_1, N_2\}$ such that $i > N$ implies

$$\begin{aligned} C(\tau_i) &\leq C(\tau'_j) + \tfrac{\epsilon}{2} \\ &\leq \inf_{\tau \in \mathcal{D}^\circ_{\text{feas}}} C(\tau) + \tfrac{\epsilon}{2} + \tfrac{\epsilon}{2} \\ &= \inf_{\tau \in \mathcal{D}^\circ_{\text{feas}}} C(\tau) + \epsilon. \end{aligned}$$

□

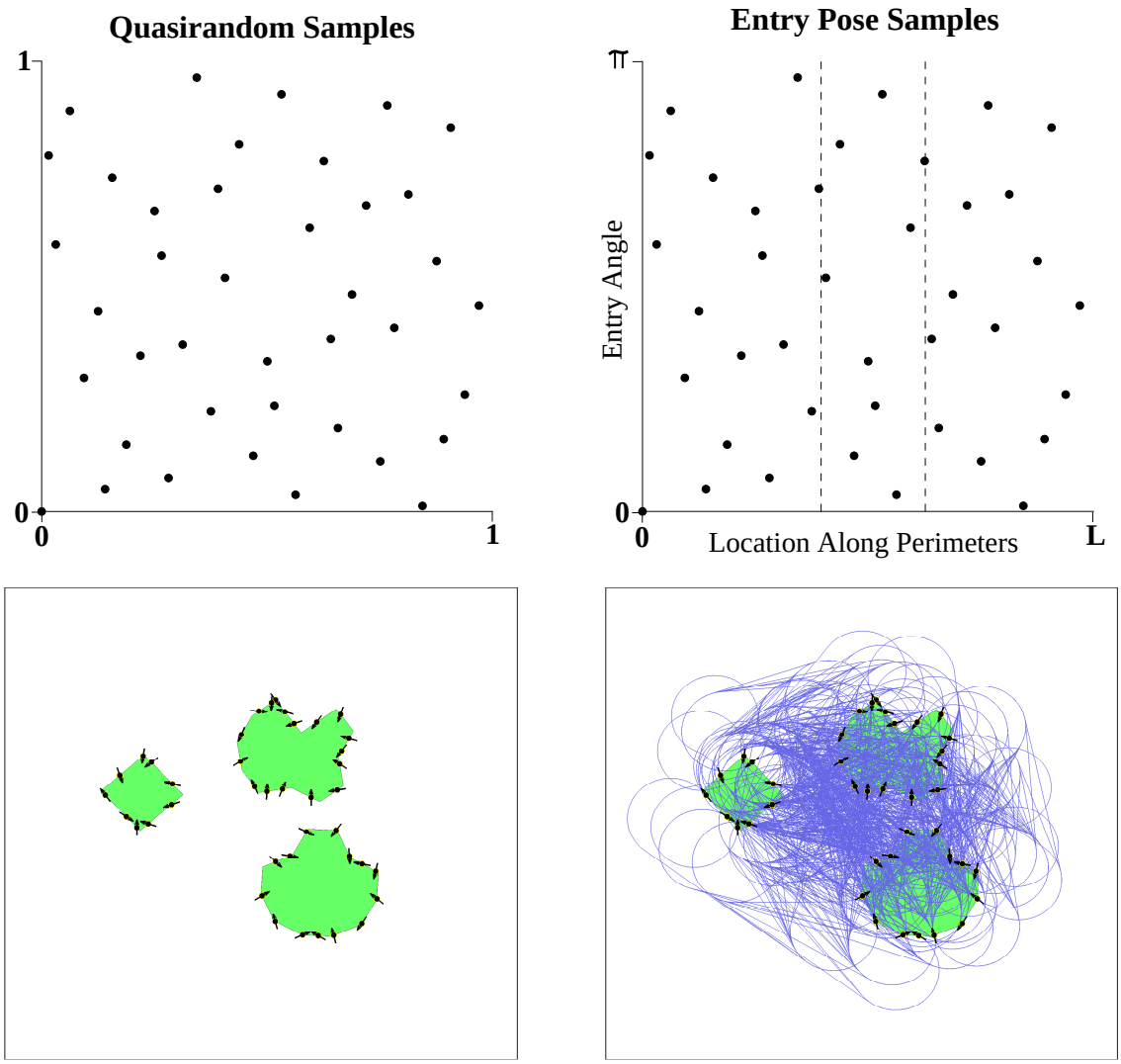

Figure 4. Green polygons are target visibility sets of a PVDTSP instance. The roadmap for a PVDTSP instance is an FOTSP instance which we construct in three steps. First we sample points from a Halton quasirandom sequence on the unit box (upper left). Second we map the quasirandom samples onto the space of entry poses represented by another box (upper right), where the horizontal axis represents a parameterization of position along the 1D polygon perimeters and the vertical axis represents entry angle in radians. These entry poses are the roadmap vertices and the dashed lines (upper right) show the separation between polygons. The samples in the boxes correspond precisely to the entry pose samples shown on the plan view of the polygons (lower left). The vertices are partitioned into clusters according to which polygon they belong to. The third and last step is to create the roadmap edges by making all possible inter-cluster connections between vertices using Dubins shortest paths (lower right, blue curves). The edges are thus weighted by their Dubins distances. For comparison, an example roadmap for collision-free path planning is shown in Fig. 11.

In words, Theorem III.9 states that the best tour cost taken from the sequence of (Halton entry pose induced) roadmaps is no greater, in the limit, than the cost of any nonisolated feasible tour. A roadmap may by chance contain an isolated global optimal tour, hence Eq. 8 is an inequality rather than equality. An example of an isolated global optimum is shown in Fig. 5.

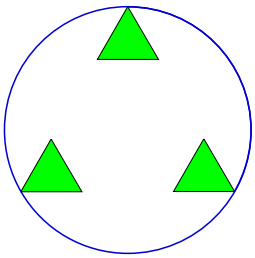

Figure 5. Isolated global optima can exist for a PVDTSP instance. In this example an isolated global optimal tour consists of the minimum turn radius circle (blue) just touching the outer vertices of the polygons.

III.B. Resolution Complete Method

We describe in this section a method which is *resolution complete*, which means it provably converges, in the limit as the number of samples in the roadmap increases, to a solution at least as good as any nonisolated solution.[j] A procedural outline is shown in Tab. 1. The input consists of n polygons, a vehicle minimum

Table 1. Outline of Resolution Complete Method for the PVDTSP

1:	Construct roadmap by sampling entry poses on the polygon boundaries
2:	Use Noon-Bean transform to convert Roadmap to an ATSP instance
3:	Solve ATSP instance
4:	Extract PVDTSP solution from ATSP solution

turn radius $r_{\min}$, and a sample count n_{samples}. First a roadmap is constructed by sampling n_{samples} Halton entry poses as per Def. III.6. Second, the roadmap FOTSP instance is converted to an ATSP instance using the *Noon-Bean transformation*,[19] illustrated in Fig. 6 and defined as follows.

