MULTI-ROBOT WAYPOINT INSPECTION PLAN MIXED INTEGER LINEAR PROGRAMMING PROJECT

Juan Carlos Cruz - ira406

ME 6033 Linear and Mixed Integer Optimization

April 2025

INTRODUCTION

- Based on previous work done in an outdoor concrete inspection multirobot framework
- Precast concrete elements require efficient inspection methods after being transported to a site
- Multi-robot approach:
  - Aerial robots locate targets
  - Ground robots perform detailed inspections
- For this project wanted to see if we could extend this idea to planning of multiple robots across an inspection site

PROBLEM DESCRIPTION

- Two robot types: aerial and ground mobile robots
- Inspection targets as waypoints in a 2D plane
- Depot location for each robot type
- Sequential operation:
  - Aerial robots verify waypoint location first
  - Ground robots perform detailed inspection second
- Robot constraints:
  - Fixed speeds (meters/minute)
  - Limited operation time (battery life time after leaving depot)
  - Required inspection time at waypoints
- Objective: Maximize waypoint visits in a single inspection loop.
- Becomes a MILP problem. Linear objective function and linear constraints but use binary and continuous variables.

LITERATURE REVIEW

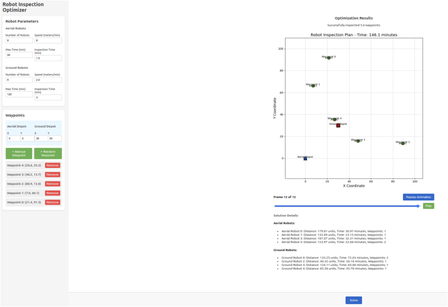
- Prior works demonstrate multi-robot planning applications
- Problem resembles multiple traveling salesman problem (mTSP)
- Traditional MILP for mTSP:
  - Routing variables
  - Subtour elimination constraints
- Initially tried this approach but solving time was too slow and larger problems became infeasible (likely implementation error) so used a simplification
- Route approximation: roundtrip distances from depot to waypoints estimate travel times

MODEL

$$
\begin{aligned}
&\text{Maximize} && \sum_{i\in N}\sum_{l\in L} w_i^{g,l} && && (1)\\
&\text{subject to:} && && && \\
&\text{Assignment:} && \sum_{k\in K} w_i^{a,k} \le 1 && \forall i\in N && (2)\\
& && \sum_{l\in L} w_i^{g,l} \le 1 && \forall i\in N && (3)\\
&\text{Robot Usage:} && \sum_{i\in N} w_i^{a,k} \ge \mathsf{use}_k^a && \forall k\in K && (4)\\
& && \sum_{i\in N} w_i^{a,k} \le n\cdot \mathsf{use}_k^a && \forall k\in K && (5)\\
& && \sum_{i\in N} w_i^{g,l} \ge \mathsf{use}_l^g && \forall l\in L && (6)\\
& && \sum_{i\in N} w_i^{g,l} \le n\cdot \mathsf{use}_l^g && \forall l\in L && (7)\\
&\text{Precedence:} && w_i^{g,l} \le \sum_{k\in K} w_i^{a,k} && \forall i\in N, \forall l\in L && (8)\\
& && g_i^l \ge a_i^k - M_G\cdot(1-z_i^{k,l}) - M_G\cdot(2-\mathsf{use}_k^a-\mathsf{use}_l^g) && \forall i\in N, k\in K, l\in L && (9)\\
&\text{Linking } z_i^{k,l}\text{:} && z_i^{k,l} \le w_i^{g,l} && \forall i\in N, k\in K, l\in L && (10)\\
& && z_i^{k,l} \le w_i^{a,k} && \forall i\in N, k\in K, l\in L && (11)\\
& && z_i^{k,l} \le \mathsf{use}_k^a && \forall i\in N, k\in K, l\in L && (12)\\
& && z_i^{k,l} \le \mathsf{use}_l^g && \forall i\in N, k\in K, l\in L && (13)\\
& && z_i^{k,l} \ge w_i^{g,l} + w_i^{a,k} + \mathsf{use}_k^a + \mathsf{use}_l^g - 3 && \forall i\in N, k\in K, l\in L && (14)\\
&\text{Time:} && a_i^k \ge t_A^{\mathsf{insp}}\cdot w_i^{a,k} && \forall i\in N, k\in K && (15)\\
& && g_i^l \ge t_G^{\mathsf{insp}}\cdot w_i^{g,l} && \forall i\in N, l\in L && (16)\\
& && a_i^k + t_{i,d_A}^A\cdot w_i^{a,k} \le T_A^{\max} + M_A\cdot(1-w_i^{a,k}) && \forall i\in N, k\in K && (17)\\
& && g_i^l + t_{i,d_G}^G\cdot w_i^{g,l} \le T_G^{\max} + M_G\cdot(1-w_i^{g,l}) && \forall i\in N, l\in L && (18)\\
&\text{Route Length:} && \sum_{i\in N} w_i^{a,k}\cdot(2\cdot t_{d_A,i}^A) + \sum_{i\in N} w_i^{a,k}\cdot t_A^{\mathsf{insp}} \le T_A^{\max} + M_A\cdot(1-\mathsf{use}_k^a) && \forall k\in K && (19)\\
& && \sum_{i\in N} w_i^{g,l}\cdot(2\cdot t_{d_G,i}^G) + \sum_{i\in N} w_i^{g,l}\cdot t_G^{\mathsf{insp}} \le T_G^{\max} + M_G\cdot(1-\mathsf{use}_l^g) && \forall l\in L && (20)\\
&\text{Domain:} && w_i^{a,k}, w_i^{g,l}, \mathsf{use}_k^a, \mathsf{use}_l^g, z_i^{k,l} \in \{0,1\} && \forall i\in N, k\in K, l\in L && (21)\\
& && a_i^k, g_i^l \ge 0 && \forall i\in N, k\in K, l\in L && (22)
\end{aligned}
$$

