Path Planning and Energy Management of Hybrid Air Vehicles for Urban Air Mobility

Satyanarayana G. Manyam\textsuperscript{1}, David W. Casbeer\textsuperscript{2}, Swaroop Darbha\textsuperscript{3}, Isaac E. Weintraub\textsuperscript{2} and Krishna Kalyanam\textsuperscript{4}

Abstract—A novel coupled path planning and energy management problem for a hybrid unmanned air vehicle is considered, where the hybrid vehicle is powered by a dual gas/electric system. Such an aerial robot is envisioned for use in an urban setting where noise restrictions are in place in certain zones necessitating battery only operation. We consider the discrete version of this problem, where a graph is constructed by sampling the boundaries of the restricted zones, and develop a path planning algorithm. The planner simultaneously solves the path planning along with the energy mode switching control, under battery constraints and noise restrictions. This is a coupled problem involving discrete decision making to find the path to travel, and determining the state of charge of the battery along the path, which is a continuous variable. A sampling based algorithm to find near optimal solution to this problem is presented. To quantify the efficacy of the solution, an algorithm that computes tight lower bounds is also presented. The algorithms presented are verified using numerical simulations, and the average gap between the feasible solutions (upper bounds) and the lower bounds are, empirically, shown to be within 15\% of each other.

I. INTRODUCTION

In the area of urban air mobility (UAM) and drone delivery, many commercial ventures are considering electric propulsion aircraft \cite{1}, \cite{2}. Given the deficiencies in state-of-the-art lithium-ion battery energy density and fuel cell technology \cite{3}, \cite{4}, it is prudent to consider alternative technologies that can help reduce our carbon footprint in the near term. Furthermore, as the world looks for faster modes of transportation and quicker delivery of goods, our skies will become saturated with the noise from these drones \cite{5}. In the most prevalent use cases, these vehicles will operate in locations where such drone background noise is unacceptable. A gasoline-electric hybrid aerial robotic vehicle is well suited for UAM or drone delivery applications, where the gasoline engine provides long endurance and electric motor facilitates the low noise mode \cite{6}. The perceived noise level when the aerial vehicle is powered by electric motor is considerably less compared to a gasoline engine \cite{7}, \cite{8}.

In this letter, we consider a noise constrained path planning problem for a hybrid gasoline-electric unmanned air vehicle. We assume the robotic vehicle is equipped with a series hybrid architecture \cite{9}, where the propellers are powered by an electric motor that draws power from either the gasoline engine-generator or from the battery. The gasoline engine, when run at full capacity (maximum flow rate of fuel), can run the motor and also charge the battery; or the engine can be run at a limited capacity to produce only sufficient power to run the motor. However, due to frictional losses, it is more efficient to run at full capacity and charge the battery to full whenever possible, and then power the motor with the battery. Therefore, we assume that the fuel rate is at maximum whenever the gasoline mode is chosen.

We further assume that the architecture facilitates instant effortless switching between these two modes. The robotic vehicles considered can make sharp turns similar to quadrotors, and therefore, we do not consider the kinematic constraints.

The path planning problem involves finding a path between a pre-specified start and goal locations in the presence of quiet zones. The aerial robot is allowed to pass through quiet zones, however it must be powered by the electric mode (gasoline-engine turned off) while flying above such zones. A candidate path can be divided into several segments, where each segment could be either gasoline or electric mode. The cost we have considered in this letter is the fuel cost, and therefore the objective is to minimize the length of the segments that are traveled in gasoline mode. This cost is appropriate for commercial applications where the objective is to minimize the fuel consumption. However, the framework presented here can easily modified for any other application that aims to minimize a different objective, for example, travel time. The decision making involves finding the path, while simultaneously determining the switching points from gas to electric and vice-versa. To do so, the planner must determine the segments along the path where the power source is the gasoline engine or the electric motor, such that state of charge remains within the capacity limits.

To find the optimal paths, one needs to model the battery characteristics, the rate of charge of the battery while the robot travels in gasoline mode and the rate of discharge in electric mode. For simplicity, in the operational limits of the battery charge level, we assume that the rates of discharge and recharge with respect to distance traveled are constants. Let $q(s)$ be the variable representing battery charge along a path, where the parameter $s$ represents the length of the path.
traveled, then

\[ q'(s) = \begin{cases} 
-\alpha & \text{if electric mode,} \\
\beta & \text{if gasoline mode,} 
\end{cases} \]

(1)

\[ q_{\min} \leq q(s) \leq q_{\max}, \]

(2)

where \( \alpha \) and \( \beta \) are the rates of discharge and recharge per unit distance traveled. This model allows for quick evaluation of feasibility for a given state of charge at initial and final position of a segment of the path. The path planning algorithm we present in this paper can accommodate higher fidelity battery models.

**Toy Example:** Suppose the initial charge is 80\%, and final charge level is required to be at least 50\%. A feasible path and the corresponding charge profile along the path are shown in Fig. 1, where there exists one quiet zone, shown in grey. The path is demarcated and shown in magenta and green for the gasoline and electric modes, respectively. Note that the charge profiles are linear due to the model assumed in (1). In this example, the robot travels using the gasoline mode and switches to electric mode before it enters the quiet zone and continues in the electric mode throughout the quiet zone, and switches to the gasoline mode after exiting from the quiet zone. It is clear that the mode of the segment in the quiet zone is electric, and the other segments could be either gasoline or electric. **The planning algorithm must choose the modes and switching points along a path that minimizes the total fuel cost.**

Let us represent a path as a series of segments \( \{(v_s, v_1), (v_1, v_2), \ldots (v_n, v_f)\} \). For a given path, one solution approach would be to determine the charge levels \( \{q_1, \ldots q_n\} \) at the vertices \( \{v_1, \ldots v_n\} \) that are feasible with respect to the battery model, and the state of charge satisfies (2) everywhere along the path. However, for the problem considered in this paper, the path itself is a decision variable; this coupling between the path planning and charge profile planning poses a challenge.