Definition III.10 (Noon-Bean Transformation)**.** *Suppose we are given an FOTSP instance specified, as in Def. III.2, by a weighted directed graph $\mathcal{G} = (V, E)$ together with a partitioning into clusters $S = \{S_1, S_2, S_3, \ldots, S_{n_S}\}$. Then the* Noon-Bean Transformation *of this FOTSP instance is an ATSP instance $\mathcal{G}' = (V', E')$ constructed as follows. Begin with $\mathcal{G}' = \mathcal{G}$, i.e., let $V' = V$ and $E' = E$, then make these three modifications to E':*

(i) For each cluster, add zero-weight directed edges to E' to create a zero-cost cycle which traverses all the vertices of the cluster (so there are a total of n_S zero-cost cycles),

(ii) cyclically shift intercluster edges of E' so that they emanate from the preceeding vertex in their respective zero-cost cycles, and

(iii) add a large penalty $M = \sum_{i,j} w_{i,j}$, i.e., the total of all weights in $\mathcal{G}$, to the weight of all intercluster edges in E'.

The third step of the method is to solve the ATSP instance, which can be done using any exact ATSP solver. The fourth and final step is to extract the PVDTSP solution form the ATSP solution by taking only the first vertex visited in each cluster, thus skipping the fictitious zero-cost edges.

The convergence of the method as the number of samples n_{samples} goes to infinity is captured in the following corollary to Theorem III.9.

Corollary III.11 (Convergence of Resolution Complete Method)**.** *Let $\{\tau_i\}_{i=1}^{\infty}$ be the sequence of tours computed by the resolution complete method when applied to the sequence of roadmaps $\{\mathcal{R}_i\}_{i=1}^{\infty}$, respectively.*

[j] This definition of resolution complete differs slightly from the useage in the context of sampling-based roadmap methods for collision free path planning, where it means that a method is guaranteed to find a nonisolated collision-free path as long as there are enough samples. However, the definitions are deliberately compatible so that a resolution complete collision-free path planner can be used in conjunction with our PVDTSP method in case we desire to solve a PVDTSP with obstacles. We address these topics further in Sec. III.E and IV.A

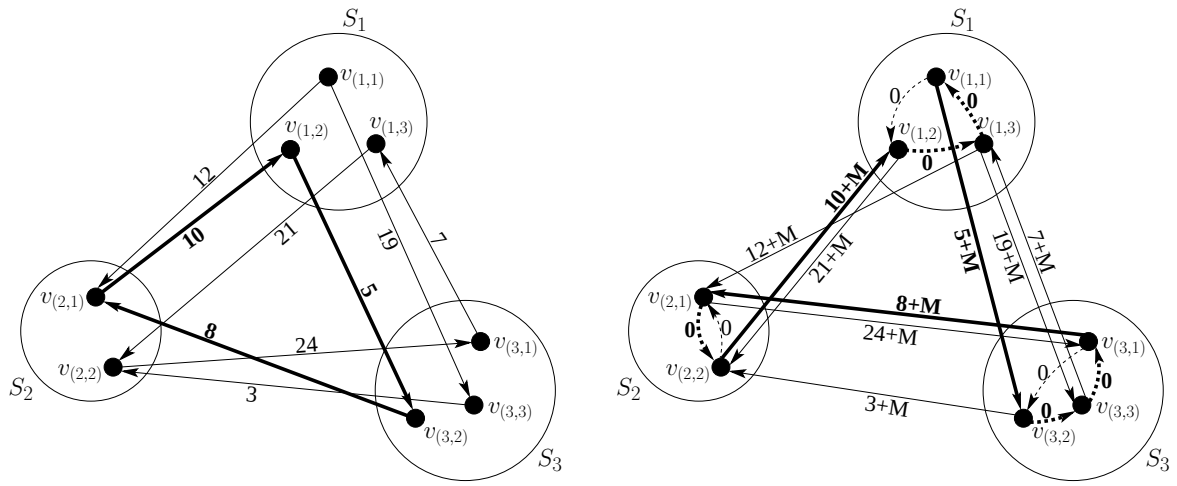

Figure 6. An FOTSP instance such as this example (left) can be transformed into an equivalent ATSP instance (right) using the Noon-Bean transformation. This transformation consists of (1) adding to each cluster a zero-cost cyle which traverses all the vertices of that cluster, (2) cyclically shifting intercluster edges so that they emanate from the preceeding vertex in their respective zero-cost cycles, and (3) adding a large penalty M to interlcuster edges so that each cluster is only visited once. Once a solution to the ATSP instance is found, a solution to the FOTSP instance can by extracted by taking only the first vertex visited in each cluster, thus skipping fictitious zero-cost edges. Bold edges show equivalent optimal tours in the example FOTSP and ATSP instances.

Then the sequence of costs $\{C(\tau_i)\}_{i=1}^{\infty}$ *is nonincreasing and*

$$\lim_{i\to\infty} C(\tau_i) \leq \inf_{\tau\in\mathcal{D}^\circ} C(\tau). \tag{12}$$

Proof. It is proven in Ref. 19 that the Noon-Bean transformation is exact in the sense that if the exact solution of the ATSP instance is found, then the extracted solution to the FOTSP instance will also be exact. This means the method will always find the best tour in a given roadmap. The corollary thus follows directly from Theorem III.9. □

Building the roadmap takes time complexity $\mathcal{O}(n_{\text{samples}}^2)$ because adding a vertex or edge costs constant time and there are $\mathcal{O}(n_{\text{samples}}^2)$ edges (no more than in a complete graph). The Noon-Bean transformation also takes time complexity $\mathcal{O}(n_{\text{samples}}^2)$ because it adds only n_{samples} zero-weight edges, but modifies one at a time the other $\mathcal{O}(n_{\text{samples}}^2)$ edges. Solving the ATSP instance of n_{samples} vertices is NP-hard and therefore we cannot expect to do it in guaranteed polynomial time. However, state-of-the-art heuristic ATSP solvers are effectively exact and have an (empirically determined) average case runtime of $\mathcal{O}(n_{\text{samples}}^{2.2})$.[31,32] We address further in Sec. III.D the issue of ATSP solver exactness versus effective exactness. Extracting the PVDTSP solution from the ATSP solution takes only $\mathcal{O}(n_{\text{samples}})$ time, so supposing we use a heuristic ATSP solver, we can expect the average case runtime of the entire method to be

$$\mathcal{O}(n_{\text{samples}}^{2.2}). \tag{13}$$

III.C. Approximate Dynamic Programming Method

The method we describe in this section uses approximate dynamic programming. A procedural outline is shown in Tab. 2. The input consists of n polygons, a vehicle minimum turn radius $r_{\min}$, and a sample count n_{samples}. First a roadmap is constructed by sampling n_{samples} Halton entry poses as per Def. III.6. Second, the *FST (Fischetti-Salazar-Toth) transformation*,[20] defined in Def. III.12 and illustrated in Fig. 7, is applied to the roadmap FOTSP instance to obtain an ATSP instance.

