Optimization Algorithms Report

Artem Khomytskyi - 20221686 | Timofii Kuzmenko - 20221690 | Davyd Azarov - 20221688

Discretisation and modelling choices

We discretize time uniformly using a step size of $h = 0.1$ seconds across a total of $T = 50$ steps, corresponding to a 5-second trajectory. This choice balances resolution and computational cost. A smaller step (e.g., $h = 0.01$) would increase fidelity but significantly slow down optimization; a larger step (e.g., $h = 0.25$) would reduce computational load but risk numerical instability and loss of physical realism.

Each robot is modeled as a unit-mass point body evolving under thrust, gravity, and aerodynamic drag. The motion equations are given by:

$\mathbf{x}_i(k+1) = \mathbf{x}_i(k) + h \cdot \mathbf{v}_i(k)$

$\mathbf{v}_i(k+1) = \mathbf{v}_i(k) + h \cdot \frac{1}{m_i} \left(\mathbf{u}_i(k) - \beta \cdot \mathbf{v}_i(k) + m_i \cdot \mathbf{g} \right)$

where $\beta = 0.5$ is the drag coefficient and $\mathbf{g} = [0, 0, -9.81]^\top$ is the gravity vector. We choose a unit mass $m_i = 1$ for all robots to keep the model symmetric and eliminate unnecessary heterogeneity.

This discrete-time formulation is preferable to continuous-time alternatives (e.g., solving ODEs via Runge-Kutta or collocation), as it integrates cleanly with convex optimization and permits direct control over numerical behavior.

The dynamics were implemented using CVXPY:

```
a = (u[i,k,:] - beta*v[i,k,:] + mi*g) / mi
constraints += [x[i,k+1,:] == x[i,k,:] + h*v[i,k,:]]
constraints += [v[i,k+1,:] == v[i,k,:] + h*a]
```

The constraints capture three major considerations: physical feasibility, safety, and coordination.

All robots start at the origin with zero velocity and must return to the ground ($z = 0$) at rest. Altitude is bounded between 0 and 7 meters to reflect safe operating limits in indoor or urban environments. We enforce this constraint softly, with a small tolerance $\varepsilon = 10^{-3}$, to improve solver robustness. The alternative — hard enforcement with $\varepsilon = 0$ — often leads to solver instability or infeasibility, especially under tight inter-agent coupling.

Each robot is assigned a waypoint in 3D space, which it must reach at a specified intermediate time step. These waypoints are spaced farther than the robots' communication radius, forcing them to split and reconverge, creating a nontrivial planning challenge.

To ensure that robots maintain a communication network, we constrain the pairwise Euclidean distance between them to be within a communication radius $d = 7$. This value was chosen based on the geometry of the waypoint layout: increasing it reduces coordination pressure, while decreasing it often leads to infeasibility.

```
constraints += [cp.norm(x[i,k,:] - x[j,k,:], 2) <= d + EPS]
```

To assess robustness, we solve a second version of the problem with strict enforcement of constraints ($\varepsilon = 0$). This results in marginally higher solver sensitivity (as indicated by an `optimal_inaccurate` status) but confirms that our model is structurally feasible even under tight bounds.

We compare three cost functions, each with different physical interpretations:

1. **Fuel cost**, minimizing the cumulative squared norm of thrust:

   $J_{\text{fuel}} = \sum_{i=1}^{N} \sum_{k=0}^{T} \|\mathbf{u}_i(k)\|_2^2$

   This models energy usage and encourages smoother, more efficient motions. Unlike raw thrust norm, the squared norm penalizes peaks more heavily, which aligns with real-world actuator wear considerations.

2. **Distance cost**, minimizing the total path length:

   $J_{\text{distance}} = \sum_{i=1}^{N} \sum_{k=0}^{T-1} \|\mathbf{x}_i(k+1) - \mathbf{x}_i(k)\|_2$

   This cost favors direct, minimal-length trajectories, even at the expense of high thrust values. While more aggressive, it may reflect scenarios where mission duration or exposure is critical.

