MULTI-ROBOT WAYPOINT INSPECTION PLAN MIXED INTEGER LINEAR PROGRAMMING PROJECT

Juan Carlos Cruz - ira406

ME 6033 Linear and Mixed Integer Optimization

April 2025

INTRODUCTION

- Motivated by outdoor construction site inspection challenges
- Precast concrete elements require efficient inspection methods
- Multi-robot approach:
  - Aerial robots locate targets
  - Ground robots perform detailed inspections
- Need for flexible inspection plans in dynamic environments
- MILP approach maximizes inspection coverage

PROBLEM DESCRIPTION

- Two robot types: aerial and ground mobile robots
- Inspection targets as waypoints in a 2D plane
- Sequential operation:
  - Aerial robots verify waypoint location first
  - Ground robots perform detailed inspection second
- Robot constraints:
  - Fixed speeds (meters/minute)
  - Limited operation time (battery endurance)
  - Required inspection time at waypoints
- Objective: Maximize waypoint visits in a single inspection loop

LITERATURE REVIEW

- Problem resembles multiple traveling salesman problem (mTSP)
- Prior works demonstrate multi-robot planning applications
- Traditional MILP for mTSP:
  - Routing variables
  - Subtour elimination constraints
- Simplification: Route length approximation
- Roundtrip distances from depot to waypoints estimate travel times

MODEL

Sets & Parameters:

- Waypoints N, indexed by i
- Aerial robots K, ground robots L
- Speeds: s_A, s_G
- Max operation times: T_A^{max}, T_G^{max}
- Inspection times: t_A^{insp}, t_G^{insp}

Decision Variables:

- $w_i^{a,k}$: Aerial robot k visits waypoint i
- $w_i^{g,l}$: Ground robot l visits waypoint i
- a_i^k: Aerial completion time
- g_i^l: Ground completion time
- use_k^a, use_l^g: Robot usage

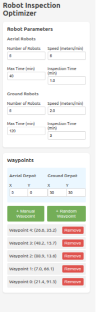
Objective: Maximize $\sum_{i\in N} \sum_{l\in L} w_i^{g,l}$

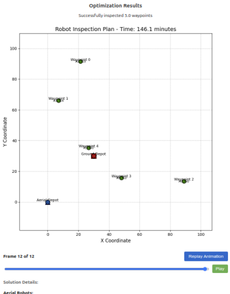
Key Constraints:

- Assignment: One robot per waypoint
- Precedence: Aerial robots visit before ground robots

SOLUTION METHOD

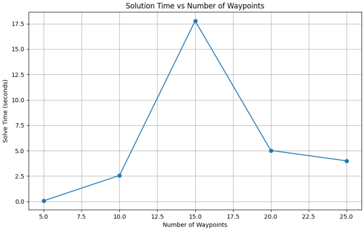
- Implemented in Python using PuLP library
- CBC solver from PuLP used for optimization
- Interactive browser-based GUI developed:
  - Parameter input for robot specifications
  - Waypoint location setting
  - Real-time solution visualization
- Code available at GitHub repository

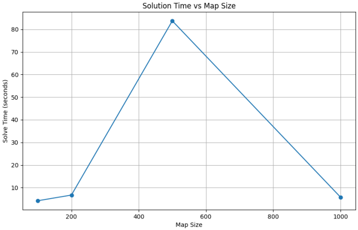




NUMERICAL RESULTS - COMPUTATIONAL PERFORMANCE

- Testing environment:
  - Intel i7, Python 3.12 Docker container
  - 500s solver time limit
- Non-monotonic scaling behavior:
  - Solution time peaks at 15 waypoints then decreases
  - Computation time peaks at 500m map size

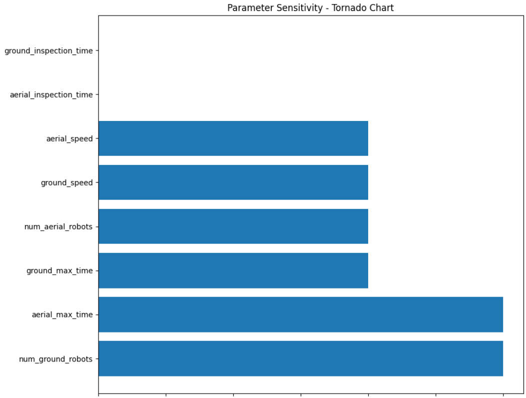




Left: Solution time vs waypoint count. Right: Solution time vs map size

NUMERICAL RESULTS - SENSITIVITY ANALYSIS

- Most influential parameters:
  - Number of ground robots
  - Aerial robot maximum operation time
- Significant impact: Robot speeds
- Minimal impact: Inspection times



DEMO VIDEO