Definition III.12 (FST Transformation)**.** *Suppose we are given an FOTSP instance specified, as in Def. III.2, by a weighted directed graph* $\mathcal{G} = (V, E)$ *together with a partitioning into clusters* $S = \{S_1, S_2, S_3, \ldots, S_{n_S}\}$*. Then the* FST Transformation *of this FOTSP instance is an ATSP instance* $\mathcal{G}' = (V', E')$ *constructed as follows. There is one vertex in* V' *for every cluster of the FOTSP instance. There is a directed edge from vertex* $v_i' \in V'$ *to vertex* $v_j' \in V'$ *if and only if there exists a directed edge from cluster* S_i *to cluster* S_j*. The*

Table 2. Outline of Approximate Dynamic Programming Method for the PVDTSP

1: Construct roadmap by sampling entry poses on the polygon boundaries
2: Use FST transform to obtain an ATSP instance from the roadmap
3: Solve ATSP instance to obtain a cluster ordering
4: Solve Dynamic Programs induced by cluster ordering from ATSP solution
5: Select best Dynamic Program solution as PVDTSP solution

weight $w'_{i,j}$ of the directed edge from each such v'_i to v'_j is the arithmetic mean of the weights of all directed edges from S_i to S_j.

Solving the ATSP instance in the third step gives an approximate order p to visit the roadmap FOTSP clusters. From this point on, any edges in the roadmap which do not satisfy order p are ignored. Making use of the cluster and vertex labeling scheme introduced in Eq. 5, we now describe the fourth step of the method. Since we can perform a relabeling as necessary, suppose without loss of generality that $p = (1, 2, 3, \ldots, n_S)$, i.e., the approximate cluster ordering is $S_1, S_2, S_3, \ldots, S_{n_S}$. Suppose further that we known the optimal roadmap FOTSP solution passes through the j^*th vertex $v_{(1,j^*)}$ of the first cluster. Treating clusters as stages, an optimal FOTSP solution satsifying order p can be found by solving the dynamic programming recursion

$$\begin{aligned} G^*(v_{(n_S,j)}) &= d(v_{(n_S,j)}, v_{(1,j^*)}) \quad (j = 1, \ldots, n_{S_{n_S}}) \\ G^*(v_{(i,j)}) &= \min_{k\in\{1,\ldots,n_{S_{i+1}}\}} \Big\{ d(v_{(i,j)}, v_{(i+1,k)}) + G^*(v_{(i+1,k)}) \Big\} \quad (i = n_S, \ldots, 2; j = 1, \ldots, n_{S_i}) \\ G^*(v_{(1,j^*)}) &= \min_{k\in\{1,\ldots,n_{S_2}\}} \Big\{ d(v_{(1,j^*)}, v_{(2,k)}) + G^*(v_{(2,k)}) \Big\}, \end{aligned} \tag{14}$$

where $G^*(v)$ denotes the optimal-cost-to-go from a vertex v, and $d(v, v')$ is the weight of the directed edge from vertex v to vertex v'. Since it is not known a prior which vertex in the first cluster is i^*, one dynamic program must be solved for each vertex in the first cluster, hence step five of the method is to select the best out of n_{S_1} dynamic program solutions.

The convergence of the method as the number of samples n_{samples} goes to infinity is captured in the following corollary to Theorem III.9.

Corollary III.13 (Convergence of Approximate Dynamic Programming Method)**.** *Let $\{\tau_i\}_{i=1}^{\infty}$ be a sequence of tours computed by the approximate dynamic programming method when applied to the sequence of roadmaps $\{\mathcal{R}_i\}_{i=1}^{\infty}$, respectively. If there exists N such that τ_i satisfies an order p for all $i > N$ then the sequence of costs $\{C(\tau_i)\}_{i=N+1}^{\infty}$ is nonincreasing and*

$$\lim_{i\to\infty} C(\tau_i) \leq \inf_{\tau\in(\mathcal{D}^{\circ}_{\text{feas}})_p} C(\tau), \tag{15}$$

where $(\mathcal{D}_{\text{feas}})_p$ is the set of all PVDTSP-feasible tours which satisfy order p.

Proof. Let $\{(\mathcal{R}_i)_p\}_{i=1}^{\infty}$ be the sequence of roadmaps as in Def. III.6, but where only edges satisfying order p are allowed. Dynamic programming will always find the best tours in these p-limited roadmaps, therefore the proof of this corollary is exactly the same as the proof for Theorem III.9, except $\mathcal{D}_{rmfeas}$ is replaced by $(\mathcal{D}_{\text{feas}})_p$ and $\{\mathcal{R}_i\}_{i=1}^{\infty}$ by $\{(\mathcal{R}_i)_p\}_{i=1}^{\infty}$. □

Building the roadmap takes time complexity $\mathcal{O}(n^2_{\text{samples}})$ because adding a vertex or edge takes constant time and there are $\mathcal{O}(n^2_{\text{samples}})$ edges (no more than in a complete graph). The FST transformation also takes time complexity $\mathcal{O}(n^2_{\text{samples}})$ because it must access each of the $\mathcal{O}(n^2_{\text{samples}})$ edges of the FOTSP instance in order compute the weights of the edges in the ATSP instance. Since the ATSP is NP-hard, we assume a state-of-the-art heuristic ATSP solver with (empirically determined) average case runtime of $\mathcal{O}(n^{2.2})$ will be used.[31,32] On average there will be $\frac{n_{\text{samples}}}{n}$ vertices per cluster, so the average case total time complexity of solving the dynamic programs is $\mathcal{O}(\frac{n^2_{\text{samples}}}{n})$. Adding up all the time complexities, we can expect the average case runtime of the entire method to be

$$\mathcal{O}(n^{2.2} + \frac{n^2_{\text{samples}}}{n} + n_{\text{samples}}). \tag{16}$$