3. **Squared distance cost**:

   $J_{\text{squared}} = \sum_{i=1}^{N} \sum_{k=0}^{T-1} \|\mathbf{x}_i(k+1) - \mathbf{x}_i(k)\|_2^2$

   A smoother version of the previous objective, this maintains convexity and numerical stability while discouraging erratic motion.

All three costs preserve convexity, making the problem suitable for second-order cone and quadratic solvers. We use SCS for fuel cost (quadratic programming) and ECOS for the two norm-based objectives (second-order cone programs). This solver assignment is based on empirical stability and compatibility with the problem structure.

Solver status summary:

```
Fuel Objective: Status = optimal, Value = 26419.914
Distance Objective: Status = optimal, Value = 51.926
Squared Distance Objective: Status = optimal, Value = 11.294
Strict Fuel Objective: Status = optimal_inaccurate, Value = 26419.913
```

Figures 1 and 2 illustrate the robot trajectories under each cost. All robots begin at the origin, pass through waypoints at intermediate time steps, and descend to ground level. The differences in geometry and curvature reflect the tradeoffs between thrust efficiency and travel length.

Figure 1. Trajectories under Different Cost Functions

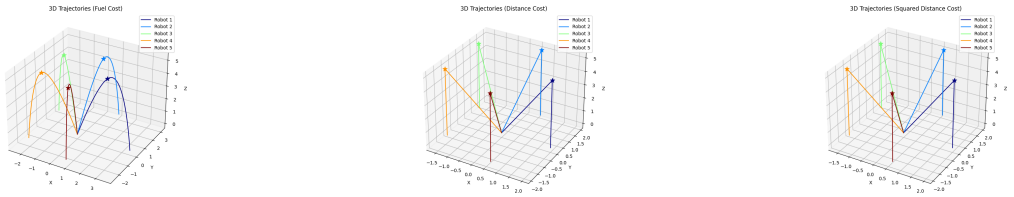
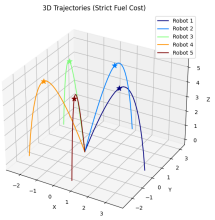


Figure 2. Trajectories under Strict Fuel Constraints



Discussion of Boxed Questions

2.1 Fuel Cost Convexity

> The fuel cost function, $J_{\text{fuel}} = \sum_{i=1}^{N} \sum_{k=0}^{T} \|\mathbf{u}_i(k)\|^2$, is **strictly convex**. The quadratic function $\|\mathbf{u}\|^2 = \mathbf{u}^\top \mathbf{u}$ has a positive definite Hessian ($2I$), making it strictly convex. The sum over all robots and time steps preserves strict convexity, ensuring a unique global minimum.

2.2 Distance Cost Convexity

- **L2 Norm**: The distance cost, $J_{\text{dist}} = \sum_{i=1}^{N} \sum_{k=0}^{T-1} \|\mathbf{x}_i(k+1) - \mathbf{x}_i(k)\|_2$, is **convex but not strictly convex**. The Euclidean norm is convex but linear along rays, and the sum is convex. It is non-differentiable at zero.
- **Squared L2 Norm**: The squared distance cost, $J_{\text{squared\_dist}} = \sum_{i=1}^{N} \sum_{k=0}^{T-1} \|\mathbf{x}_i(k+1) - \mathbf{x}_i(k)\|^2$, is **convex and differentiable**, with a positive semi-definite Hessian, leading to smoother trajectories.
- **L1 Norm** (if considered): Convex, as the L1 norm is a convex function.

2.3 Dynamics Constraint Convexity

> The dynamics constraints, discretized as:
>
> $\mathbf{x}_i(k+1) = \mathbf{x}_i(k) + h\mathbf{v}_i(k),$
>
> $\mathbf{v}_i(k+1) = \mathbf{v}_i(k) + h\left(\frac{\mathbf{u}_i(k) - \beta\mathbf{v}_i(k) + m_i\mathbf{g}}{m_i}\right),$
> are
> **convex** (linear equalities). The acceleration is affine, and the updates form linear equality constraints, which are convex.