On its own, path planning in the presence of quiet zones resides in a continuous space resulting in an infinite dimensional problem. Sampling the continuous space is a popular technique used to translate the problem to a discrete planning problem. We discretize the problem by sampling the boundaries of the quiet zones, and generate a graph similar to a visibility graph [10], shown in Fig. 2. We address the discrete version of the problem by formulating the path planning and the energy management problem on the discrete graph. It allows us to reduce the infinite dimensional path planning problem into a finite dimensional discrete problem.

There is a loss of optimality using this approach, however, it yields a simplified and potentially tractable approach to solve the coupled infinite dimensional problem. Note that, the loss of optimality depends on the sampling intervals along the boundaries, and therefore the loss could be made sufficiently small with sufficiently large sampling rate. We also develop an algorithm that produces tight lower bounds to this problem, and thereby corroborates the quality of the feasible solutions. **The key idea to obtain this lower bound is to partition the domain of a set of continuous variables, that determine the state of charge at each vertex, into a set of sub-intervals. The state of charge at these vertices is allowed to be discontinuous, i.e. the vehicle can arrive at and exit a vertex with different charge levels; but the two charge levels are forced to lie in one of the sub-intervals. This problem is relaxed compared to the original because we are allowing it to violate the continuity of the state of charge at the vertices.**

**Related literature:** The path planning problems are solved using sampling based algorithms such as RRT [11], RRT* [12], BIT* [13], road-map based search techniques such as PRM methods [14], incremental graph search methods such as D* Lite [15]. Other path planning techniques include visibility graph, Voronoi diagram based or potential field methods [16]. There exists few results in the area of energy aware path planning. In [17], a planning problem is considered where a robot needs to accomplish a set of goals while maintaining a minimum energy threshold. In [18], an energy aware coverage planning problem is addressed, where a robot needs to cover an area that minimizes the total energy consumption. A planning problem to minimize energy consumption in the presence of disturbances is addressed in [19]. The paper uses a model predictive approach to estimate safety critical states, and presents a self triggering schedule to re-plan, and is compared to periodic re-planning. Zhang et al. addressed a route planning for a plug-in hybrid vehicle is addresses in [20] that minimizes energy consumption. The authors address the coupled routing problem that aims to simultaneously optimize the decision making for path and the power management.

The problem considered in this paper addresses the path planning problem for a hybrid aerial robot, in the presence of (noise-)restricted zones. Here, we aim to optimize the travel cost for a robotic vehicle that can switch between two different modes. To the best of our knowledge, a path planning that allows a robot to switch between travel modes, with constraints on the modes in certain regions, has not addressed before in the literature. The application of this novel problem to urban air mobility and drone delivery is of particular relevance and deserves attention. In sampling based and incremental search techniques, a tree is iteratively constructed by following a sampling procedure. To determine the switch points along the path while simultaneously
The main contributions of this work are: (i) we present a novel path planning and energy management problem for a hybrid robot suited for an urban air mobility application, (ii) we formulate the coupled problem on the discrete graph that involves discrete decision making for the path, and continuous variables for the state of charge along the path, (iii) we present a sampling based approach to find near optimal solutions, (iv) we develop a partitioning algorithm to find tight lower bounds to the optimal solution, (v) we validate the presented algorithms using numerical experiments on some benchmark areas of operation.

The rest of the paper is organized as follows. In Section II, a graph is constructed by sampling the boundaries of the quiet zones, and the hybrid path planning is defined on this graph. The algorithms to compute near optimal solutions and tight lower bounds using sampling and partitioning approach, respectively, are presented in Section III. The algorithms presented were tested using computational experiments, the results are presented in Section IV, and some concluding remarks are provided in Section V.

II. PROBLEM FORMULATION

Before formalizing the problem, we define the graph, $G_s$ using Algorithm 1, which takes as inputs the start and goal locations, $v_s$ and $v_t$, a set of quiet zones, $\mathcal{O}$, and a sampling interval, $\delta L$. The boundaries of each quiet-zone are sampled uniformly by choosing a point at every $\delta L$ units of distance. The vertex set, $V_s$, is built by creating a vertex corresponding to these samples, the start and goal positions (steps 4–6 of Algorithm 1). An edge is added if the two vertices are visible: note that, the edges between the vertices that belong to the same quiet zone are added too (step 8). The edge set consists of edges both inside and outside the quiet zones. The sampled graph of the example problem is shown in Fig. 2. The dotted lines inside the quiet zone represent the feasible segments of the path, where the mode of travel is restricted to electric. Though we consider only the polygonal quiet zones in the examples, this method directly applies to non-polygonal quiet zones too. For non-convex shapes, one may consider convex-hull as done in [21], and generate the graph. If one needs to consider obstacles along with quiet zones, it is sufficient to add the vertices of the obstacles to $V_s$, and corresponding edges that does not intersect with the obstacles.

