Why Shorter Isn't More Efficient: A Physics-Based Approach to Multi-Target Drone Routing

Max Ratcliff

March 19, 2026

Abstract

Traditional autonomous routing solutions typically optimize for distance or time. However for energy-constrained systems like drones, non-linear forces inherent to flight like drag, wind, and hover consumption mean that the shortest path is not necessarily the most energy efficient. In this paper we present a physics-based energy model that accounts for these forces, and use it to compare a physics-aware route optimization to a naive distance-based solution to the Traveling Salesman Problem (TSP). First we find the energy optimizing speed to fly between targets (with respect to the external forces), and then use this energy cost as edge weights to solve the TSP which becomes asymmetric due to massively different energy costs due to the direction of the wind. Finally we can compare a variety of optimization methods that solve the asymmetric TSP both exactly to find the exact optimal solution, and approximately to find near-optimal solutions with lower compute. Our results show that the physics-aware optimization can reduce energy consumption considerably compared to a distance-based TSP solution, and that approximate methods such as Nearest Neighbor with 2-opt can achieve near-optimal performance while remaining computationally efficient for larger target sets.

1 Introduction

In many applications like search-and-rescue, delivery, and multi-point infrastructure inspection, a drone must visit a set of targets and return to its origin. Because these drones are often expensive and running out of energy can be catastrophic, it is crucial to optimize their routes to minimize energy consumption. This optimization is done in two stages: Optimizing the flight path of each route segment to minimize energy, and optimizing the route taken to reach every target. This second stage is typically modeled as a Traveling Salesman Problem (TSP), where the goal is to find the shortest possible route that visits each target exactly once and returns to the origin. However, for drones "shorter" doesn't necessarily mean "more efficient". Unlike terrestrial vehicles, a drone often has less inertia making it significantly more affected by forces like drag and wind. In addition drag often scales with speed and drones being high speed vehicles often mean they face significantly more drag and wind forces than a car might. Drones also have to spend a significant amount of energy just hovering, so flying too slow can waste energy as opposed to saving it by minimizing external forces. Furthermore, the presence of wind creates a directional asymmetry in the energy cost between targets, since flying with the wind can save energy while flying against it can be significantly more expensive. This leads to an important question: To what extent can a physics-based routing model that accounts for wind, drag, and hover expenditure, reduce total energy consumption compared to a naive distance-based TSP solution?

1.1 Roadmap

To answer this question, we first develop a physics-based energy model that accounts for the effects of drag, wind, and hover power on the energy consumption of a drone flying between two targets. Then we compare a variety of optimization methods to solve the resulting TSP, including both exact and approximate methods. Finally we validate our model and optimization methods, and present results comparing the total energy consumption of the different optimization methods for a set of targets in a 2D domain with a constant wind vector.

2 Physical Model

First we begin with the problem of optimizing the drone's flight path between two targets on a two dimensional plane to minimize energy consumption. To do this we must first define a velocity profile for the drone to follow, and then use this velocity profile to compute the forces acting on the drone so that we can compute the power and energy required to fly between the two targets. For a given path between two targets there are many ways to assign a velocity profile the only constraint being the drone must start and end at rest at each target, but the choice of velocity profile will affect the forces acting on the drone and therefore the energy consumption.

2.1 Kinematic Profile

To reflect realistic drone applications where the drone has to stop at each target to perform some task, and to keep our model simple so that we can analytically validate it and optimize it, we prescribe a smooth parabolic velocity profile of the form

$$v(t) = \alpha t(T - t),$$

where T is the total travel time for the segment and α is a scaling constant. This profile ensures the drone smoothly accelerates and decelerates between each target. Enforcing the distance constraint so that the drone travels the pre-defined distance to the next target gives

$$d = \int_0^T v(t)\,dt = \int_0^T \alpha t(T - t)\,dt = \frac{\alpha T^3}{6} \implies \alpha = \frac{6d}{T^3}.$$