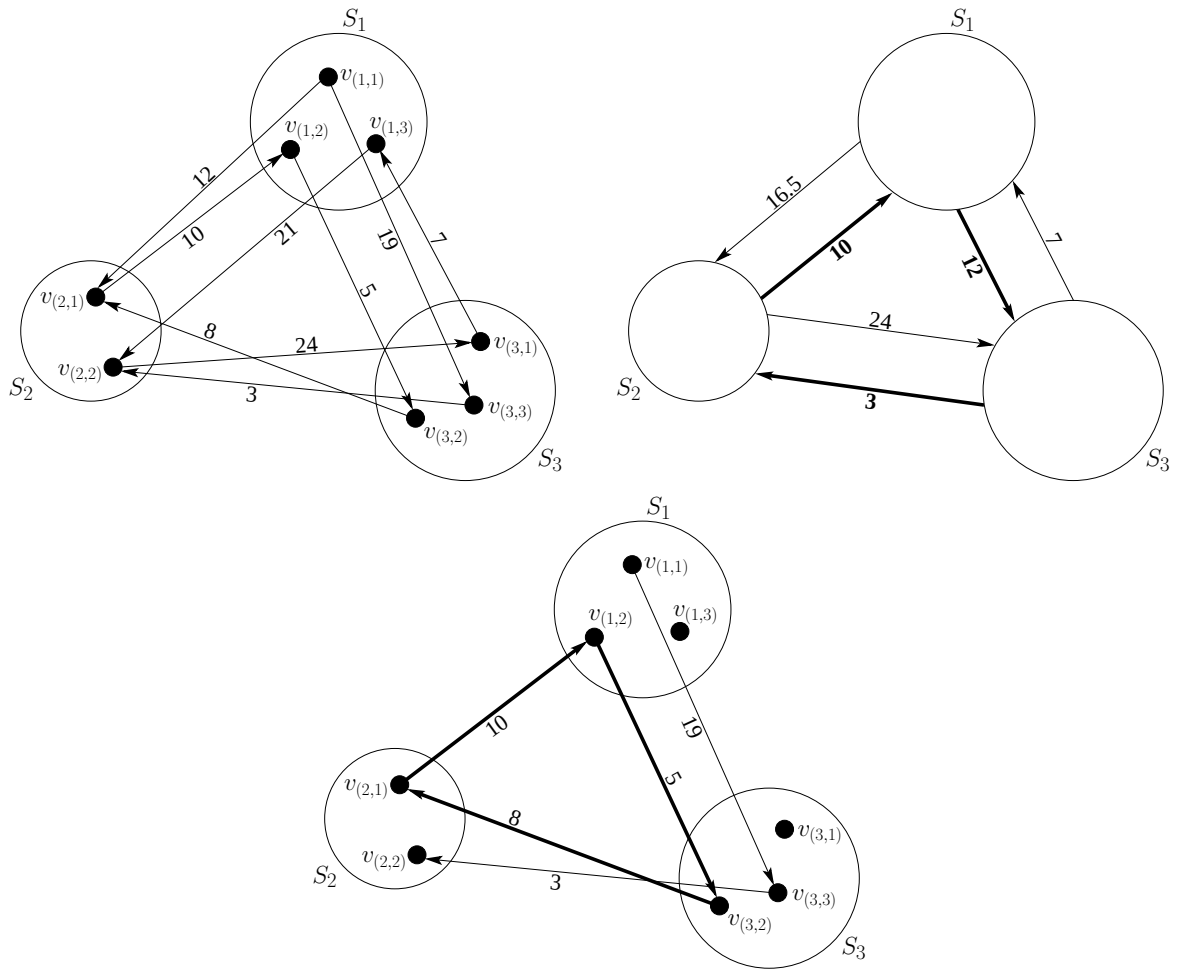

Figure 7. The FST transform of an FOTSP instance (upper left) is an ATSP instance (upper right) where (1) the vertices of the ATSP instance correspond to clusters of the FOTSP instance, and (2) the weight of each directed edge in the ATSP instance is the average weight of directed edges between the respective clusters in the FOTSP instance. Solving the ATSP instance gives an approximate ordering, say p, of the clusters of the FOTSP instance. If one ignores in the FOTSP instance all edges not satisfying the ordering p (bottom), then an approximate FOTSP solution (bottom bold) can be found by dynamic programming with the clusters as stages.

III.D. Numerical Study

We have implemented the sampling-based roadmap methods of Sec. III.B and III.C in C++ on a 2.33 GHz i686. For solving ATSP instances, our implementations call the powerful LKH[32] solver as a subroutine. Strictly speaking, LKH is based on what is known as the Lin-Kernighan heuristic, and therefore is an inexact solver, i.e., it is not guaranteed to find the global optimal solution to an ATSP instance. Using an inexact ATSP solver with what we have been calling the "resolution complete method" means that it is no longer truly resolution complete and therefore not guaranteed to converge to a nonisolated global optimum. However, allowing ourselves a slight abuse of terminology, we retain the name "resolution complete method" in the presentation of our numerical results because state-of-the-art heuristic TSP solvers, e.g., LKH or Linkern, perform so well in practice that they are widely accepted as *effectively exact* for ATSP instances having up to hundreds or even thousands of nodes.[32,33] Moreover, exact ATSP solvers can be extremely slow, sometimes taking hours to find a solution that an inexact heuristic solver finds in only seconds. [k] Hours of computation time may not be available in UAV applications requiring online solutions.

Out of several dozen problem instances we experimented with, the results from three representative examples are shown in Tab. 3, Fig. 8, Fig. 9, and Fig. 10. In all examples the aircraft minimum turn

[k] According to Ref. 32, the empirically determined average case run-time of LKH on an ATSP instance with n_{nodes} nodes is $\mathcal{O}(n_{\text{nodes}}^{2.2})$.

radius was $r_{\min} = 3$ m. Both methods deliver good solutions and are suitably fast for online purposes when applied to PVDTSP instances having up to about 20 targets. The computation times for the approximate dynamic programming method are generally a little shorter than for the resolution complete method, but the resulting tours are also a little longer. The plots of computation time vs. sample count seem to match the predicted average case time complexities in Eq. 13 and 16. One can see, from the plots of solution quality vs. sample count and computation time vs. sample count, that a user of either method can indirectly trade off computation time for solution quality by adjusting the number of samples. The resolution complete method appears to monotonically converge to nonisolated global optima as the number of samples grows, so we presume the LKH solver is indeed effectively exact for these examples having up to 20 targets and 1500 samples. In just a few examples out of dozens we tested did we observe slight nonmonotonicity. This mostly occured when there were greater than 20 targets and 1500 samples. Although this indicates LKH is no longer effectively exact for the larger size problem instances, the approximate solutions it gave were consistently very good.

The PVDTSP instances used for experimentation in this section are the same as those used for testing the genetic algorithm presented in Ref. 6. For small instances with around 5 targets or less, the performance of the genetic algorithm, in terms of solution quality per computation time, is comparable to that of the sampling-based roadmap methods. For larger problem instances with greater than 5 targets the sampling-based roadmap methods perform significantly better.

Table 3. Statistics from resolution complete and approximate dynamic programming algorithms implemented in C++ on a 2.33 GHz i686.

Instance	No. of Samples	Resolution Complete		Approx. Dynamic Programming	
		Computation Time	Tour Length	Computation Time	Tour Length
Fig. 8 5 targets	400	8.05 s	37.64 m	6.12 s	37.64 m
Fig. 9 10 targets	800	53.54 s	67.44 m	51.42 s	69.89 m
Fig. 10 20 targets	1500	506.07 s	118.99 m	447.14 s	142.03 m

III.E. Relationship to Sampling-Based Roadmap Methods for Collision-Free Path Planning

As mentioned in Sec. I, sampling-based roadmap methods in the robotics literature, surveyed nicely in the recent texts Ref. 21 and 22, have traditionally been used exclusively for planning collision-free paths through continuous spaces by discretizing the obstacle-free portion of the space into a finite directed graph called a *roadmap*, e.g., as shown in Fig. 11. The roadmap can then be searched using standard shortest path algorithms such as Dijkstra or A*.[34] It is interesting to note that for collision-free path planning the roadmap vertices must be sampled from the full 3-dimensional Dubins state space, yet for a PVDTSP roadmap it is sufficient to sample only on the 2-dimensional space of entry poses (Fig. 4 vs. Fig. 11).

IV. Extensibility

IV.A. Handling Wind, Airspace Constraints, and Any Vehicle Dynamics

In Sec. II we formulated the minimum time reconnaissance path planning problem as finding a sequence of states $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ from which the targets can be photographed. We assumed a minimum time state-to-state trajectory planner was available as a "black box" that could be accessed by our algorithms in order to evaluate the distance function $d(\mathbf{x}, \mathbf{x}')$, which is all we need to evaluate the goodness of any candidate solution. In this way, the minimum time state-to-state trajectory planner is a *module* within our algorithms. We could therefore use our algorithms with any of the minimum time state-to-state trajectory planners available in the literature. These include planners which can handle wind, no-fly zones, and any vehicle dynamics. The literature on nonholonomic trajectory planning is vast, so we survey only briefly a few works most relevant.