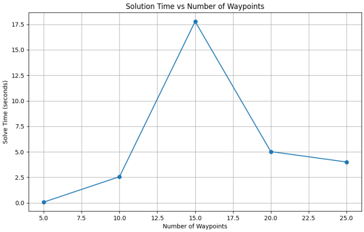
SOLUTION METHOD

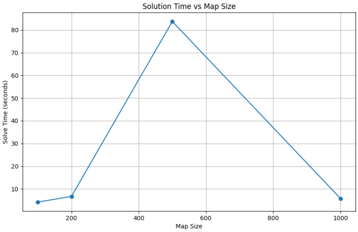
- Implemented in Python using PuLP library
- CBC solver from PuLP used for optimization
- Interactive browser-based GUI developed:
  - Parameter input for robot specifications
  - Waypoint location setting
  - Real-time solution visualization
- Code available at GitHub repository



NUMERICAL RESULTS - COMPUTATIONAL PERFORMANCE

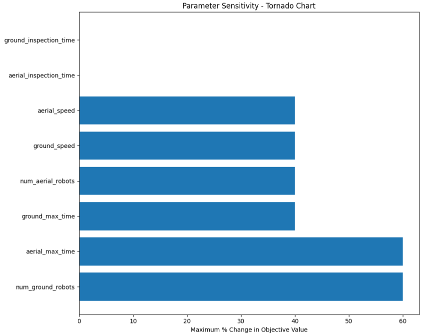
- Testing environment:
  - Intel i7, Python 3.12 Docker container
- Tested waypoint scaling (5 to 25 waypoints) and map size scaling (100 to 1000 m)
- Non-monotonic scaling behavior:
  - Solution time peaks at 15 waypoints then decreases
  - Computation time peaks at 500m map size