We can then compute the acceleration profile as

$$a(t) = \frac{dv}{dt} = \alpha(T - 2t).$$

2.2 Forces and Energy Model

The drone operates in a 2D plane with a constant wind vector w and experiences linear aerodynamic drag with coefficient C. In this environment the drone must generate enough thrust to overcome both its own weight as well as the forces of wind and drag to maintain a stable flight path. Assuming a linear drag model, the thrust force required to maintain the flight path is:

$$F(t) = ma(t) + C(v(t) - w),$$

Where m is the mass of the drone, and $v(t) - w$ is the relative velocity of the drone with respect to the wind, which determines the drag force. The total power is then modeled as the sum of the constant hover power required to maintain lift and the power required to generate the thrust to overcome drag and wind:

$$P(t) = P_{\text{hover}} + |F(t) \cdot v(t)|.$$

Which means we can integrate this power over the duration of the trip to get the total energy required to traverse a segment of length d:

$$E(d, T) = \int_0^T P(t)\, dt.$$

This model captures the fundamental trade off in drone flight, where flying too slow wastes energy hovering, and flying too fast causes a huge increase in drag, using this model we can find the ideal trip time for each segment. Furthermore, by incorporating the wind vector into the model, we create an asymmetric energy costs between targets, causing one direction $E(d_{ij}, T)$ to be more expensive than the other $E(d_{ji}, T)$, which allows us to take advantage of the wind to reduce energy consumption turning our TSP into an Asymmetric TSP (ATSP), which is more complex to solve but allows us to find more energy efficient routes.

2.3 Physical Model Validation

To validate the physical model, we can set the drag coefficient to 0 which means the power is just the hover power and the work required to get from one target to the next, which can be solved analytically as

$$W = \int_0^T ma(t)v(t)dt = 2.25m\frac{d^2}{T^2}.$$

Which we can compare to the numerical integration given by the trapezoidal rule to confirm that our numerical integration when we set the drag and wind to 0 is within 0.001% of the analytical solution (all of the other parameters were left the same as our other results).

We can also validate the physical model by verifying that the energy $E(d, T)$ exhibits the expected U-shaped curve as a function of travel time T for a fixed distance d.