Without obstacles or wind, the procedure for computing an optimal Dubins pose-to-pose path was first shown using measure theoretic arguments in Ref. 3, then later more concisely using Pontryagin's maximum principle from optimal control in Ref. 24. More recently it has been shown that in a constant wind field without obstacles, a shortest pose-to-pose Dubins path can be caclulated in constant time to fixed accuracy.[4,35,36] Using these methods, pose-to-pose shortest path queries with no obstacles can be computed in constant time.

Given a polygonal environment with polygonal holes represented by a total of m vertices, the shortest collision-free Euclidean path (no curvature constraint) can be calculated in $\mathcal{O}(m \log m)$ time.[37] Unfortunately the same problem with a Dubins vehicle is NP-hard in m.[38] However, much work has been done to quickly find nearly optimal obstacle avoiding paths, surveyed further in Ref. 21, 22, 39. Trajectory planners specifically intended for fixed-wing UAVs, which use a branch and bound technique, are described in Ref. 40, 41. Another approach to nonholonomic motion planning with obstacles is to use a MILP (Mixed Integer Linear Program).[42,43]

IV.B. Open-Path vs. Closed-Tour Problems

So far in this article, we have considered only closed-tour solutions to the reconnaissance UAV path planning problem. One may alternatively wish to find an open reconnaissance path from a fixed initial pose $\mathbf{x}_{\text{initial}}$ to a different fixed final pose $\mathbf{x}_{\text{final}}$. In this case, instead of the closed-tour cost function in Eq. 2, we use the *open path cost function*

$$C(\mathbf{x}_1, \ldots, \mathbf{x}_n) \;=\; d(\mathbf{x}_{\text{initial}}, \mathbf{x}_1) + \sum_{i=1}^{n-1} d(\mathbf{x}_i, \mathbf{x}_{i+1}) + d(\mathbf{x}_n, \mathbf{x}_{\text{final}}).$$

The sampling-based roadmap methods can be applied to open-path problems with only slight modification in how the roadmap is constructed. The open-path roadmap has all the vertices and edges that the closed-tour roadmap of Def. III.3 does, but in addition has two degenerate (single-vertex) clusters, one for the initial pose and one for the final pose. The initial degenerate cluster is connected by distance d-weighted edges outgoing to all vertices in nondegenerate clusters. The final degenerate cluster is connected (1) by a zero-weight edge outgoing to the initial cluster vertex, and (2) by distance d-weighted edges incoming from all vertices in the nondegenerate clusters.

V. Conclusion

We have formulated the general aircraft visual reconnaissance problem for static ground targets in terrain and shown that, under simplifying assumptions, it can be reduced to a variant of the Traveling Salesman Problem which we call the PVDTSP (Polygon-Visiting Dubins Traveling Salesman Problem). The PVDTSP reduces to the well-studied DTSP and ETSP in the *sparse limit* as targets are very far apart, but our worst-case analysis demonstrated the importance of developing specialized algorithms for the PVDTSP in the *dense limit* as targets are close together and polygons may overlap significantly. We designed two sampling-based roadmap methods for the PVDTSP. These methods operate by sampling finite discrete sets of vehicle states to approximate a PVDTSP instance by an FOTSP instance, then applying existing FOTSP algorithms. One of our sampling-based roadmap methods uses what is known as the Noon-Bean transformation and is *resolution complete*, which means it provably converges to a nonisolated global optimum as the number of samples grows. Our other sampling-based roadmap method achieves faster computation times by using an approximate dynamic programming technique, but consequently only converges to a nonisolated global optimum modulo target order. In numerical experiments, both sampling-based roadmap methods delivered good solutions suitably quickly for online purposes when applied to PVDTSP instances having up to about 20 targets. For problem instances with greater than 5 targets the sampling-based roadmap methods significantly outperformed the genetic algorithm in Ref. 6. Additionally, both methods allow trade-off of computation time for solution quality and are extensible to handle wind, airspace constraints, any vehicle dynamics, and open-path problems. The methods could also be used in a receding horizon fashion for stochastic scenarios with pop-up targets.

While the algorithms we have presented are essentially ready to be fielded, there is much room for future work. We are currently investigating extensions to multiple vehicles, constant factor approximation guarantees, a way to calculate how many samples a roadmap needs to guarantee a prescribed accuracy, and

optimal ratios of roadmap orientation dispersion to position dispersion. Aside form improvements to the existing algorithms, it would be interesting to numerically evaluate hybrid approaches.

Acknowledgments

This work was supported by a US Department of Defense SMART Graduate Fellowship (`http://www.asee.org/fellowships/smart/`). Thanks to the following people for helpful comments: F. Bullo (UCSB), R. Holsapple (WPAFB), D. Kingston (WPAFB), M. Mears (WPAFB), S. Rasmussen (Miami Valley Aerospace LLC), C. Schumacher (WPAFB), V. Shaferman (Technion).

References

[1]Chandler, P., Pachter, M., and Rasmussen, S., "UAV Cooperative Control," *American Control Conference*, Arlington, VA, June 2001, pp. 50–55.

[2]Ryan, A., Zennaro, M., Howell, A., Sengupta, R., and Hedrick, J. K., "An overview of emerging results in cooperative UAV control," *IEEE Conf. on Decision and Control*, 2004.

[3]Dubins, L. E., "On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents," *American Journal of Mathematics*, Vol. 79, 1957, pp. 497–516.

[4]McGee, T. G., Spry, S., and Hedrick, J. K., "Optimal path planning in a constant wind with a bounded turning rate," *AIAA Conf. on Guidance, Navigation and Control*, San Francisco, CA, Aug. 2005, Electronic Proceedings.

[5]Gutin, G. and Punnen, A. P., *The Traveling Salesman Problem and Its Variations*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1st ed., 2002.

[6]Obermeyer, K. J., "Path Planning for a UAV Performing Reconnaissance of Static Ground Targets in Terrain," *AIAA Conf. on Guidance, Navigation and Control*, Chicago, IL, Aug. 2009, To appear.

[7]Betts, J. T., "Survey of Numerical Methods for Trajectory Optimization," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 21, No. 2, 1998, pp. 193–207.

[8]Papadimitriou, C. H., "The Euclidean Traveling Salesman Problem is NP-Complete," *Theoretical Computer Science*, Vol. 4, 1977, pp. 237–244.

[9]Ny, J. L., Frazzoli, E., and Feron, E., "The curvature-constrained traveling salesman problem for high point densities," *IEEE Conf. on Decision and Control*, 2007, pp. 5985–5990.

[10]Nygard, K. E., Chandler, P. R., and Pachter, M., "Dynamic network flow optimization models for air vehicle resource allocation," 2001.

[11]Schumacher, C., Chandler, P. R., and Rasmussen, S. R., "Task allocation for wide area search munitions," *American Control Conference*, 2002.