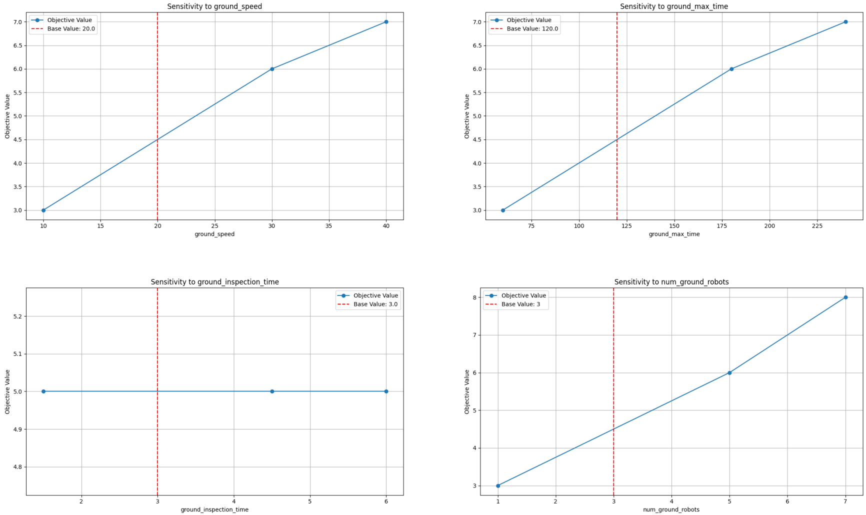


NUMERICAL RESULTS - SENSITIVITY ANALYSIS

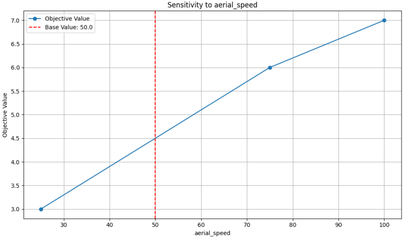
- Using fixed waypoints and map size we solve multiple times varying 8 params: robot speeds, operation times, inspection times, and fleet sizes.
- Most influential parameters:
  - Number of ground robots
  - Aerial robot maximum operation time
- Minimal impact: Inspection times

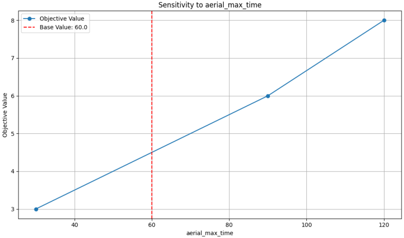


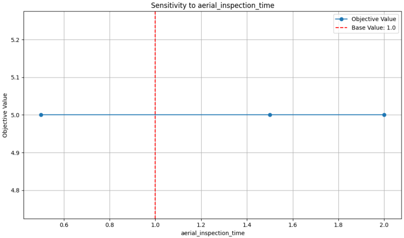
NUMERICAL RESULTS - GROUND ROBOT SENSITIVITY ANALYSIS

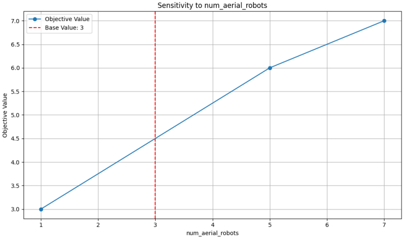


NUMERICAL RESULTS - AERIAL ROBOT SENSITIVITY ANALYSIS

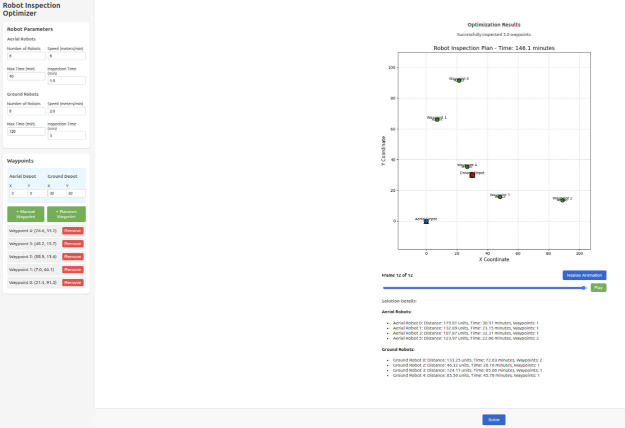








DEMO VIDEO



Click to watch the demo video