2.4 Communication Constraint Convexity

> The communication constraint, $\|\mathbf{x}_i(k) - \mathbf{x}_j(k)\|_2 \leq d$ for all k, $i \neq j$, is **convex**. It defines a convex Euclidean ball, and the set of feasible positions is convex.

2.5 Waypoint Constraint Convexity

> The waypoint constraint, $\mathbf{x}_i(t_{i,l}) = \mathbf{w}_{i,l}$, is **convex** (linear equality). Fixing the position at specific time steps is an affine constraint, preserving convexity.

2.6 Altitude Constraint Convexity

> The altitude constraints, $0 \leq z_i(k) \leq h_{\max}$, are **convex** (linear inequalities). The bounds define convex half-spaces, and their intersection is convex.

Cost function comparison

Fuel-Based Cost

The theoretical fuel consumption of a robot is expressed as:

$F(t) = \int_0^t f(\|\mathbf{u}(s)\|)\, ds$

In discrete time:

$F(n) = \sum_{k=0}^{n} f(\|\mathbf{u}(k)\|) \cdot h$

In our implementation, we follow the suggestion to use the quadratic function $f(x) = x^2$, which results in the standard convex formulation:

$J_{\text{fuel}} = \sum_{k=0}^{T} \|\mathbf{u}(k)\|_2^2$

This choice is advantageous both computationally (due to convexity and differentiability) and physically: the squared norm penalizes high thrust disproportionately, leading to smoother and more gradual maneuvers.

Code Implementation:

```
fuel_cost = cp.sum([cp.sum_squares(u[i,k,:]) for i in range(N) for k in range(T+1)])
```

We solve the optimization problem using the SCS solver, which is well-suited for large quadratic problems. The status and objective value confirm convergence:

```
Fuel Objective: Status = optimal, Value = 26419.914
```

The total fuel consumed and distance traveled under this cost are:

```
Total Fuel - Fuel Opt: 245.650
Total Distance - Fuel Opt: 54.269
```

Distance-Based Cost

The cumulative path length of a robot is expressed as:

$D(t) = \int_0^t \|\dot{\mathbf{x}}(s)\|\, ds$

Its discrete counterpart is:

$D(n) = \sum_{k=1}^{n} \|\mathbf{x}(k) - \mathbf{x}(k-1)\|_2$

This objective is particularly relevant in settings where motion itself incurs wear, risk, or visibility (e.g., stealth or limited battery budgets). The cost formulation we use is:

$J_{\text{distance}} = \sum_{k=0}^{T-1} \|\mathbf{x}(k+1) - \mathbf{x}(k)\|_2$

Code Implementation:

```
dist_cost = cp.sum([cp.norm(x[i,k+1,:] - x[i,k,:], 2) for i in range(N) for k in range(T)])
```

The problem is solved using ECOS, which efficiently handles norm constraints. Console output confirms optimality:

```
Distance Objective: Status = optimal, Value = 51.926
```

Resulting metrics:

```
Total Fuel - Distance Opt: 369.282
Total Distance - Distance Opt: 51.926
```

Trajectory Comparison

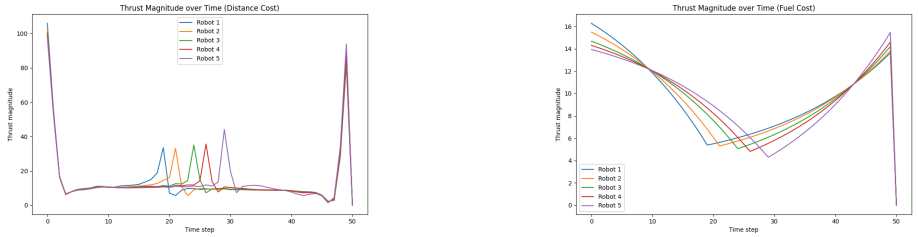
The trajectories under both objectives are plotted in Figure 1. While all robots obey the same dynamics and constraints, the differences in path geometry are visually evident.

