THE ANDREW & ERNA VITERBI FACULTY OF ELECTRICAL ENGINEERING

CRML Control Robotics and Machine Learning Laboratory

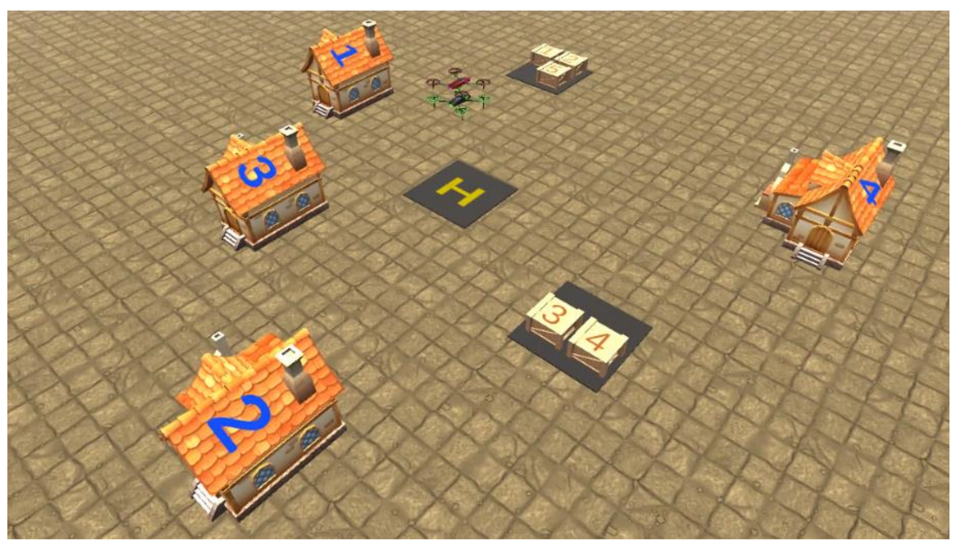
TECHNION Israel Institute of Technology

Designing a multi-agent physical simulation

Saar Stern and Yamit Pisman, Supervised by Ayal Taitler

Introduction

- Drone-based delivery revolutionizes logistics, posing new challenges in goods transportation.
- Optimizing drone routes for efficiency and energy use adds significant complexity.
- Coordinating multiple drones amplifies this challenge, demanding advanced planning algorithms.



Our simulation Environment

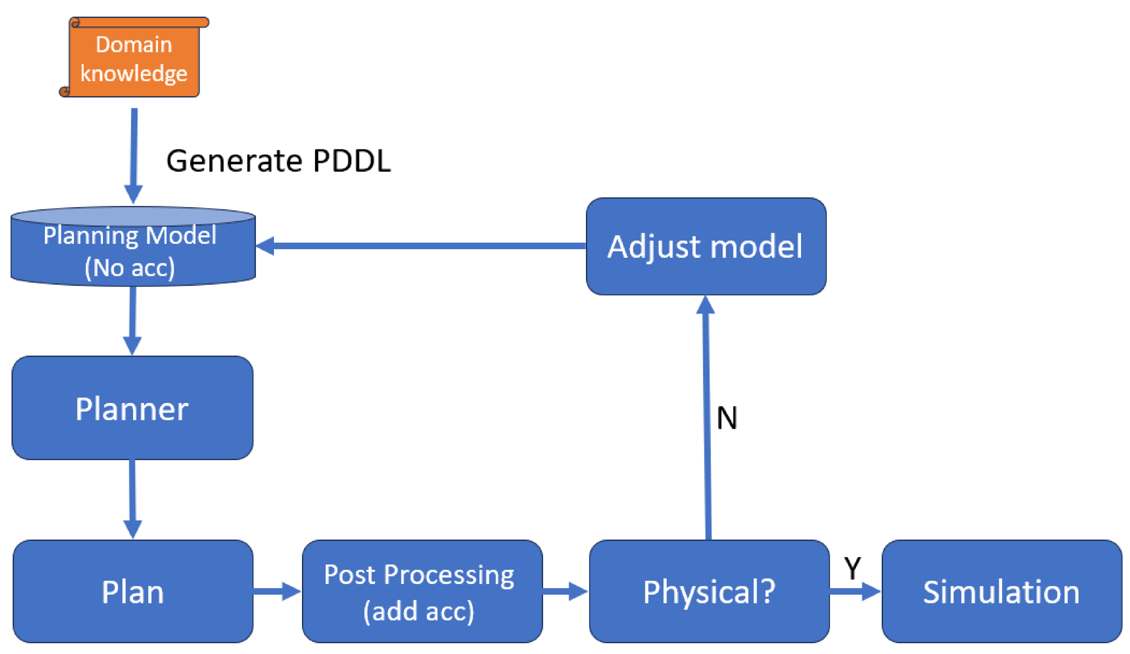
Goal

- Design drone's pickup and delivery physical simulation, which contains time and space synchronization.