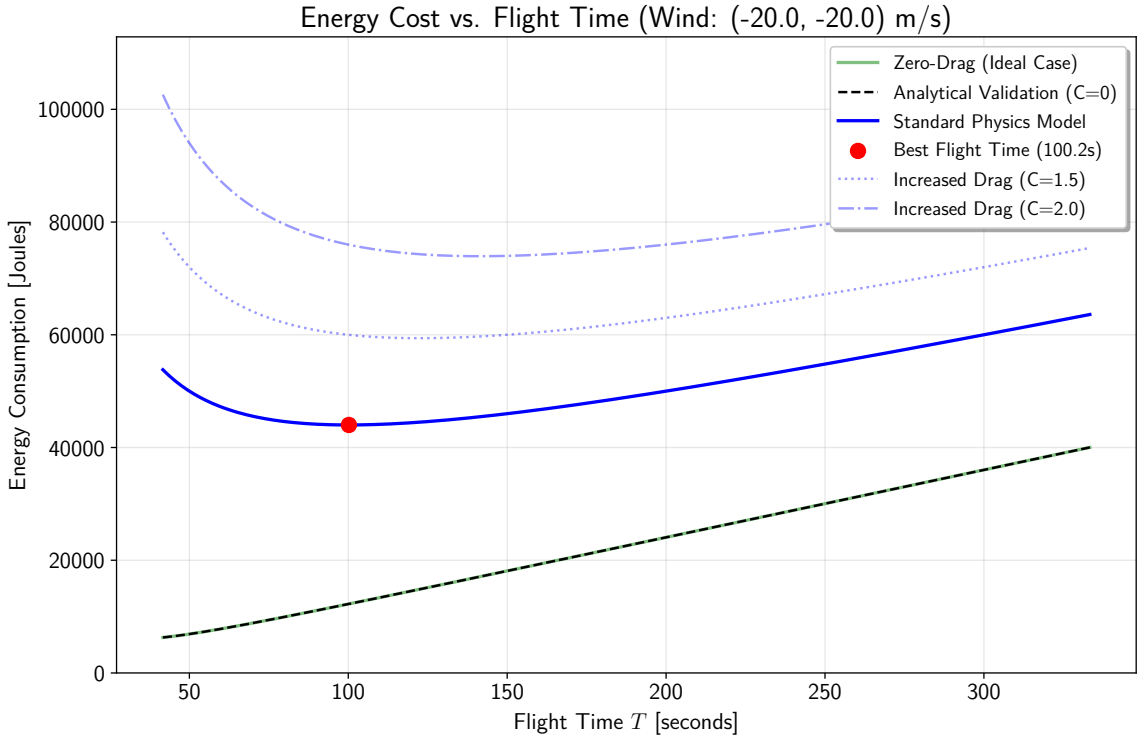

Figure 1: Segment energy $E(d, T)$ vs. trip time T with a fixed distance ($d = 1000$m). The minimum indicates the optimal travel time T^* that balances thrust and hover power. Standard Physics model Parameters: $m = 1.00$ kg, $C = 1.00$ N/(m/s), $P_{\text{hover}} = 120.0$ W , Zero drag and analytical validation both use the same parameters but with $C = 0$ and the analytical solution for energy. The increased drag curves also use the same parameters but with $C = 1.5$ and $C = 2.0$ respectively to exaggerate the effect of drag on the energy curve.

At low T, the drone must fly very fast to cover the distance, which increases drag and therefore energy, the effect of this can be exaggerated by increasing the drag coefficient which causes the energy cost to rise faster with higher speed. At high T, the drone flies very slowly, which means it spends more time hovering and wastes energy. The optimal travel time T^* that minimizes energy occurs at the minimum of this curve, confirming that our model behaves as expected and that the optimization will succeed in finding a minimum.

3 Optimization Methods

The routing optimization is done in two stages: first we compute the optimal segment energy for every pair of targets. Then we use this computed energy matrix as edge weights to solve the Traveling Salesman Problem (TSP) using various methods.

3.1 Segment Energy Optimization

Because the energy function $E(d, T)$ is non linear and we want to find the time that minimizes energy like in figure 1, we use a numerical optimization method to find the optimal travel time T^* for each segment. To keep the simulation realistic and to get bounds to use for our numerical optimization we use reasonable physical constraints $v_{\max} = 18m/s$ and $a_{\max} = 6m/s^2$ to set the

bounds on the optimization, this we only consider physically feasible trajectories and stop the drone from accelerating infinitely and prevent it from flying so fast it risks damage. Time bounds are calculated by finding the time at which the velocity and acceleration profiles reach these limits

$$T_{\text{low}} = \max\left(\sqrt{\frac{6d}{\alpha v_{\max}}}, \sqrt{\frac{6d}{\alpha a_{\max}}}\right),$$

and then we give the drone a reasonable amount of time to complete the segment by setting the upper bound as:

$$T_{\text{high}} = \max\left(4T_{\text{low}}, 0.7d\right).$$

Finally we minimize $E(T)$ over this feasible interval $[T_{low}, T_{high}]$ to find the optimal travel time T^* for that segment. T^* is the flight duration that gives us the optimal flight speed. This way the drone doesn't waste energy in hovering by flying too slow or lose too much energy to drag by flying too fast. We repeat this optimization process for every pair of targets to construct an $N \times N$ energy matrix where each entry $E^*(d_{ij})$ represents the minimum energy required to travel from target i to target j for all N targets.

3.2 Route Optimization

To find the most energy optimal route through all the targets we implement and compare several optimization methods with varying computational complexity and accuracy. Because we included wind in the model to compute the energy matrix it is important to keep in mind that the energy to get from target i to target j may not be the same as the energy to get from target j to target i.

- **Systematic Search (Brute Force):**

  For small N, we can simply compute the total energy for every permutation of routes, and select the route that uses the least total energy. This provides an absolutely optimal path, but is only feasible to compute on our hardware for $N < 11$ since the algorithmic complexity is $O(N!)$.

- **Held Karp:**

  Because we start and end at the same point our traveling salesman problem is a hamiltonian cycle which means we can use the Held-Karp algorithm [2] which keeps track of the optimal path of each subset to avoid re-computing known segments. This allows us to compute the exact global minimum for up to $N < 25$ with an algorithmic complexity of $O(N^2 2^N)$.

- **Nearest Neighbor (NN):**

  In Nearest Neighbor (NN)[3] starting from an initial target, we repeatedly select the next unvisited target requiring the least energy to reach from the current position, until all targets are visited. This is a greedy algorithm that can't look ahead to see if it is working against itself, which causes it get stuck pursuing the wrong paths, but is incredibly fast compared to the exact methods with an algorithmic complexity of $O(N^2)$. This allows us increase the problem size to much larger sets of targets.

- **Nearest Neighbor + 2-opt:**

  The 2-opt improvement [1] makes local improvements to the nearest neighbor route by using a seed route from the NN algorithm. Then iteratively re-inserting each target into every possible position in the route. A swap is accepted only if it reduces the total route energy. The process finishes when a full pass produces no further improvement. The 2-opt improvement also has an algorithmic complexity of $O(N^2)$ but requires the Nearest Neighbor solution to be computed first so it is always slower.