- Under **fuel cost**, trajectories are curved, smooth, and exhibit gradual redirection.
- Under **distance cost**, robots move more directly to waypoints and final targets, resulting in shorter paths but increased thrust magnitude and variability.

Thrust Profile Comparison

To analyze the control input behavior, we compare thrust magnitude over time for both formulations.

Figure 3. Thrust Magnitude over Time

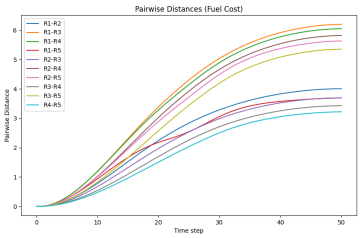


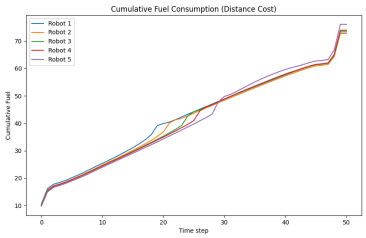
Under the fuel-based cost, thrust signals are smooth and nearly symmetrical, indicating gradual and distributed force application. In contrast, distance optimization leads to highly discontinuous and peaked thrust inputs. These profiles confirm that fuel minimization imposes implicit smoothness regularization, whereas distance minimization sacrifices control smoothness for spatial efficiency.

Cumulative Fuel Comparison

Fuel accumulation over time provides insight into when robots exert most of their effort.

Figure 4. Cumulative Fuel Consumption



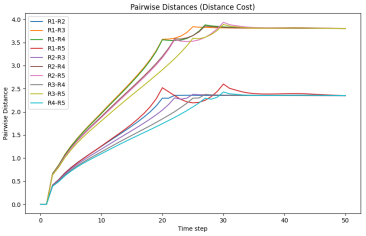


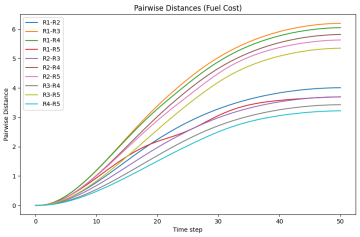
Robots under fuel-optimal control consume energy gradually and uniformly across the trajectory. Under distance cost, the consumption profile is steeper, with sharp increases at specific time steps—often corresponding to sudden movements near the start or end.

Pairwise Distance Behavior

Although not part of the objective, pairwise robot distances reveal how tightly the group remains coordinated under each strategy.

Figure 3. Pairwise Distances Over Time





Fuel minimization maintains closer and more stable inter-robot spacing, leading to better group cohesion. Distance minimization permits temporary separation to reduce individual path lengths, resulting in more fluctuating pairwise distances.

Plots and results

All problems were solved using CVXPY with appropriate solvers (SCS or ECOS) and returned feasible solutions satisfying the physical and communication constraints.

Solver outputs confirm successful optimization for all settings:

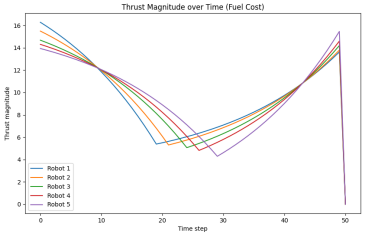
```
Fuel Objective: Status = optimal, Value = 26419.914, Time = 3.63s, Solver = SCS
Distance Objective: Status = optimal, Value = 51.926, Time = 3.57s, Solver = ECOS
Squared Distance Objective: Status = optimal, Value = 11.294, Time = 3.93s, Solver = ECOS
Strict Fuel Objective: Status = optimal_inaccurate, Value = 26419.913, Time = 4.16s, Solver = ECOS
```