[12]Tang, Z. and Özgüner, Ü., "Motion Planning for Multi-Target Surveillance with Mobile Sensor Agents," *IEEE Transactions on Robotics*, Vol. 21, No. 5, 2005, pp. 898–908.

[13]Rathinam, S., Sengupta, R., and Darbha, S., "A Resource Allocation Algorithm for Multi-Vehicle Systems with Non holonomic Constraints," *IEEE Transactions on Automation Sciences and Engineering*, Vol. 4, No. 1, 2007, pp. 98–104.

[14]Savla, K., Frazzoli, E., and Bullo, F., "Traveling Salesperson Problems for the Dubins vehicle," *IEEE Transactions on Automatic Control*, Vol. 53, No. 6, 2008, pp. 1378–1391.

[15]Le Ny, J. and Feron, E., "An Approximation Algorithm for the Curvature-Constrained Traveling Salesman Problem," *Allerton Conf. on Communications, Control and Computing*, Monticello, IL, Sept. 2005.

[16]de Berg, M., Gudmundsson, J., Katz, M. J., Levcopoulos, C., Overmars, M. H., and van der Stappen, A. F., "TSP with Neighborhoods of Varying Size," 2002, pp. 21–35.

[17]Dumitresco, A. and Mitchell, J. S. B., "Approximation Algorithms for TSP with Neighborhoods in the Plane," *Journal of Algorithms*, Vol. 48, No. 1, 2003, pp. 135–159.

[18]Mata, C. S. and Mitchell, J. S. B., "Approximation Algorithms for Geometric Tour and Network Design Problems," *Symposium on Computational Geometry*, 1995, pp. 360–369.

[19]Noon, C. E. and Bean, J. C., "An Efficient Transformation of the Generalized Traveling Salesman Problem," Tech. Rep. 91-26, Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, 1991.

[20]Fischetti, M., Salazar-González, J. J., and Toth, P., *The Traveling Salesman Problem and its Variations*, chap. 13, Kluwer, 2002.

[21]Choset, H., Lynch, K. M., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L. E., and Thrun, S., *Principles of Robot Motion: Theory, Algorithms, and Implementations*, MIT Press, 2005.

[22]LaValle, S. M., *Planning Algorithms*, Cambridge University Press, 2006, Available at `http://planning.cs.uiuc.edu`.

[23]Oberlin, P., Rathinam, S., and Darbha, S., "A Transformation for a Heterogeneous, Multiple Depot, Multiple Traveling Salesmen Problem," *American Control Conference*, 2009, pp. 1292–1297.

[24]Boissonnat, J.-D., Cérézo, A., and Leblond, J., "Shortest paths of bounded curvature in the plane," *Journal of Intelligent and Robotic Systems*, Vol. 11, 1994, pp. 5–20.

[25]Shaferman, V. and Shima, T., "Cooperative UAV Tracking Under Urban Occlusions and Airspace Limitations," *AIAA Conf. on Guidance, Navigation and Control*, Honolulu, Hawaii, Aug 2008, Electronic Proceedings.

[26]Shaferman, V. and Shima, T., "Co-Evolution Genetic Algorithm for UAV Distributed Tracking in Urban Environments," *ASME Conference on Engineering Systems Design and Analysis*, Jul 2008.

[27]Ghosh, S. K., *Visibility Algorithms in the Plane*, Cambridge University Press, 2007.

[28]Bresenham, J. E., "Algorithm for Computer Control of a Digital Plotter," *Seminal Graphics: Pioneering Efforts that Shaped the Field*, Association for Computing Machinery, New York, NY, USA, 1998, pp. 1–6.

[29]Niederreiter, H., *Random Number Generation and Quasi-Monte Carlo Methods*, No. 63 in CBMS-NSF Regional Conference Series in Applied Mathematics, Society for Industrial & Applied Mathematics, 1992.

[30]Halton, J. H., "On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals," *Numerische Mathematik*, Vol. 2, 1960, pp. 84–90.

[31]Applegate, D., Bixby, R., Chvátal, V., and Cook, W., "On the solution of traveling salesman problems," *Documenta Mathematica, Journal der Deutschen Mathematiker-Vereinigung*, Berlin, Germany, Aug. 1998, pp. 645–656, Proceedings of the International Congress of Mathematicians, Extra Volume ICM III.

[32]Helsgaun, K., "An effective implementation of the LinKernighan traveling salesman heuristic," *European Journal of Operational Research*, Vol. 126, No. 1, October 2000, pp. 106–130.

[33]Applegate, D. L., Bixby, R. E., and Chvátal, V., *The Traveling Salesman Problem: A Computational Study*, Applied Mathematics Series, Princeton University Press, 2006.

[34]Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C., *Introduction to Algorithms*, MIT Press, 2nd ed., 2001.

[35]McNeely, R., Iyer, R. V., and Chandler, P., "Tour Planning for an Unmanned Air Vehicle under Wind Conditions," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 30, No. 5, 2007, pp. 1299–1306.

[36]Techy, L. and Woolsey, C. A., "Minimum-Time Path Planning for Unmanned Aerial Vehicles in Steady Uniform Winds," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 32, No. 6, 2009, pp. 1736–1746.

[37]Hershberger, J. and Suri, S., "An Optimal Algorithm for Euclidean Shortest Paths in the Plane," *SIAM Journal on Computing*, Vol. 28, 1999, pp. 2215–2256.

[38]Reif, J. and Wang, H., "The Complexity of the Two Dimensional Curvature-Constrained Shortest-Path Problem," *In Proc. Third International Workshop on the Algorithmic Foundations of Robotics*, 1998, pp. 49–57.

[39]Latombe, J.-C., *Robot Motion Planning*, Kluwer Academic Publishers, 1991.

[40]Eele, A. and Richards, A., "Path-Planning with Avoidance using Nonlinear Branch-and-Bound Optimisation," *AIAA Conf. on Guidance, Navigation and Control*, Hilton Head, South Carolina, 2007, Electronic Proceedings.

[41]Eele, A. and Richards, A., "Comparison of Branching Strategies for Path-Planning with Avoidance using Nonlinear Branch-and-Bound," *AIAA Conf. on Guidance, Navigation and Control*, Honolulu, Hawaii, 2008, Electronic Proceedings.

[42]Borrelli, F., Subramanian, D., Raghunathan, A. U., and Biegler, L. T., "MILP and NLP Techniques for centralized trajectory planning of multiple unmanned air vehicles," *American Control Conference*, 2006.

[43]Richards, A. and How, J. P., "Aircraft trajectory planning with collision avoidance using mixed integer linear programming," *American Control Conference*, 2002, pp. 1936–1941.