We formulate and solve the path planning and energy management problem using the sampled graph, $G_s(V_s, E_s)$. With this approximation, the path planning reduces to finding an ordered sequence of nodes $S_p := \{v_s, v_{s_1}, \ldots, v_{s_k}, v_t\}$ on graph $G_s$ along with the state of charge of the battery at each of these nodes, $\{q_s, q_{s_1}, \ldots, q_{s_k}, q_t\}$. Let $v_l$ indicate the $l$th node in $S_p$. The following must hold for $S_p$ and the corresponding battery charge at each node $v_l \in S_p$: (i) the state of charges, $q_{s_l}, q_{s_{l+1}}$, on every edge $(v_l, v_{l+1}) \in S_p$ satisfies the battery dynamics (1) and (2), (ii) the segments of the path in the quiet zones are in electric mode, (iii) the state of charge of the battery at the terminal node is greater than a specified value, $q_{goal}$, and (iv) the total cost of travel is minimized. The cost we aim to minimize is the total fuel consumed while traveling in gasoline mode. Therefore, each edge has a zero cost if it is traveled completely in electric mode.

![Fig. 2. A graph generated by sampling the boundaries of the quiet zones.](image-url)

**Algorithm 1 Construction of the sampling graph**

1: function $\text{SamplingGraph}(v_s, v_t, \mathcal{O}, \delta L)$
2: $V_s \leftarrow \text{InitiateNodeSet}()$
3: $E_s \leftarrow \text{InitiateEdgeSet}()$
4: $V_s \leftarrow V_s \cup \{v_s, v_t\}$
5: for $O_i \in \mathcal{O}$ do
6: $V_s \leftarrow V_s \cup \text{DiscreteSampling}(O_i, \delta L)$
7: for $v_i, v_j \in V_s$ do
8: if $\text{CheckEdgeFeasibility}(v_i, v_j)$ then
9: $E_s \leftarrow E_s \cup (v_i, v_j)$
10: $G_s \leftarrow \text{CreateGraph}(V_s, E_s)$
11: return $G_s$

Let $I := \{1, \ldots, |V_s|\}$ be the set of indices of the vertices in $V_s$. For any $i, j \in I$, let $x_{ij}$ denote the binary variable such that $x_{ij} = 1$ if an edge $(v_i, v_j)$ is chosen to be on the path, and $x_{ij} = 0$, otherwise. Given the states of charge of the battery $q_i$ and $q_j$ at the vertices $v_i$ and $v_j$, respectively, let $c_{ij}(q_i, q_j)$ be the cost of travel from $v_i$ to $v_j$. Let $x$ represent a matrix of all binary variables, and $\mathcal{X}$ be the set of all feasible paths from $v_s$ to $v_t$ in $G_s$. The optimization problem, $\mathcal{P}_1$, is stated as the following.

$$\mathcal{P}_1 : \min_{x \in \mathcal{X}, q_i \in [q_{\text{min}}, q_{\text{max}}], \forall k \in I} \sum_{i,j \in I} x_{ij}c_{ij}(q_i, q_j) \tag{3}$$

In the following sections, we present the algorithms that simultaneously optimizes the two sets of variables, $x$ and $\{q_i, i \in V_s\}$. The approach involves sampling of the state of charge at every vertex in $V_s$, constructing a graph $G_u(V_s, E_u)$, and solving a shortest path problem on this graph; the solution to the shortest path problem on $G_u$...
produces a feasible solution (upper bound to optimal solution) to \( \mathcal{P}_1 \). For a given map with quiet zones, the construction of the base graph without the start and goal can be done offline. We only need to add the start and goal vertices and corresponding edges to compute the shortest path, which significantly reduces the online computation time. We also present a partitioning approach, similar to the upper bounding algorithm, that produces tight lower bounds. In this way, we provide both upper and lower bounds to the optimal solution and hence the gap between the two is the maximum gap between the optimal and feasible (upper bound) solutions.

### III. Technical Approach

The algorithms to compute the upper bounds and lower bounds follow a method similar to Algorithm 1 of creating nodes and edges. These algorithms sample and partition the state of charge at each vertex in \( V_u \). In the upper bounding algorithm, the state of charge at each vertex is uniformly sampled, creating a new node for each sample of state of charge at each vertex. In contrast, the lower bounding algorithm partitions the feasible interval of state of charge into small sub-intervals and builds a graph where each node represents a sub-interval of the state-of-charge at each vertex in \( V_u \). This partitioning approach is similar to that found in [22]–[25]. For coupled optimization problems involving both discrete and continuous variables, this approach is found to produce tight lower and upper bounds, and therefore guarantees the quality of the upper bounds with respect to the optimal solution.

#### A. Feasible Solution (Upper bounds)

In problem \( \mathcal{P}_1 \), at any vertex \( v_k \in V_s \setminus \{v_s, v_t\} \), the charge \( q_k \) is a continuous variable and \( q_k \in [q_{\min}, q_{\max}] \). To compute near optimal feasible solutions, for every \( v_k \in V_s \setminus \{v_s, v_t\} \), we choose a discrete set of values, \( Q_k^b \), sampled uniformly in the interval \([q_{\min}, q_{\max}]\). Such a sampling procedure transforms \( \mathcal{P}_1 \) into the discretized problem \( \mathcal{P}_2 \) below:

\[
P_2 : \min_{x \in X, q_k \in Q_k^b \forall k \in I} \sum_{i,j \in I} x_{ij} c_{ij}(q_i, q_j).
\]  