While the exact methods always provide the global minimum, they come at a significant computational cost. In contrast, the approximate NN and NN+2opt methods offer much more scalability in exchange for potentially not finding the optimal solution. Figure 2 compares the runtime of these different optimization methods as a function of the number of targets N, demonstrating the trade off between optimality and computational efficiency. The figure illustrates how quickly the exact brute force or Held-Karp methods become infeasible, and show how while Nearest Neighbor and the 2opt improvment both scale at the same rate, the 2-opt improvement is always slower by a constant factor since it relies on the seed route.

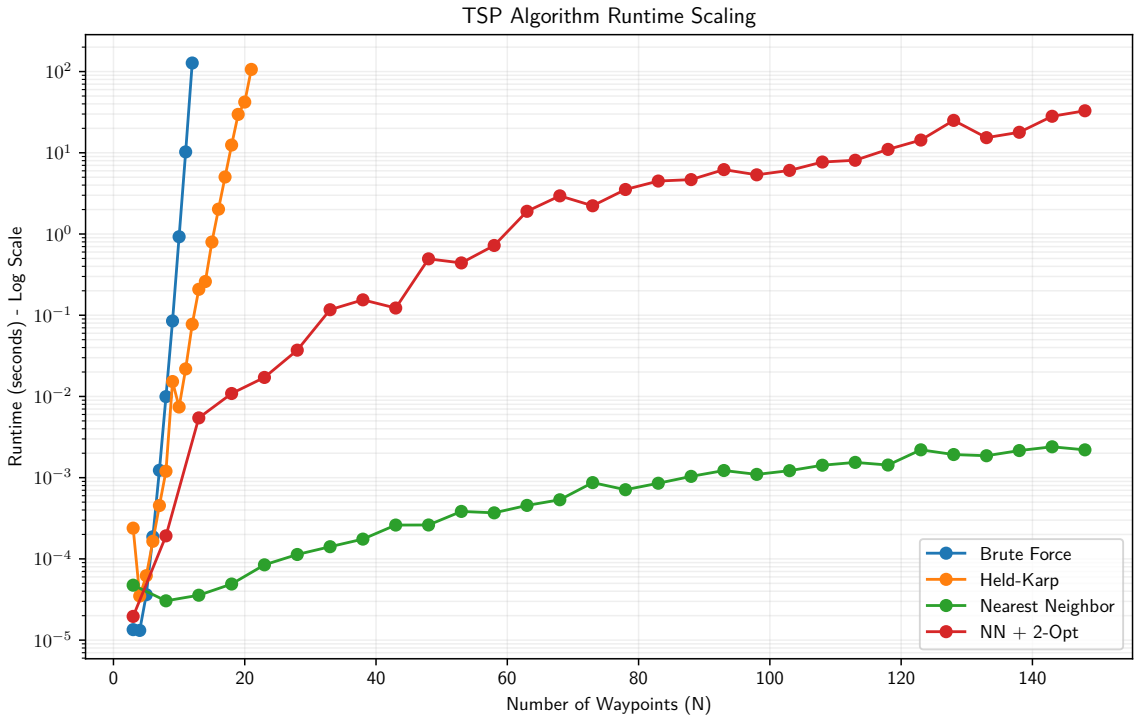

Figure 2: Runtime comparison of TSP optimization methods as a function of target count N. The brute force method becomes infeasible beyond $N = 10$ due to $O(N!)$ algorithmic complexity, while Held-Karp is viable up to $N = 25$ with its $O(N^2 2^N)$ complexity. The Nearest Neighbor and NN+2opt algorithms scale much more favorably at $O(N^2)$, allowing them to handle larger target sets efficiently. All benchmarks were run on an M2 Mac, and each method was tested on the same set of random target configurations.

3.3 Optimization Method Validation

Since a systematic or brute force search is the simplest method to understand and implement, we used it as a benchmark against the Held-Karp algorithm to confirm that both of the exact methods produce the exact same route and energy minimum for all the target set sizes that could be reasonably computed with brute force. We generated 10 random configurations for each target set size from $N = 3$ to $N = 10$, and directly compared the list order and the energy to confirm that they were identical for both methods, which gives us confidence that both of these methods are correctly implemented and producing the global minimum energy route for these small target sets. We also confirmed that the approximate methods never produce a lower energy route than the global minimum for these small target sets, which gives us confidence that they are also performing

as expected.