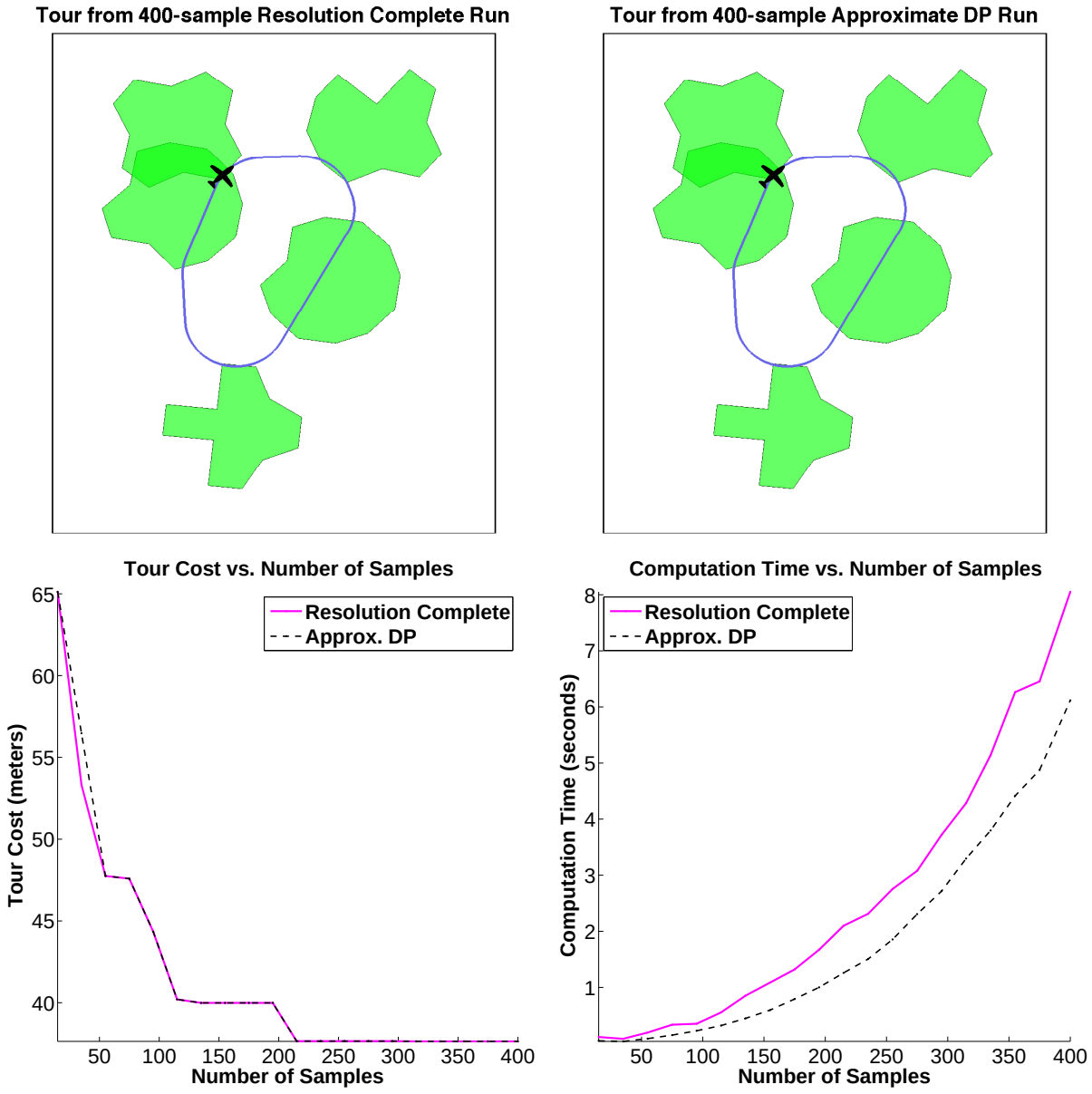

Figure 8. Computed example with $n = 5$ targets, aircraft minimum turn radius $r_{\min} = 3$ m. Green polygons represent the target visibility regions. Black dots are the tour nodes. Cf Tab. 3.

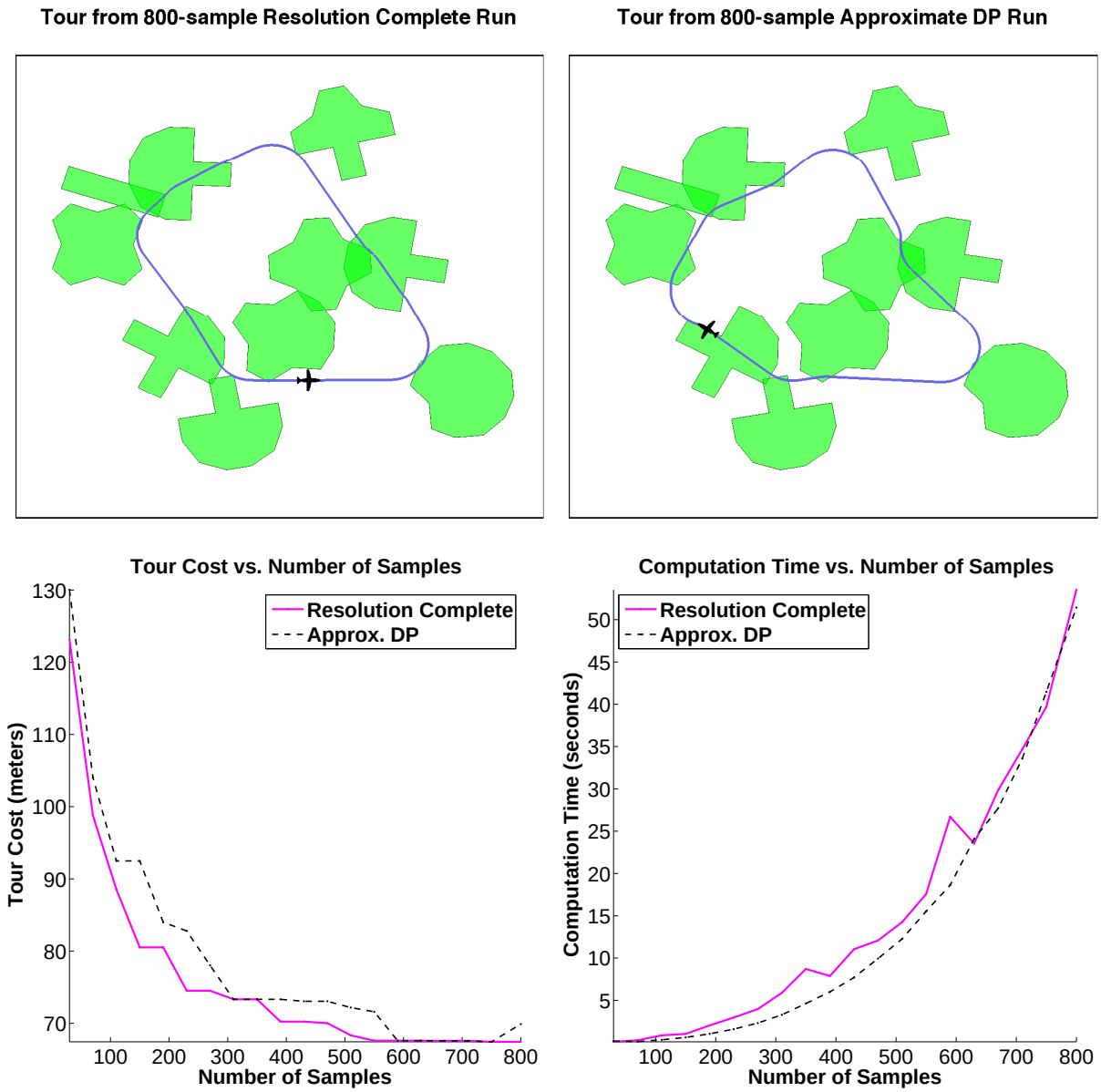

Figure 9. Computed example with $n = 10$ targets, aircraft minimum turn radius $r_{\min} = 3$ m. Green polygons represent target visibility regions. Black dots are the tour nodes. Cf Tab. 3.