By construction, any solution to \( \mathcal{P}_2 \) is a feasible solution to \( \mathcal{P}_1 \). To solve \( \mathcal{P}_2 \), we construct a graph \( G_u(V_u, E_u) \), where the set of nodes, \( V_u \), consists of all nodes corresponding to every combination of \( v_k \in V_s \setminus \{v_s, v_t\} \) and \( q_k \in Q_k^b \). One can choose the discrete set \( Q_k^b \) to be the same for every node \( v_k \), and let it be \( Q_u \). For the start node, there is only one value \((v_s, q_{\text{init}})\), and the feasible charges for the goal node is sampled from the set \([q_{\text{goal}}, q_{\max}]\); denote this set of charges as \( Q_{\text{goal}} \). Then \( V_u \) contains the Cartesian product of the sets \( V_s \setminus \{v_s, v_t\} \) and \( Q_u \), and the nodes corresponding to the start and goal nodes, that is \( V_u := \{v_s\} \cup \{v_t\} \times Q_u \cup \{(v_s, q_{\text{init}})\} \cup \{(v_t, q_{\max})\} \cup \{(v_s, q_{\text{goal}})\} \). An illustration of the graphs \( G_s \) and \( G_u \) is shown in Fig. 3(a) and 3(b).

For any nodes, \( v_l \in V_u \), let the position corresponding to the node \( v_l \) be \( p_l \) and state of charge of the battery, \( q_l \). We check if a feasible edge exists between a pair of nodes that complies with the battery dynamics (1), and compute the cost of such edge if it exists. For a given pair of nodes \( v_{u_i}, v_{u_j} \in V_u \), where \( v_{u_i} = (v_i, q_i) \), \( v_{u_j} = (v_j, q_j) \), Algorithm 2 checks if edge \((v_{u_i}, v_{u_j})\) is feasible with respect to the battery dynamics. The algorithm returns the cost of the corresponding edge as zero if the travel mode is completely electric. Notice that the gasoline engine runs at full capacity whenever it is turned on, and therefore it is sufficient to find the length of the segments that are traveled in gasoline mode to evaluate the cost. If an edge is traveled in both gasoline and electric modes, the algorithm computes the distance traveled in gasoline mode, \( \lambda d_{ij} \), and returns the fuel cost of travel, \( c_f \lambda d_{ij} \), where \( c_f \) is cost of fuel per unit distance traveled.

#### Algorithm 2 Evaluation of an edge

1: \( \text{function SOCEASIBILITY}(v_{u_i}, v_{u_j}) \)
2: \[ d_{ij} = \mid \text{position}(v_{u_i}) - \text{position}(v_{u_j}) \mid \]
3: if \( q_j > \min(q_{\max}, q_i + 3d_{ij}) \) then \( \text{FeasCheck} \leftarrow \text{false} \)
4: else if \( q_i - \alpha d_{ij} > q_{\min} \land q_j \ge q_i - \alpha d_{ij} \) then \( \text{FeasCheck} \leftarrow \text{true} \)  
5: \( \lambda \leftarrow 0 \)
6: else \( \text{partly gasoline/electric} \)
7: \( \lambda_1 \leftarrow (q_{\max} - q_i)/(\beta \cdot d_{ij}) \)
8: \( \lambda_2 \leftarrow (q_{\max} - q_j)/(\alpha \cdot d_{ij}) \)
9: if \( \lambda_1, \lambda_2 > 0 \land \lambda_1 + \lambda_2 \le 1 \) then \( \lambda \leftarrow 1 - \lambda_1 - \lambda_2 \)
10: else \( \lambda \leftarrow (q_j - (-q_i + \alpha d_{ij})/(\alpha + \beta) \cdot d_{ij}) \)
11: if \( 0 \leq \lambda \leq 1 \) then \( \text{FeasCheck} \leftarrow \text{true} \)
12: \( \text{return FeasCheck, cost} \) \( \lambda \)

Algorithm 2 checks if “all electric” mode is feasible in step 5. The variable \( \lambda \) indicates how much of the edge is traveled in gasoline mode. When an edge is traveled in partly gasoline and partly electric mode, without loss of generality, we assume that the robot travels in gasoline mode first and then switches to electric. Further, we allow a maximum of three switch points on an edge. When an edge is sufficiently long, more than three switch points might be necessary; however, such cases can be accommodated by breaking the long edge into smaller edges by introducing artificial vertices. If an edge is traveled using both modes, the value of \( \lambda \) is computed in steps 9–14, and the corresponding cost is computed in step 18.

To solve the problem \( \mathcal{P}_2 \), we construct graph \( G_u(V_u, E_u) \), such that a shortest path on this graph produces a solution to the problem \( \mathcal{P}_1 \). The pseudocode of the algorithm that solves \( \mathcal{P}_2 \) is presented in Algorithm 3. The problem prescribes a state of charge of the robot at the start, and therefore we can create a node, \( v_{u_s} \), correspondingly, and add to \( V_u \), shown
in step 6, the discrete set, $Q_u$, is obtained by sampling the interval $[\min q, \max q]$. In steps 9–10, for every combination of $(v_i, q_j), v_i \in V_u \backslash \{v_s, v_f\}, q_j \in Q_u$, we create a node and add to $V_u$. A minimum state of charge, $q_{\text{goal}}$, is required at the goal position, and to satisfy that, we sample uniformly in the interval $[\min q, \max q]$, and create the set of nodes, $V_{\text{goal}}$ (shown in step 12), that correspond to the goal position and a state of charge $q_j \in Q_{\text{goal}}$. In steps 14–19, we construct the set of edges, $E_u$, by adding an edge for every pair of nodes in $V_u$, if Algorithm 2 returns a feasible solution; the costs returned by the algorithm are set as the weights of those edges. Note that there could be multiple nodes in $V_{\text{goal}}$, that correspond to the goal position, and satisfy the minimum charge required at goal. Therefore, any path from $v_{u_s}$ to a node in $V_{\text{goal}}$ is a feasible path. To find the minimum cost path, we add another node $v_{u_f}$, that corresponds to the goal, and add zero cost edges between all nodes in $V_{\text{goal}}$ and $v_{u_f}$, shown in steps 20–22. Finally, we use Dijkstra’s algorithm to find the optimal shortest path from $v_{u_s}$ to $v_{u_f}$ in step 24. The time complexity of Dijkstra’s algorithm is $O(|V_u|^2)$, and it returns the minimum cost path on the graph $G_u$. The trajectory for the robot is constructed using the positions of the vertices in the shortest path.