Targets (N)	Brute Force Cost (J)	Held-Karp Cost (J)	Order
3	76.9855	76.9855	*Identical*
4	147.0061	147.0061	*Identical*
5	169.4157	169.4157	*Identical*
6	219.2157	219.2157	*Identical*
7	238.9384	238.9384	*Identical*
8	264.1546	264.1546	*Identical*
9	237.6267	237.6267	*Identical*
10	260.7734	260.7734	*Identical*

Table 1: Validation of Exact Solvers: Brute Force vs. Held-Karp. All trials demonstrate a 100% match in energy cost, confirming the mathematical integrity of the dynamic programming approach.

4 Results

Our results indicate that incorporating a physics based energy model into route planning significantly reduces total energy consumption compared to naive distance based TSP algorithms, and that algorithms like as NN+2opt can achieve near-optimal performance while remaining computationally efficient.

4.1 Multi Target Route Optimization

To isolate the effect of the route optimization we compare the total route energy for the different optimization methods using the same initial conditions, so that the only difference is the route optimization method. Figure 3 compares the total route energy for a set of 20 targets using the same parameters as in the segment validation in Figure 1.

- The sequential route which is just the order the targets were generated in produces substantially higher total energy consumption due to inefficient ordering that results in long jumps between targets and numerous path crossings that waste energy.
- The exact (Held-Karp) solution provides the global minimum energy route, fully eliminating crossings and traveling with the wind to minimize drag.
- The distance based 2-opt TSP solution produces a route that is much better than the naive sequential ordering, but still higher energy than either of the energy aware solutions, since it treats the problem as a symmetric TSP and completely ignores the effect of wind.
- The Nearest Neighbor algorithm produces a route that is better than the sequential ordering, but still significantly higher energy than the exact solution or 2-opt improvement, due to its greedy nature that can lead to suboptimal local choices.
- The Nearest Neighbor + 2opt improvement closely matches the optimal energy, significantly improving upon the Nearest Neighbor route while remaining computationally efficient. The energy aware route is on average $2 - 5\%$ more energy efficient than the distance optimized route calculated using the same algorithm. Demonstrating that optimizing for energy rather than distance is advantageous.