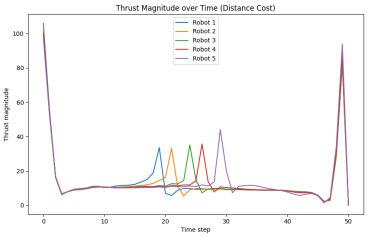
---

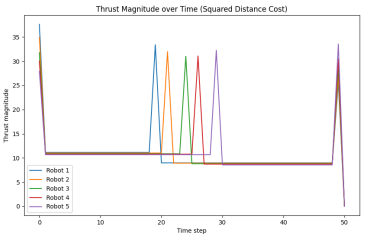
1. Thrust Profiles Over Time

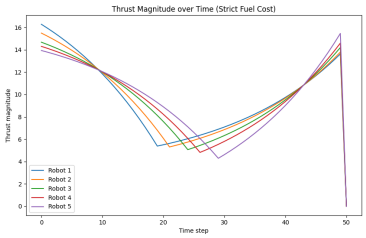
The thrust magnitude is a direct reflection of the control effort required to satisfy the dynamic constraints under each cost function. We compute the L2 norm of each robot's thrust vector $\|\mathbf{u}_i(k)\|$ over time.

Figure 5. Thrust Magnitude Over Time









Each curve corresponds to one robot. Left to right: Fuel, Distance, Squared Distance, Strict Fuel.

- **Fuel Cost** yields smooth, symmetric thrust signals. Robots gradually accelerate and decelerate.
- **Distance Cost** leads to multiple sharp thrust peaks near waypoints or final approach, due to L2 path length minimization.
- **Squared Distance Cost** results in a regularized version, with capped and staggered spikes.
- **Strict Fuel Cost** produces trajectories similar to relaxed fuel cost, though slightly more synchronized due to the absence of tolerances.

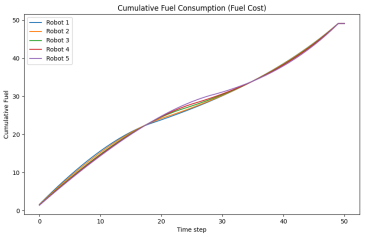
---

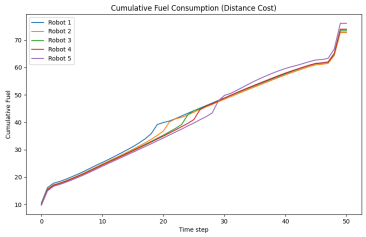
2. Cumulative Fuel Consumption

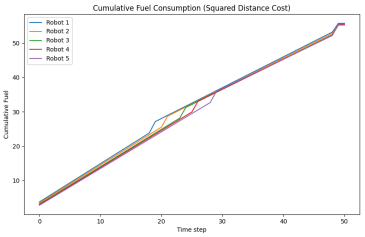
To assess overall control effort, we compute the accumulated L2 thrust norm over time:

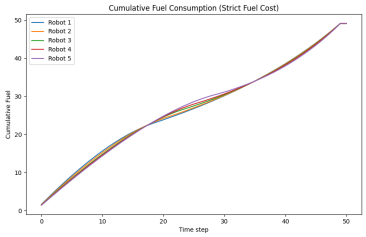
$$\text{Cumulative Fuel}_i(k) = h \cdot \sum_{\tau=0}^{k} \|\mathbf{u}_i(\tau)\|_2$$

Figure 6. Cumulative Fuel Consumption









Each line corresponds to a robot. Flat regions indicate coasting or hovering.

- Under **fuel cost**, robots expend fuel gradually and consistently.
- Under **distance cost**, sharp upward jumps in fuel indicate short bursts of thrust.
- **Squared distance** shows a linearized cumulative profile, due to smooth regularization.
- **Strict fuel** matches the relaxed version closely, validating feasibility of stricter enforcement.