**Algorithm 3** Construction of the graph, $G_u$

```plaintext
1: function SAMPLINGGRAPH($G_u, \delta q$)
2:   $V_u \leftarrow$ INITIATENodeSet()
3:   $E_u \leftarrow$ INITIATEEdgeSet()
4:   $v_{u_s} \leftarrow$ CREATENode($v_{u_s}, \delta q_{\text{init}}$)
5:   $v_{u_f} \leftarrow$ CREATENode($v_{u_f}$)
6:   $V_u \leftarrow V_u \cup \{v_{u_s}\}$
7:   $Q_u \leftarrow$ DISCRETE SAMPLING($[\min q, \max q], \delta q$)
8:   $Q_{\text{goal}} \leftarrow$ DISCRETE SAMPLING($[\min q, \max q], \delta q$)
9:   for $v_i \in V_u, q_j \in Q_u$ do
10:      $V_{\text{goal}} \leftarrow V_{\text{goal}} \cup$ CREATENode($v_i, q_j$)
11:   for $q_j \in Q_{\text{goal}}$ do
12:      $V_{\text{goal}} \leftarrow V_{\text{goal}} \cup$ CREATENode($v_i, q_j$)
13:   $V_u \leftarrow V_u \cup V_{\text{goal}}$
14:   for $v_{u_i}, v_{u_j} \in V_u$ do
15:      if CHECKEDGEFEASIBILITY($v_{u_i}, v_{u_j}$) then
16:         feas, cost $\leftarrow$ SOCFEASIBILITY($v_{u_i}, v_{u_j}$)
17:      if feas then
18:         $E_u \leftarrow E_u \cup (v_{u_i}, v_{u_j})$
19:         $c_{ij} \leftarrow$ cost
20:   $V_u \leftarrow V_u \cup \{v_{u_s}\}$
21:   for $v_{u_k} \in V_{\text{goal}}$ do
22:      $E_u \leftarrow E_u \cup (v_{u_k}, v_{u_f})$
23:   $G_u \leftarrow$ CREATEGRAPH($V_u, E_u$)
24:   path$_u \leftarrow$ SHORTESTPATH($G_u, v_{u_s}, v_{u_f}$)
25: return path$_u$
```

The algorithm returns the shortest path, path$_u$, which is comprised of a series of nodes $\{v_{u_1}, \ldots, v_{u_n}\}$, and each of these nodes correspond to the vertices in $V_u$. This series of vertices in $V_u$ is the path taken by the robot. The states of charge corresponding to each of the nodes in path$_u$ are the charges at corresponding vertices in $V_u$. The feasibility check in step 16 of Algorithm 3 ensures the feasibility of the whole path. Since, the weight of each edge is the cost of travel in gasoline mode, the shortest path is the path of minimum fuel cost. The optimal solution of $P_2$ is a feasible solution to $P_1$, but may not be optimal due to the discrete sampling. A tight lower bound could corroborate the quality of a feasible solution, and an algorithm to compute tight lower bounds is presented in the next section.

![Graph construction to compute the upper bounds and lower bounds to $P_1$](image)

**B. Lower Bounds**

To compute lower bounds, it is a common practice to relax a set of constraints and the optimal solution of the resulting relaxed problem gives a lower bound to the original problem. The Held-Karp bounds for traveling salesman problem [26], is a good example of this technique. In recent work, for coupled problems involving discrete and continuous decision variables, a technique was developed where a set of constraints are relaxed, and another set of ‘loose’ constraints are added, such that it produces tight lower bounds. This was shown to produce very tight lower bounds for routing problem with turn radius constraints [23], [24] and for neighborhood traveling salesman problem [25]. The problem in this paper is also a coupled problem; and we develop an algorithm using a similar idea of partitioning the continuous decision variables, that produces tight lower bounds.
To compute the lower bound, we pose a relaxation of the problem $P_1$. Any feasible solution to $P_1$ would be given as a sequence of nodes $\{v_1, \ldots, v_p\}$, and a state of charge at each of these nodes, $\{q_1, \ldots, q_p\}$. The continuity of the state of charge dictates that at any intermediate node $v_j$ along the path, the state of charge at the end of prior edge is same as the state of charge at the beginning of the following edge. For example, if a feasible solution contains two successive edges, $(v_i, v_j)$ and $(v_j, v_k)$, let $q_{ij}$ be the charge at the end of edge $(v_i, v_j)$, and $q_{jk}$ be the charge at the start of edge $v_jv_k$. The position of the end of the edge $(v_i, v_j)$ is same as the start of the edge $(v_j, v_k)$; therefore, the continuity of the charge profile dictates that $q_{ij} = q_{jk}$. We relax this continuity constraint and allow $q_{ij}$ and $q_{jk}$ to be different but we restrict them to lie in an interval, i.e., $q_{ij}, q_{jk} \in (q_p, q_{p+1})$. We refer to this relaxed problem as $P_3$. Since, this is a relaxation to $P_1$, every feasible solution to $P_1$ is also feasible to the relaxed problem $P_3$. Therefore, the optimal solution of $P_3$ is a lower bound to the optimal solution of $P_1$.