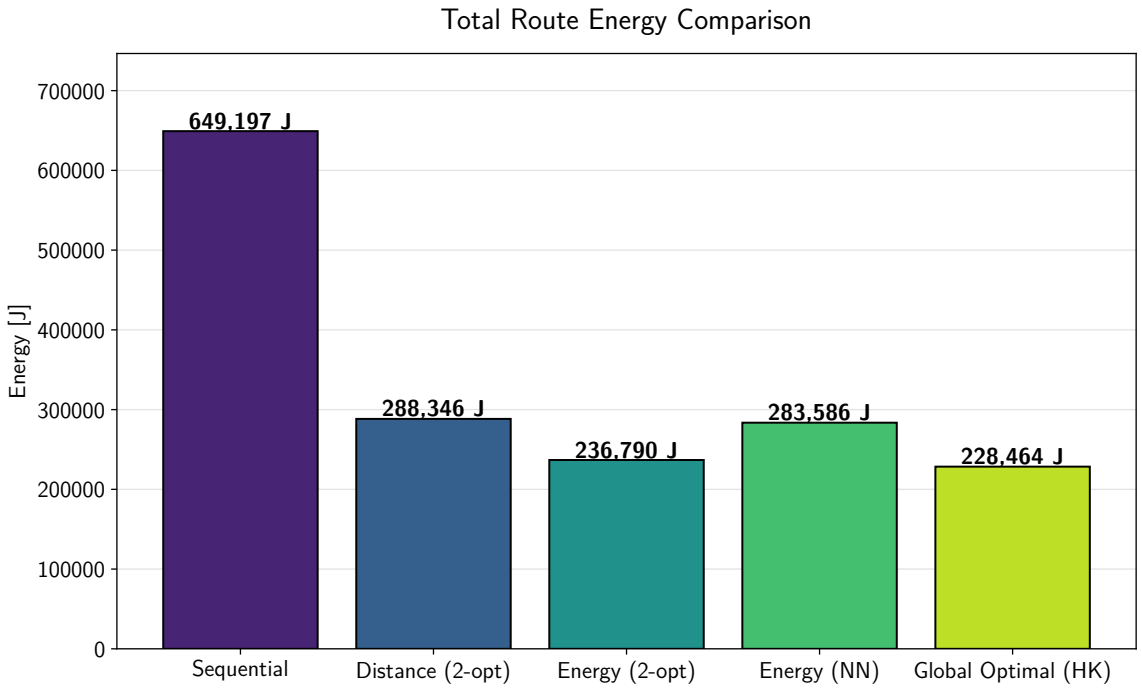

Figure 3: Total route energy comparison for N targets (in this case 20) in a 2D domain with coordinate bounds $(x, y) \in [0, 2000] \times [0, 2000]$ meters. Total route energy is computed as the sum of optimal segment energies along the route. Parameters: $m = 1.0$ kg, $C = 1.00$ N/(m/s), $P_{\text{hover}} = 120.0$ W, a diagonal wind vector = [-20, -20] m/s, and a random seed of 76 which produced a particularly challenging target configuration to highlight the differences between the optimization methods. Bars correspond to the energy cost of the naive target ordering, the distance based TSP solver, the exact minimum-energy route found by held karp, the Nearest Neighbor, and the NN+2opt route.

These results show that when scaled to high target missions, the energy-aware optimization methods produce significantly lower total energy routes compared to both a naive first-come-first-serve ordering, or a distance-based TSP solution that ignores the advantages of wind and disadvantages of drag.

4.2 Route Visualization

plotting the routes generated by the different optimization methods as shown in Figure 4 provides further insight into how the energy aware optimization takes advantage of the wind physics to reduce energy. For the same simulation as Figure 3, we can see that the distance based TSP solution produces a tight loop that minimizes distance, but fights the wind often, while the energy aware solutions produce much more efficient routes that take wider loops to take advantage of wind and reduce drag, additionally we can visually see how the 2-opt improvement removes crossings and long jumps from the initial Nearest Neighbor route and achieves a route much closer to the global optimum. However it still suffers from the inherent greedy nature of the Nearest Neighbor seed route, and fails to identify the other direction around the loop that the Held-Karp algorithm found which minimizes energy by flying with the wind for longer.