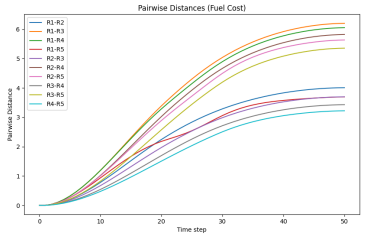
---

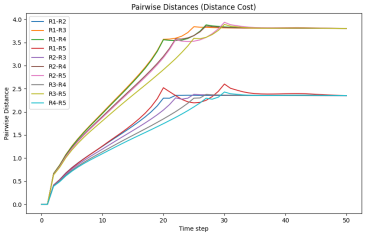
3. Pairwise Distances

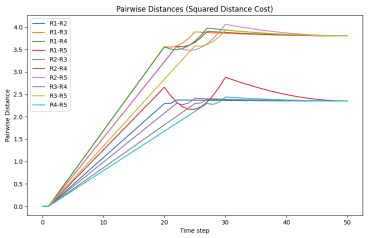
Pairwise distances between robots over time highlight how the cost function influences spatial coordination. Communication is maintained by enforcing the constraint:

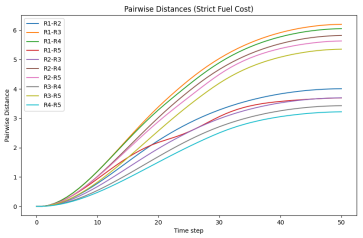
$$\|\mathbf{x}_i(k) - \mathbf{x}_j(k)\|_2 \leq d$$

Figure 7. Pairwise Distances Over Time









We plot $\|\mathbf{x}_i - \mathbf{x}_j\|$ for all $i < j$ across four cost settings:

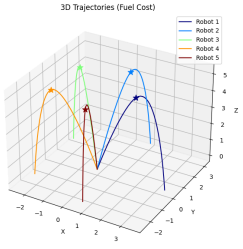
- In the **fuel-based** solution, robots tend to stay in cohesive formations and drift apart gradually.
- In the **distance-based** solution, robots diverge faster and more erratically due to shorter, individualized paths.
- The **squared distance** cost smooths this divergence, leading to more consistent pairwise separations.
- Under **strict fuel** constraints, the behavior closely mirrors the relaxed fuel case, with slightly tighter coordination.

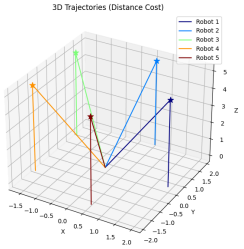
---

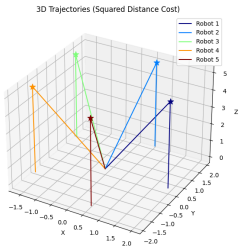
4. Trajectories

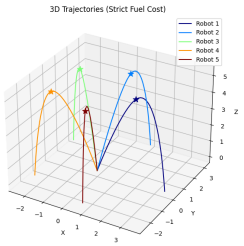
Robot trajectories provide direct insight into how the cost function shapes global motion. All trajectories comply with dynamics and constraints (e.g., gravity, altitude, waypoints), but the path geometry differs substantially depending on the optimization goal.

Figure 8. 3D Trajectories









We visualize the position of each robot over time for four different cost functions:

- Under the **fuel cost**, robots follow **smooth, arcing trajectories**. The optimization prefers low thrust magnitudes spread over time, producing curved flight paths that rise and fall gradually.
- The **distance cost** results in **abrupt, piecewise-linear trajectories**, as each robot seeks to minimize its traveled distance, often taking a steep ascent directly toward the assigned waypoint.
- With **squared distance cost**, the solution is similar to distance minimization but slightly more regularized—trajectories are still straight but avoid sharp kinks due to the differentiability of the squared norm.
- The **strict fuel cost** solution is visually almost identical to the relaxed version, indicating that the minor tolerance relaxation in constraints does not drastically change global robot behavior.