To this end, at every node, we partition the feasible interval of state of charge into $n_l$ sub-intervals. At the vertices $v_i \in V_s \setminus \{v_s, v_t\}$, the set of intervals would be $Q_{i}^l = \{[q_{\min}, q_1], [q_1, q_2], \ldots, [q_{n_l-1}, q_{\max}]\}$. At the goal node, the minimum charge required is $q_{\text{goal}}$, and therefore the intervals would be $\{[q_{\text{goal}}, q_1], [q_1, q_2], \ldots, [q_{n_g-1}, q_{\max}]\}$, for some $n_g$. Let $q_i$ represent an interval of states of charge at a node $v_i$. Now the relaxed problem $P_3$ is stated as follows:

$$ P_3 : \min_{x \in X, q_k \in Q_i^l, \forall k \in I} \sum_{i,j \in I} c_{ij}(q_i, q_j). $$

We solve $P_3$ by constructing a graph, $G_l(V_l, E_l)$, similar to the $G(u, E_u)$. The nodes in $V_l$ are the combination of the vertices $v_i \in V_s$ and the intervals $q_i \in Q_i^l$. For a pair of nodes $v_{i_1}, v_{i_2} \in V_l$, let $(q^e_{i_1}, q^e_{i_2})$ and $(q^m_{i_1}, q^m_{i_2})$ be the corresponding charge intervals. The cost of the corresponding edge in $E_l$ is the minimum cost of the edge,

$$ \min_{q_i \in (q^e_{i_1}, q^e_{i_2}), q_j \in (q^m_{i_1}, q^m_{i_2})} c_{ij}(q_i, q_j). $$

Due to the linear battery dynamics, this cost could be found by using Algorithm 2 with the upper limit of the interval at the first node, and lower limit of the interval at the second node, $q^*_k+1, q^*_m$. The graph construction of $G_l(V_l, E_l)$ is similar to the one presented in Algorithm 3, however it differs only in two aspects: (i) in steps 7–8, the discrete sampling is replaced with continuous partition of the interval $[q_{\min}, q_{\max}]$ and $[q_{\text{goal}}, q_{\max}]$, respectively, and (ii) the edge cost, in step 16, are assigned using the solution of (6). Similar to the steps 10, 12 of Algorithm 3, nodes are created corresponding to a combination of vertices $v_i \in V_s$ and intervals $q_i \in Q_i^l$. The rest of the graph construction goes similar, and therefore, to avoid the repetition, we do not present the pseudocode for the lower bounding algorithm. An illustration of the construction of the graphs $G_u$ and $G_l$ is shown in Figs. 3(b) - 3(c). In the following theorem, we formally prove that the optimal solution to $P_3$ is a lower bound to the optimal solution of $P_1$.

**Theorem 1.** The optimal solution of $P_3$ is a lower bound to the optimal solution of $P_1$.

**Proof.** It is sufficient to show that every feasible solution to $P_1$ is also a feasible solution to $P_3$, and the cost of the solution to $P_3$ is less than or equal to the cost of the solution to $P_1$. Let $V_{f_{\text{feas}}} := \{v_s, v_{s1}, \ldots, v_{s_n}, v_t\}$ be the sequence of vertices in a feasible solution to $P_1$, and $Q_{f_{\text{feas}}} := \{q_s, q_{s1}, \ldots, q_{s_n}, q_t\}$ be the corresponding states of charge at those vertices. For each vertex $v_k \in V_{f_{\text{feas}}}$ and the corresponding state of charge $q_k$, there exists an interval $q_k \in Q_k^l$, such that $q_k \in [q_k]$. Construct a path in $G_l$ by identifying the nodes corresponding to $v_k$ and $q_k$. This gives a feasible path from $v_s$ to $v_t$, in $G_l$, and let $V_{f_{\text{feas}}}$ be the sequence of nodes. The cost of each edge between successive nodes in $V_{f_{\text{feas}}}$ is less than or equal to the cost of corresponding edge in $V_{f_{\text{feas}}}$ due to (6), and thus cost of the path in $V_{f_{\text{feas}}}$ is less than or equal to the one in $V_{f_{\text{feas}}}$. □

**IV. COMPUTATIONAL RESULTS**

To evaluate their performance, we have tested the algorithms to compute the upper bounds and the lower bounds using several scenarios constructed from randomly generated maps and benchmark maps. Since the problem of path planning in the presence of obstacles is closely related to the problem we consider, we use previously established benchmark maps [27] to test our methods. We constructed the maps by randomly generating polygonal restricted zones, where the centers of the polygons are sampled from an uniform distribution. The number of sides are also randomly generated for each restricted zone. There are 10, 15, 20 and 25 restricted zones in $\text{map1}$, $\text{map2}$, $\text{map3}$ and $\text{map4}$, respectively. We have constructed two more maps using the benchmark instances ‘boston2’ and ‘newyork0’ from [27]. These are based on real world maps of a region in the cities Boston and New York; we identified the regions directly above the buildings as restricted zones, and they are appropriate for drone delivery applications as discussed in Section I. An interested reader can access the Julia code to extract the buildings from the OpenStreetMaps data from https://github.com/manyangupta/HybridPathPlanning.git.