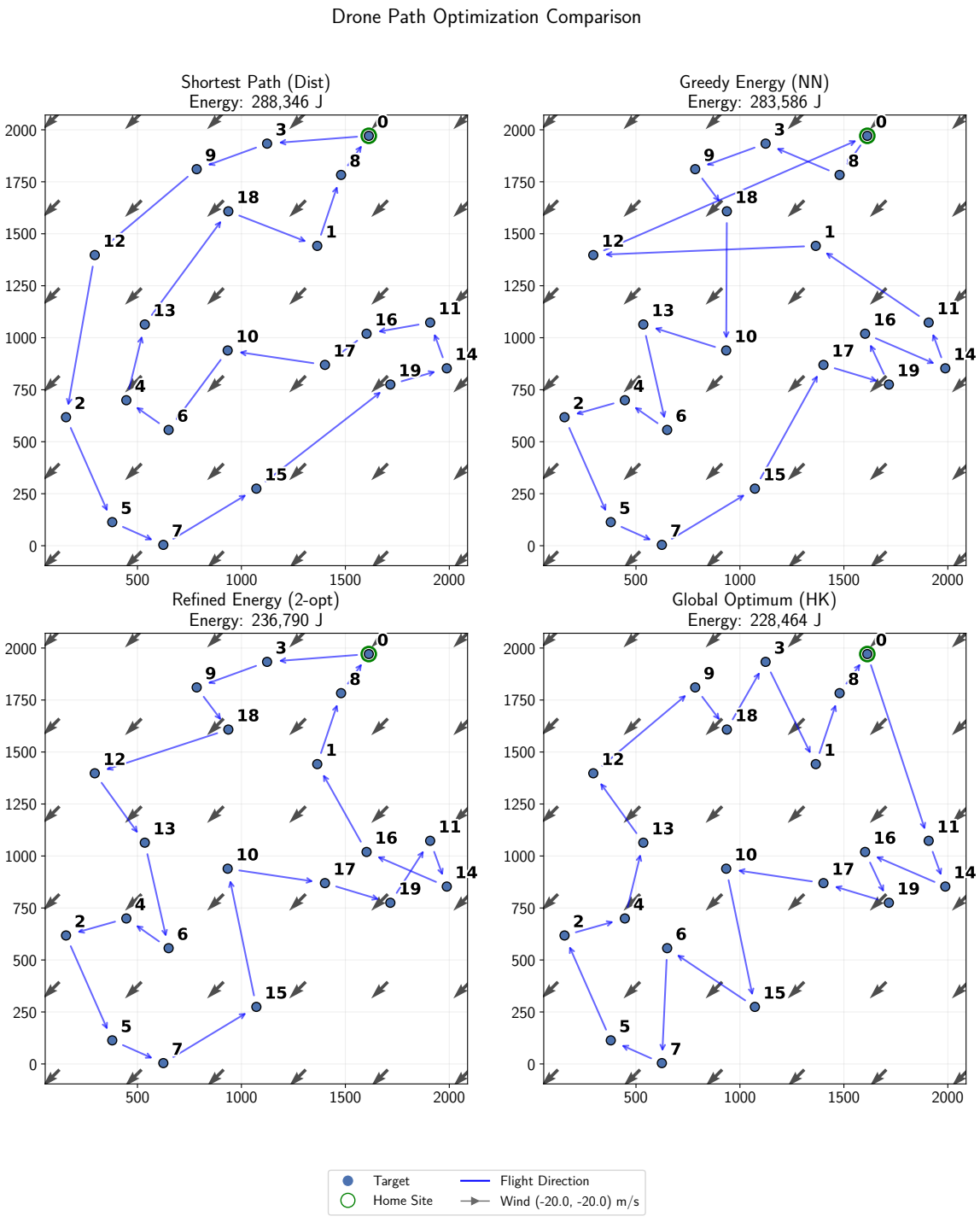

Figure 4: Route visualizations for the same target set used in Figure 3, with coordinates generated within $(x, y) \in [0, 2000] \times [0, 2000]$ meters. Top Left: distance based NN+2opt. Top Right: exact minimum-energy route obtained via held karp. Bottom Left: Nearest Neighbor. Bottom Right: NN+2opt improvement route. Blue arrows indicate direction of travel, and grey arrows represent the direction of the wind. The way points are numbered in the order they are generated. Routes are evaluated using the same energy model parameters as in Figure 1 and Figure 3.