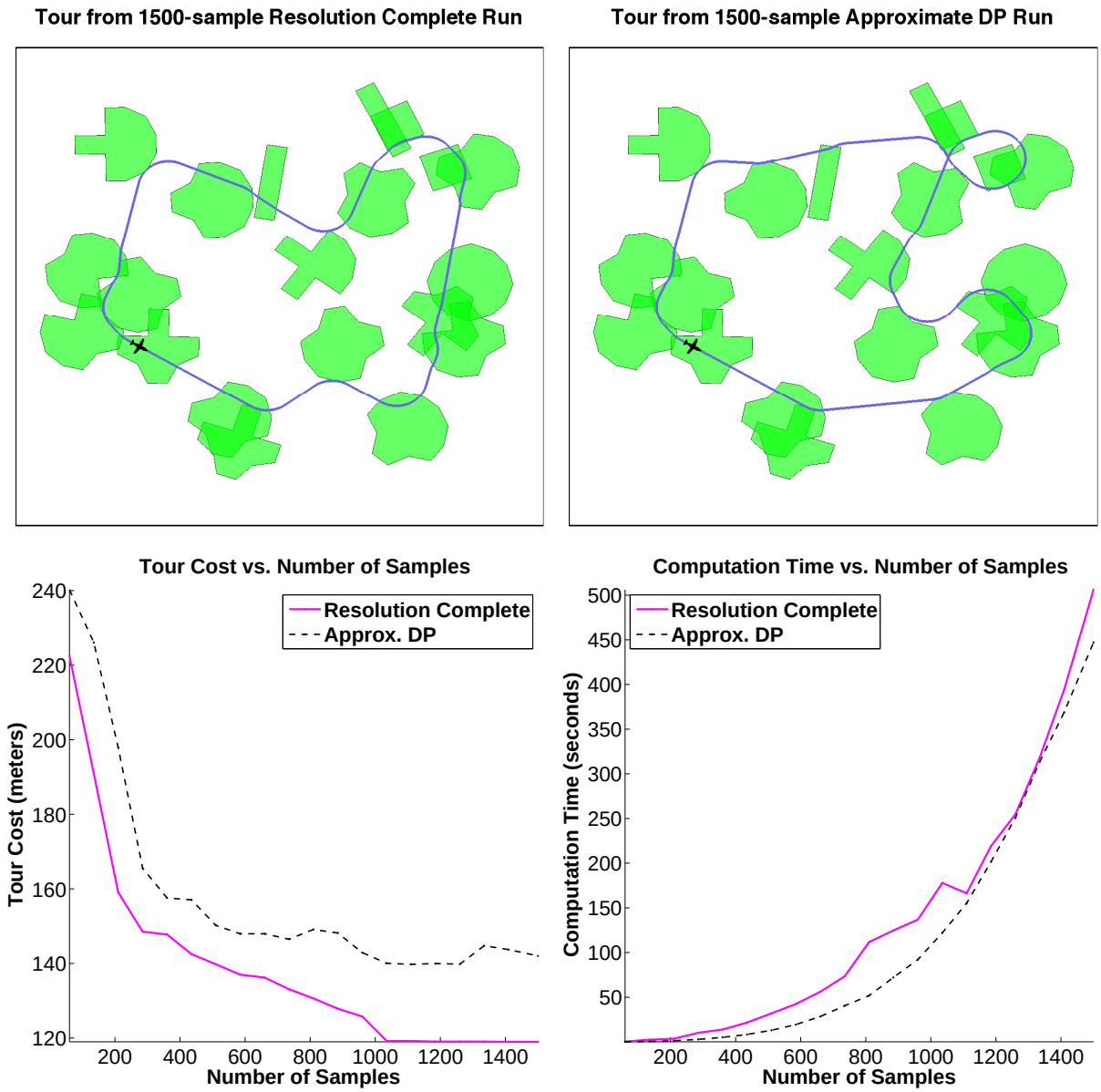

Figure 10. Computed example with $n = 20$ targets, aircraft minimum turn radius $r_{\min} = 3$ m. Green polygons represent target visibility regions. Black dots are the tour nodes. Cf Tab. 3.

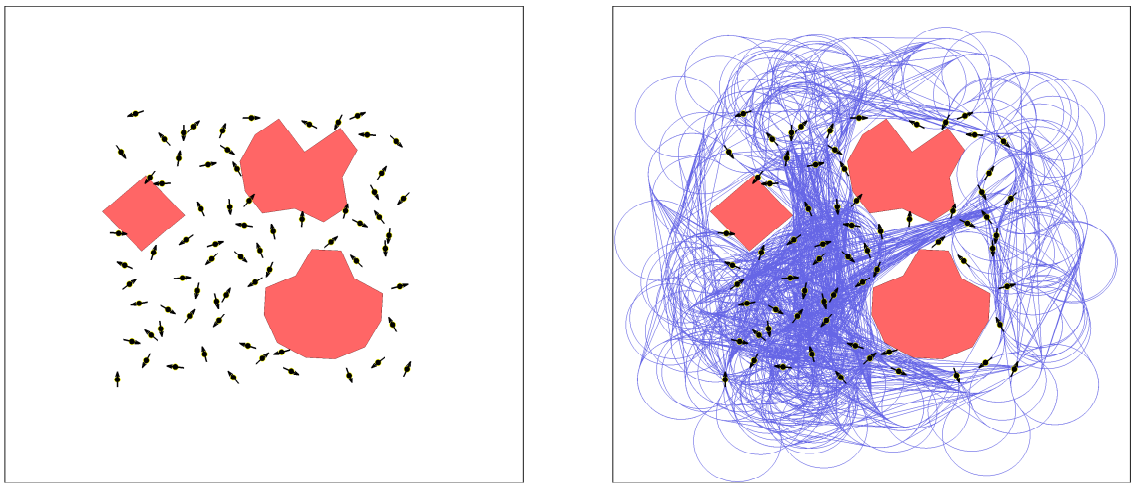

Figure 11. Suppose the red polygons are obstacles. A roadmap for computing collision-free Dubins paths between pairs of poses is a finite directed graph whose vertices are poses sampled from the freespace (black arrows, left) and whose edges are obtained by attempting to make collision-free connections between samples using a *local planning method*, usually a Boundary Value Problem solver (blue curves, right). In this example poses were sampled using a Halton quasirandom sequence in SE(2), however, one could alternatively use random or uniform grid sampling. To find a collision-free path between two poses, those poses are connected to the roadmap using the local planning method, then the roadmap is searched using a shortest path algorithm such as Dijkstra or A*. Cf. Fig. 4.