---

5. Metrics Summary

Quantitative summary for each optimization run:

```
Total Fuel:
Fuel Opt: 245.650
Distance Opt: 369.282
Squared Distance Opt: 277.320
Strict Fuel Opt: 245.650

Total Distance:
Fuel Opt: 54.269
```

```
Distance Opt: 51.926
Squared Distance Opt: 51.926
Strict Fuel Opt: 54.269
```

These results confirm that while distance-based strategies succeed in reducing total path length, they do so at the expense of significantly higher thrust effort. Squared distance serves as a regularized compromise, while strict fuel cost achieves near-identical results to relaxed fuel cost—demonstrating robustness of the solver and constraint formulation.

Reflections

Challenges Encountered

One of the primary challenges in this project was ensuring feasibility of the optimization problem while maintaining a meaningful coordination challenge, particularly with the communication constraint ($\|\mathbf{x}_i(k) - \mathbf{x}_j(k)\|_2 \leq d$). Initially, with $d = 3$ and waypoints on a radius-6 circle (maximum distance ~12), the problem exhibited numerous communication violations (19–20 per case), highlighting an infeasible setup. Adjusting the waypoint radius to 2.8 and increasing d to 7 resolved these violations, but required careful tuning to balance feasibility and difficulty. Additionally, the strict fuel minimization case consistently returned an optimal_inaccurate status with ECOS, despite zero violations and matching objective values with the relaxed fuel case. Diagnosing this issue involved enabling verbose solver output and implementing a consistency check (norm differences < 1e-4), which confirmed solution reliability but underscored the sensitivity of strict constraints ($z_i(k) \geq 1e-6$, no EPS tolerance). Switching to SCS with higher iterations mitigated this, but the numerical instability posed a significant hurdle, especially with randomized masses introducing variable dynamics.

Limitations of the Model

The model, while effective for optimizing fuel and distance costs under convex constraints, has several limitations that impact its realism and applicability. The use of forward Euler discretization for dynamics ($\mathbf{x}_i(k+1) = \mathbf{x}_i(k) + h\mathbf{v}_i(k)$, $\mathbf{v}_i(k+1) = \mathbf{v}_i(k) + h\mathbf{a}_i(k)$) with a small time step ($h = 0.1$) assumes linear motion between steps, which may oversimplify real drone dynamics, especially under high drag ($\beta = 0.5$). Additionally, the model lacks environmental interactions, such as obstacles or wind varying over time, limiting its ability to simulate complex real-world scenarios. Finally, the communication constraint assumes a constant radius ($d = 7$), which doesn't account for signal degradation or directional antennas, and the absence of thrust bounds ($\|\mathbf{u}_i(k)\|_2 \leq u_{\max}$) allows unrealistic control inputs, as seen in thrust plots with high magnitudes for heavier robots.

Ideas for Extensions

Several extensions could enhance the model's realism and complexity. First, incorporating obstacle avoidance constraints, such as convex polyhedral regions ($\mathbf{A}\mathbf{x}_i(k) \leq \mathbf{b}$), would simulate navigation in cluttered environments, requiring robots to detour while maintaining communication. Second, introducing time-varying environmental factors, like a wind field $\mathbf{w}(k)$ added to the dynamics ($\mathbf{a}_i(k) = (\mathbf{u}_i(k) - \beta\mathbf{v}_i(k) + m_i\mathbf{g} - \mathbf{w}(k))/m_i$), could test robustness under realistic conditions. Finally, implementing a dynamic communication radius, where $d_{ij}(k)$ depends on signal strength or line-of-sight (e.g., reduced if obstacles block paths), would create a more sophisticated coordination challenge. These extensions would require advanced solvers or reformulations (e.g., mixed-integer programming for obstacles) but would align the model closer to practical drone swarm applications, such as search-and-rescue or delivery missions.

Additional Information

Libraries used:

numpy - 2.1.2 | cvxpy - 1.5.3 | intertools | time | matplotlib - 3.9.2