Distributed Optimisation for Exploration and Trajectory Planning of UAVs

Logan Sritharan

Aims

- Explore an unknown environment **efficiently**
- Single-agent systems are **unsuitable** - low area coverage, ground robots dependant on terrain, not robust
- Multi-agent systems (i.e. UAVs) are increasingly popular due to the wide range of applications
  - Planetary Exploration: extraterrestrial life, valuable resources, human habitation
  - Search and Rescue missions
  - Autonomous 3D Mapping and Photogrammetry
- Centralised computing scales poorly with the number of agents ⟶ Horizontal scaling via distributed computing

We focus on developing a **distributed optimisation** algorithm for **real-time** domain exploration using **Model Predictive Control**

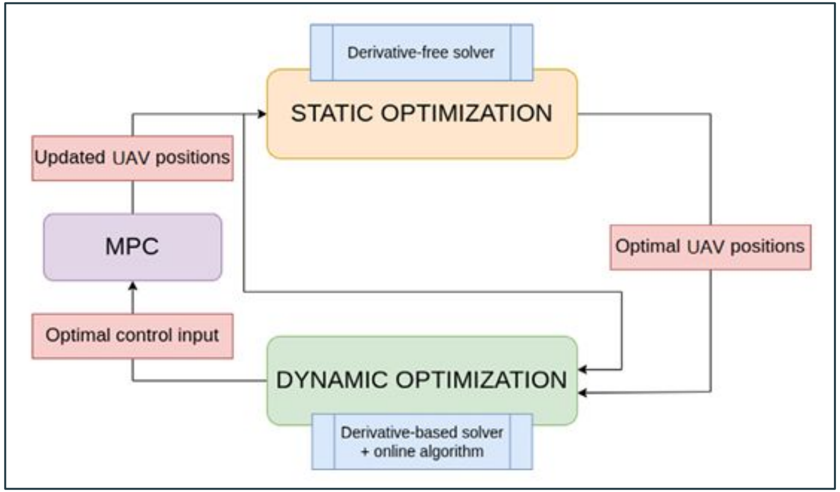
Literature Review

- **Path Planning**
  - Unmanned Aerial Vehicle Dynamic Path Planning in an Uncertain Environment; **Min Yao et al, 2014**
  - Distributed Model Predictive Control for Unmanned Aerial Vehicles; **Sina Sharif Mansouri et al, 2015**
  - Decentralized Model Predictive Control of Cooperating UAVs; **Arthur Richards et al, 2004**
- **Exploration**
  - Time-Efficient Mars Exploration of Simultaneous Coverage and Charging Multiple Drones; **Yuan Chong et al, 2020**
  - Dynamic Pathfinding for Swarm Intelligence Based UAV Control Model using Particle Swarm Optimisation; **Lewis Pyke et al, 2021**

Majority of the work out there focus on exploration and path-planning **independently**; in addition, exploration is tackled via **heuristic** or **statistic**-based method

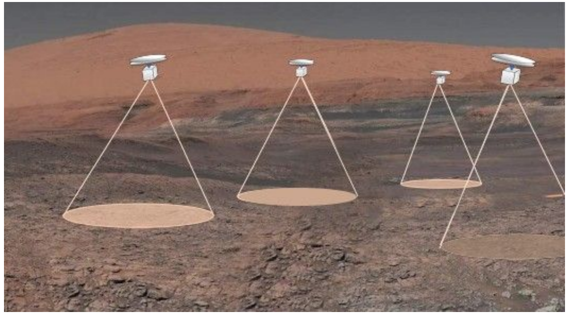
Linking Exploration ↔ Path Planning: Hierarchical Optimisation

- <u>Exploration ↔ Static Optimisation</u>: What are the idea MAV positions in 3D space such that we maximise the mapped area?
- <u>Path Planning ↔ Dynamic Optimisation</u>: What are the ideal trajectories to the static optimisation solution such that we minimise important metrics (i.e. battery consumption, time-of-flight, collision distance, etc.)



Problem Formulation - Exploration

- Each MAV's exploration horizon is represented as a circle centred at the (x,y) location, with the radius as a function of altitude
- **Interesting challenging problem**: computing the exact area of the union of an arbitrary number of circles $\longrightarrow$ **non-convex problem**
- Problem solved using graph-theoretical concepts: "Greens Method"
- Optimised via **Mesh-Adaptive Direct Search** $\longrightarrow$ **fast, derivative-free** (required!)



Algorithm 1 Computing the area of the union of N circles.