We have generated 50 scenarios with each of the above maps, where the start and goal positions are chosen from a random distribution such that the straight line distance between them is greater than a specified limit. Further, the instances are run with different levels of discretization of the states of charge. The rate of discharge, $\alpha$, and the rate of recharge, $\beta$ are chosen such that the ratio $\frac{\alpha}{\beta}$ is equal to two, i.e., the battery discharges twice as fast as it recharges per unit distance traveled.

In Fig. 4(a), a path of the feasible solution is shown for a scenario generated in the ‘newyork0’ map, and the charge profiles of the feasible path and the lower bound are shown in Fig. 4(b). We consider no-fly zones in this scenario, shown in red, and were addressed as explained in Section II. The vertical lines are the positions of the vertices in Fig. 4(b), and one may observe the charge profile is not continuous for
the lower bound path. This is expected due to the relaxation of the charge continuity, and the resulting solution is a lower bound, rather than a feasible solution. For this scenario, the cost of the feasible path produced by Algorithm 3 is 5318 and the lower bound is given as 5137, therefore, the gap between upper bound and lower bound is around 3.5%. This infers that the feasible solution is within less than 3.5% from the optimal solution.

The percent gap between the lower bounds and upper bounds is a measure of the quality of the feasible solutions. This is the maximum gap between the feasible solution and the optimal solution. For each of the maps, there are 50 scenarios, and each scenario is solved with 20, 30 and 40 discretizations. The box plot of the percentage gap is shown in Fig. 5(a). Clearly, with higher sampling rate, the algorithm produces better solutions as evident from the reducing gap with higher discretizations. The maps Boston and New York have a few quiet zones with very small edges, and because of these, the lower bound graph consists of many zero cost edges. For example, let \([q^a, q^b]\) be the charge interval corresponding to the vertices \(v_k\) and \(v_l\) in \(G_u\). If the charge required to travel on the edge \((v_k, v_l)\) is less then \(q^b - q^a\), this edge can be on a path with zero cost, and have same charge at the start and end of the edge. This resulted in loose lower bounds, and hence the higher gap. However, the algorithm produces tighter lower bounds by choosing sufficiently large number of sub-intervals \(n_l\) while constructing the graph \(G_u\). But, this comes at a higher cost in computational time required. The computation time required to find the upper bounds and lower bounds are shown as box and whisker plots in Figs. 5(b) and 5(c). The higher computation times for the Boston and New York maps is due to the higher number of quiet zones.

A significant part of the computational effort in Algorithm 3 is spent constructing the graph \(G_u(V_u, E_u)\). However, for UAM applications the restricted zones, start and end locations are known a priori and can be computed ahead of time. For other applications, like package delivery, the restricted zones and the problem parameters are known a priori, but the positions of the start and goal may not be known. In practice, one may construct the parts of the graph \(G_u(V_u, E_u)\) offline without the nodes corresponding to the start and goal position, and edges incident on them. And when the robot’s start and goal positions are specified, the corresponding edges of the graph could be constructed and added to the graph with Algorithm 3. Therefore, to validate the feasibility of online implementation of this algorithm, it is sufficient to analyse the computational effort of the online part. The online computation time required by Algorithm 3 is shown in Fig 5(b). Though the effort required increases with higher number of restricted zones and higher sampling rate, it is still in the order of seconds, and therefore, is viable for on-board implementation.

To evaluate the cost savings from the proposed framework, we solved the related but different path planning problem where the quiet zones are considered to be no-fly zones (i.e., feasible paths must completely avoid the quiet zones). This is done by removing the quiet zone edges from the graph \(G_u\), and solving for the shortest path thereafter. Note that, we still consider the hybrid mode of the robotic vehicle, and the switching between the gasoline and electric mode still exists. If we were to restrict this to gasoline mode only, the savings would be much larger. The percentage reduction in cost using Algorithm 3 is presented in Fig. IV. The cost savings are around 5 to 10% for most cases, with some cases having much higher savings. This large variance is due to the randomly generated start and goal locations; the cost reduction depends on the difference in length between the shortest path that avoids the “no-fly zones” and path generated from our work that passes through the restricted zones.

V. CONCLUSIONS

A novel hybrid path planning problem that arises from urban air mobility is presented. In this path planning problem, the hybrid vehicle is required to run in electric mode in certain regions to comply with noise restrictions. The path planner needs to generate a path and schedule for switching between gasoline and electric modes. Algorithms based on sampling and partitioning are presented yielding upper and lower bounds to the coupled path planning and energy management problem. The paper assumes a linear battery model, but a higher fidelity model could be easily integrated with the algorithms presented here. The solutions produced by the presented algorithms are empirically shown to yield upper and lower bounds that are within 15% of one another, indicating that the feasible solutions are of high quality.
the path. Boundaries of the quiet zones that chooses higher number thus overcome the cost of higher sampling. Another direction is an iterative scheme to compute lower bounds that refines the quiet zones.

Fig. 6. Percentage cost reduction compared to path planning that avoids quiet zones.

As a future research direction, one may develop an iterative scheme to compute lower bounds that refines the partitioning in each iteration only where it is necessary, and thus overcome the cost of higher sampling. Another direction of future research includes the adaptive sampling of the boundaries of the quiet zones that chooses higher number of samples on the boundaries that are more likely to be on the path.