4.3 Statistical Analysis and Algoritm Performance

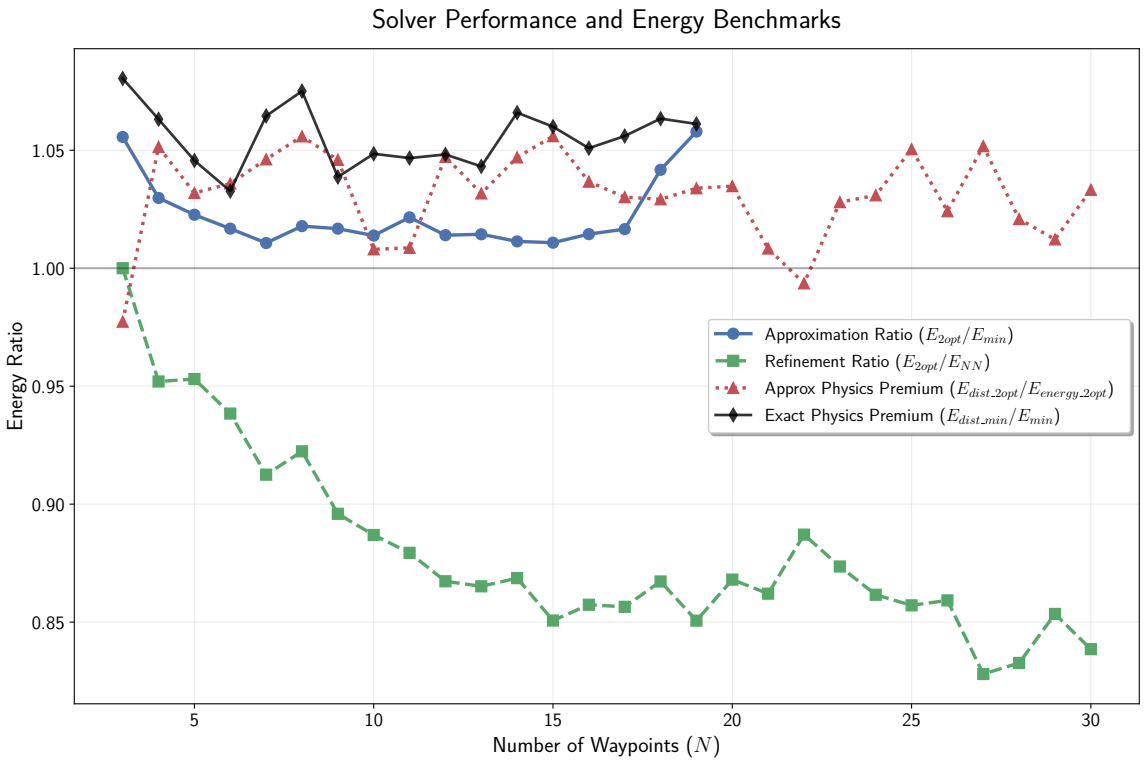

Figure 5: the approximation ratio and refinement ratio for the NN and NN+2opt improvement compared to the exact solution as well as the physics premium or gain across 10 random target configurations for each target set size N. The approximation ratio quantifies how much more energy the approximate solutions consume compared to the exact minimum, while the refinement ratio measures the improvement of NN+2opt over the Nearest Neighbor seed route. The Physics premium lines compare two of the same optimzation methods, both the exact Held-Karp and Approximate 2-opt improvement comparing the results when optimizing for distance compared to energy.

To further evaluate the performance of the different optimization methods we performed a monte carlo analysis where we generated 10 different configurations of varying target set sizes from $N = 3$ to $N = 20$, and then compared the "Approximation Ratio", and the "Refinement Ratio" for the different methods. The Approximation ratio is defined as the percentage increase in energy compared to the exact solution, and the Refinement Ratio is the percentage decrease in energy when comparing the 2-opt algorithm to the NN algorithm.

As illustrated in Figure 5, the approximation ratio for the NN algorithm remains relatively low even for target sets up to $N = 20$, only about 2% higher than the exact solution, however it begins to increase with N, which suggests that even the 2-opt algorithm begins to perform worse at higher target counts. In contrast the NN+2opt algorithm maintains a larger refinement ratio getting over 15% showing that despite additional computational cost the 2-opt improvement is valuable and the fact the refinement ratio is growing faster with time than the approximation ratio demonstrates that the the NN solution is getting worse as the problem gets more complex, but the 2-opt improvement is able to fix a lot of those problems and get much closer to the optimal solution. The physics premium lines show that optimizing for energy rather than distance consistently yields lower energy

routes, although the gap here remains relatively constant as N increases staying between $2-5\%$ for the most part but getting up to 8.55% better highlighting the importance of incorporating physical energy models into route optimization for drones.

5 Discussion and Conclusion

Our results demonstrate that incorporating a physics-based energy model into route planning, can improve energy consumption by up to 8.55% compared to a distance-based TSP solution. They also show that greedy approximate methods such as NN+2opt can achieve near-optimal performance while remaining computationally efficient for larger sets of targets.

While our model provides a significant step towards more physcally accurate energy aware route optimization, we made several simplifying assumptions such as restricting motion to two dimensions, ignoring altitude, assuming linear drag, treating hover power as constant, and not modeling battery dynamics such as discharge nonlinearity. In reality, many drones have significant three-dimensional motion, nonlinear aerodynamic drag, fly in complex wind fields, and in many applications are constantly changing payloads which significantly affects the energy model.

Future work could extend the model to three-dimensional motion, incorporate nonlinear aerodynamic drag, include wind fields, or impose battery discharge constraints. Further investigation into even more scalable optimization methods for large target sets would also be valuable, as well as testing the model and optimization methods on real drone hardware to validate the energy savings in practice.

Ultimately as drones become tasked with increasingly complex missions, and battery technology continues to lag behind the energy demands of these drones, optimizing for energy rather than simply distance or time will become increasingly important for maximizing the utility of this form of transportation. And our work shows that even a simple physics-based routing model can reduce total energy consumption compared to a naive distance-based TSP solution.

References

[1] G. A. Croes. A method for solving traveling-salesman problems. *Operations Research*, 6(6):791–812, 1958.

[2] Michael Held and Richard M. Karp. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied Mathematics*, 10(1):196–210, 1962.

[3] Daniel J. Rosenkrantz, Richard E. Stearns, and Philip M. Lewis, II. An analysis of several heuristics for the traveling salesman problem. *SIAM Journal on Computing*, 6(3):563–581, 1977.