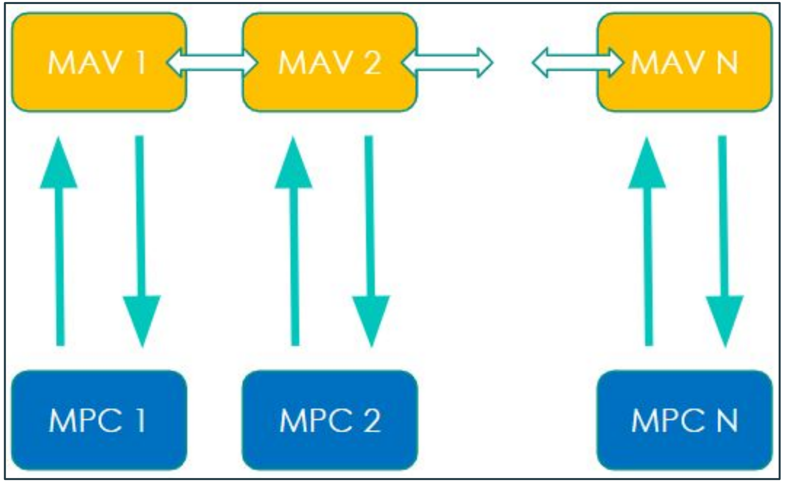
Require: $N, \{C_i\}_{i \in I}$ ▷ Set of all circles
$U := \{i \mid \exists j \neq i \text{ s.t. } C_i \subseteq D_j\}$ ▷ Covered circles
$I \leftarrow I \setminus U$ ▷ Remove covered circles
for all $i \in I$ **do**
    for all $j \in I$ **do**
        if $C_i \cap C_j \neq \emptyset$ **then**
            Add points $l \in \xi_i$
            Add edge $(i, j) \in E$
        end if
    end for
    Compute Θ_i and L_i
    $L_i \leftarrow L_i \setminus W$ ▷ Remove covered circular arcs
end for
Compute clusters γ_k
for all $k \in K$ **do**
    Compute circular arcs L^k and boundaries B^k_m
    for all $m \in M_k$ **do**
        Sort B^k_m clockwise
        Get area Z^k_m enclosed by each contour
    end for
    Compute $Z^k := 2 \max_m Z^k_m - \sum_{m=1}^{M_k} Z^k_m$
end for
$A \leftarrow \sum_{k=1}^{K} Z^k$ ▷ Total area of the union

Problem Formulation - Path Planning

- Compute solution from current position to ideal position (output of exploration phase)
- Standard LQR tracking cost:

$$(x_N - x_f)^T Q_f (x_N - x_f) + \sum_{k=1}^{N-1} (x_N - x_f)^T Q (x_k - x_f) + u_k^T R u_k$$

- Model Predictive Control: non-optimal but **good enough**
- Online distributed algorithm: each MAV computes its own ideal trajectory, future states shared between MAVs after each iteration
- Collision avoidance implemented via spherical constraints + danger zone cost



Results

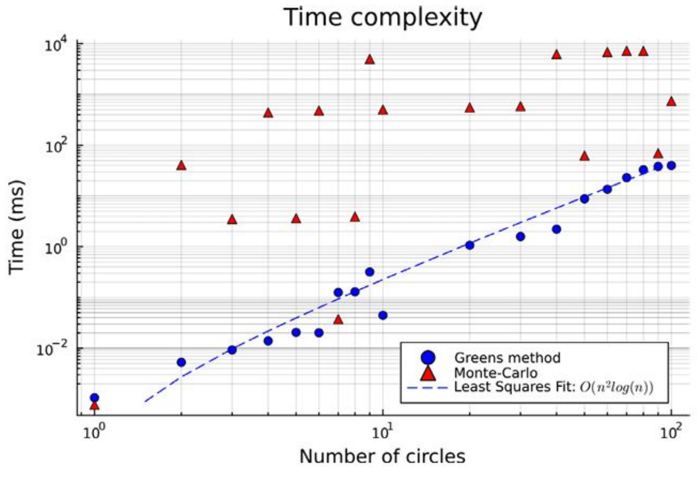


Fig. 3. Time complexity plot of Green's Method and Monte-Carlo simulations.