REFERENCES

[1] S. Hasan, “Urban air mobility (UAM) market study,” NASA, Tech. Rep., 2019.
[2] M. D. Patterson, D. R. Isaacsen, N. L. Mendonca, N. A. Neogi, K. H. Goodrich, M. Metcalfe, B. Bastedo, C. Metts, B. P. Hill, D. DeCarne, et al., “An initial concept for intermediate-state, passenger-carrying urban air mobility operations,” in AIAA Scitech Forum, no. 1626, 2021.
[3] K. Button, “Faith in batteries,” Aerospace America, pp. 36–42, Oct. 2021.
[4] V. Smil, “Decarbonization Algebra: The COP26 Calls for Impossibly Steep Cuts in Carbon Emissions: Numbers Don’t Lie,” IEEE Spectrum, vol. 59, no. 2, pp. 20–21, Feb. 2022.
[5] S. A. Rizzi, D. L. Huff, D. D. Boyd, B. S. Henderson, K. A. Pascom, D. C. Sargent, D. L. Josephson, M. Marsan, H. B. He, et al., “Urban air mobility noise: Current practice, gaps, and recommendations,” NASA, Tech. Rep., 2020.
[6] W. J. Fredericks, M. D. Moore, and R. C. Busan, “Benefits of hybrid-electric propulsion to achieve 4x cruise efficiency for a VTOL UAV,” in International Powered Lift Conference, no. 4324, 2013.
[7] R. Cabell, F. Grosveld, and R. McSwaian, “Measured noise from small unmanned aerial vehicles,” in Inter-Noise and Noise-Conf Congress and Conference Proceedings, vol. 252, no. 2. Institute of Noise Control Engineering, 2016, pp. 345–354.
[8] H. D. Kim, A. T. Perry, and P. J. Ansell, “A review of distributed electric propulsion concepts for air vehicle technology,” in AIAA/IEEE Electric Aircraft Technologies Symposium (EATS). IEEE, 2018, pp. 1–21.
[9] J. Lieh, E. Spahr, A. Behbahani, and J. Hoying, “Design of hybrid propulsion systems for unmanned aerial vehicles,” in 47th AIAA/ASME/SAE/IEEE Joint Propulsion Conference & Exhibit, 2011, p. 6146.
[10] W. J. Fredericks, M. D. Moore, and R. C. Busan, “Benefits of hybrid propulsion systems for unmanned aerial vehicles,” in 47th AIAA/ASME/SAE/IEEE Joint Propulsion Conference & Exhibit, 2011, p. 6146.
[11] J. Kuffner and S. LaValle, “RRT-connect: An efficient approach to single-query path planning,” in Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), vol. 2, 2000, pp. 995–1001.
[12] S. Karaman and E. Frazzoli, “Sampling-based algorithms for optimal motion planning,” The International Journal of Robotics Research, vol. 30, no. 7, pp. 846–894, 2011.
[13] J. D. Gunnell, T. D. Barfoot, and S. S. Srinivasa, “Batch informed trees (BIT*): Informed asymptotically optimal anytime search,” The International Journal of Robotics Research, vol. 39, no. 5, pp. 543–567, 2020.
[14] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” IEEE Transactions on Robotics and Automation, vol. 12, no. 4, pp. 566–580, 1996.
[15] S. Koenig and M. Likhachev, “Fast replanning for navigation in unknown terrain,” IEEE Transactions on Robotics, vol. 21, no. 3, pp. 354–363, 2005.
[16] O. Souissi, R. Benatitallah, D. Duvivier, A. Artiba, N. Belanger, and P. Feyzeau, “Path planning: A 2013 survey,” in Proceedings of 2013 International Conference on Industrial Engineering and Systems Management (IEMS), 2013, pp. 1–8.
[17] T. X. Lin, E. Yel, and N. Bezzo, “Energy-aware persistent control of heterogeneous robotic systems,” in American Control Conference (ACC), 2018, pp. 2782–2787.
[18] C. Di Franco and G. Buttazzo, “Energy-aware coverage path planning of UAVs,” in IEEE International Conference on Autonomous Robot Systems and Competitions, 2015, pp. 111–117.
[19] N. Bezzo, K. Mohta, C. Nowzari, I. Lee, V. Kumar, and G. Pappas, “Online planning for energy-efficient and disturbance-aware UAV operations,” in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016, pp. 5027–5033.
[20] Q. Zhang, K. Wu, and Y. Shi, “Route planning and power management for PHEVs with reinforcement learning,” IEEE Transactions on Vehicular Technology, vol. 69, no. 5, pp. 4751–4762, 2020.
[21] H. Huang and A. V. Savkin, “Viable path planning for data collection robots in a sensing field with obstacles,” Computer Communications, vol. 111, pp. 84–96, 2017.
[22] S. G. Manyam, S. Rathinam, D. Casbeer, and E. Garcia, “Tightly bounding the shortest Dubins paths through a sequence of points,” Journal of Intelligent & Robotic Systems, vol. 88, no. 2, pp. 495–511, 2017.
[23] S. G. Manyam and S. Rathinam, “On tightly bounding the Dubins traveling salesman’s optimum,” Journal of Dynamic Systems, Measurement and Control, vol. 140, no. 7, p. 071013, 2018.
[24] S. Rathinam, S. G. Manyam, and Y. Zhang, “Near-optimal path planning for a car-like robot visiting a set of waypoints with field of view constraints,” IEEE Robotics and Automation Letters, vol. 4, no. 2, pp. 391–398, 2019.
[25] P. Váha and J. Faigl, “On the Dubins traveling salesman problem with neighborhoods,” in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 2015, pp. 4029–4034.
[26] M. Held and R. M. Karp, “The traveling-salesman problem and minimum spanning trees,” Operations Research, vol. 18, no. 6, pp. 1138–1162, 1970.
[27] N. Sturtevant, “Benchmark for grid-based pathfinding,” Transactions on Computational Intelligence and AI in Games, vol. 4, no. 2, pp. 144–148, 2012.