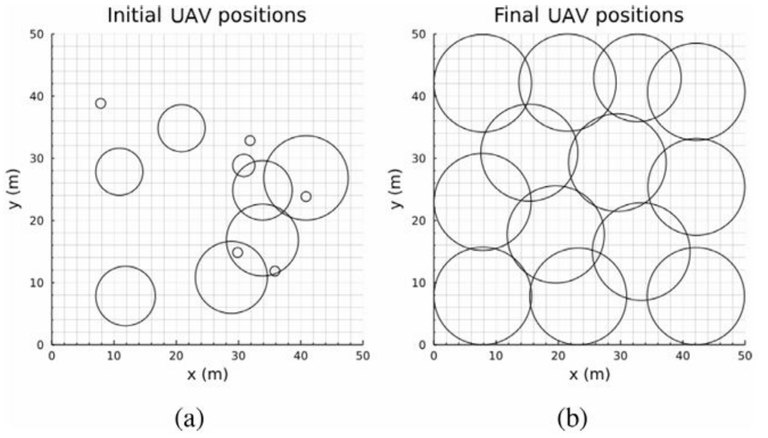


Fig. 6. (a) Initial UAV positions (b) Target UAV positions for $N = 13$. Note altitude depends on the radius.

Results

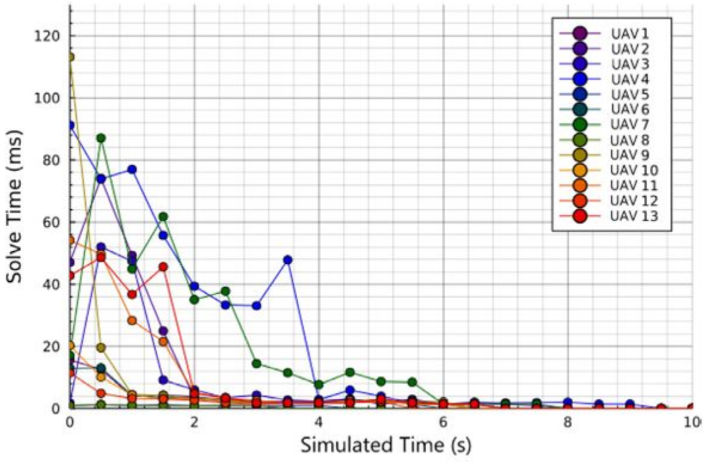


Fig. 8. Trajectory generation solve time for $N = 13$.

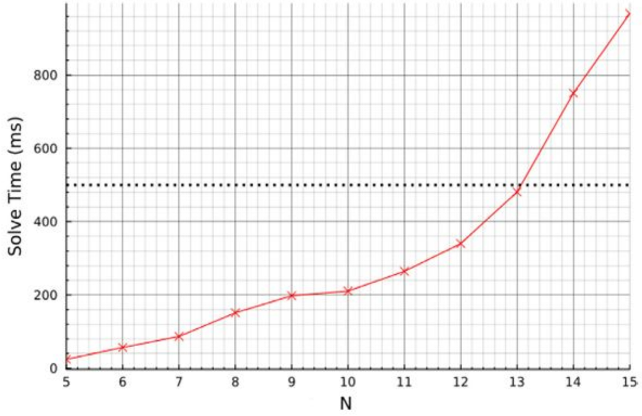


Fig. 9. Maximal area coverage solve time against the number of UAVs N. Dotted line at $dt = 500$ ms indicates the solve time limit imposed by the chosen NMPC time-step.

Conclusion

- We have proposed a **hierarchical control framewor**k for a multi-agent system using UAVs to quickly cover a predefined area
- Our approach would overcome the slow convergence usually encountered in fully distributed systems, by fusing a **centralized area coverage** problem with **distributed trajectory planning**
- The area coverage phase is tackled by developing a **novel algorithm**, that computes the union area of multiple intersecting circles using Green's theorem
- Robustness during real-time control is ensured by implementing an **NMPC** scheme that implicitly handles plant-model mismatch and allows the UAVs to adapt to external disturbances
- We demonstrate that our area coverage computation **outperforms Monte-Carlo simulations** for medium-scale systems and show that our control system can **operate in real-time** for up to 13 UAVs

Future Work

- Further research could attempt to develop a map that stores the areas that have already been visited by the UAVs. This will provide information on which further areas require to be mapped and hence **close the loop** on the exploration phase by mapping a domain to completion
- One could also explore the possibility of **determining a closed-form expression** for the union area of multiple circles and checking if the function is differentiable. This would allow the potential use of derivative-based solvers and may result in significant computational time reduction
- Additionally, further work is required to theoretically analyze the performance of our algorithms and **formally prove the time complexity** that was observed experimentally
- More realistic models should also be considered to better represent real-life scenarios. Since our framework has been tested only in simulation, one could go further by implementing the algorithms on real UAVs and run **hardware-in-the-loop** experiments in order to validate the design
- Finally, the current implementation is designed with the upper-level maximum area coverage solution being **computed at each time-step**. Further work can explore the possibility of relaxing this constraint to allow acceptable performance in real-time for a larger system