Challenges

- Temporal – spatial planning
- Time windows for actions (e.g., House 3 accept delivers only at 14:00-16:00)
- Physical constraints
  - Max velocity, max acceleration, battery
- Minimal overall time
- Minimal battery use
- Robust to number of drones, houses, warehouses, packages, slots per drone.

General Scheme



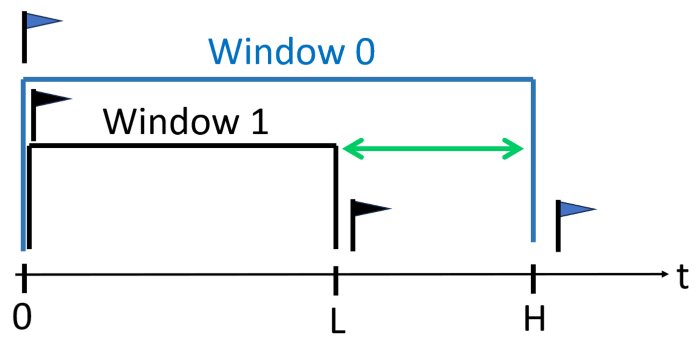
General scheme for physical simulation

- Starting with the domain knowledge, generating PDDL (Planning language) files without constrains on acceleration.
- Solving the problem using a planner, then adding acceleration into consideration.
- Checking whether the result plan is physical (e.g., obeys max velocity). If so, continue to simulation. If not, adjust model and start again.

State and Action Space

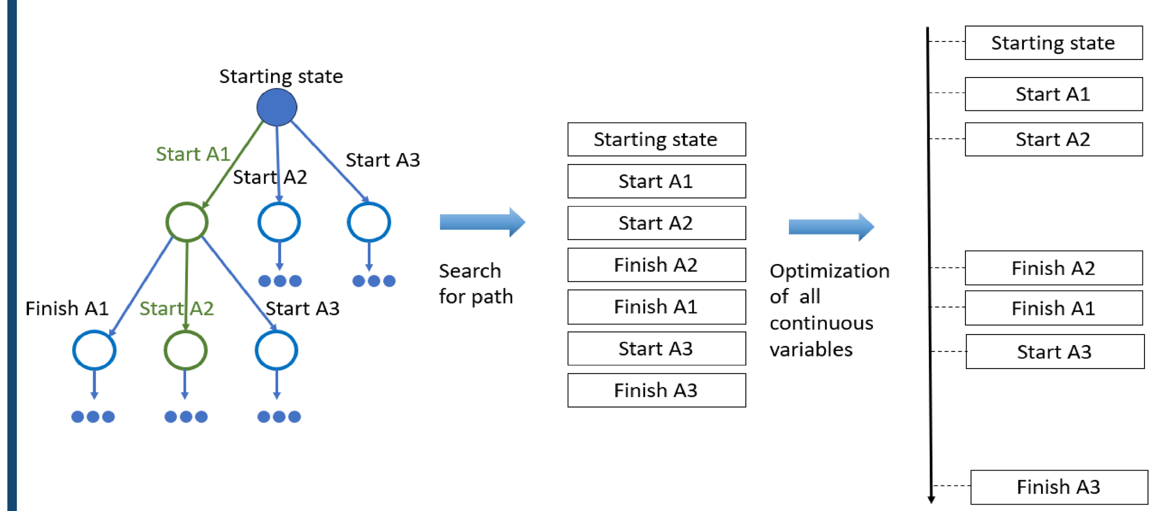
- PDDL files are generated according to arguments (e.g., no. of drones, no. of packages)
- event = start\end of action
- Inducing a large State\Action space consists of:
  - Time
  - Control variables: velocity for each drone, can change only in event
  - Position and Battery for each drone
  - Global predicates (e.g., 'has-package(d, p, s)' Indicates a package p is with drone d at slot s)
  - Actions (e.g., 'pickup(d,p,w,s)' is an action in which drone d is flying to pickup package p from warehouse w into slot s of the drone.)

Time Windows



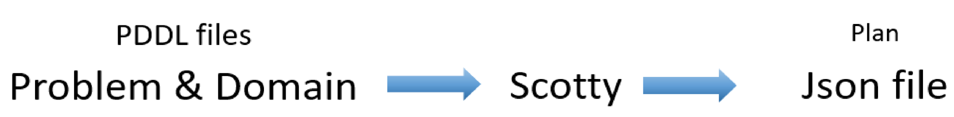
- How can we constrain an action A to a time window $[L, H]$?
- Solution: create 2 actions starting at $t = 0$ and ending at $t = L$ and $t = H$ respectively, each raising a flag while active.
- Add the precondition $flag_0 = 1$ & $flag_1 = 0$ to A.

ScottyActivity planner



ScottyActivity planner scheme for solving temporal problems

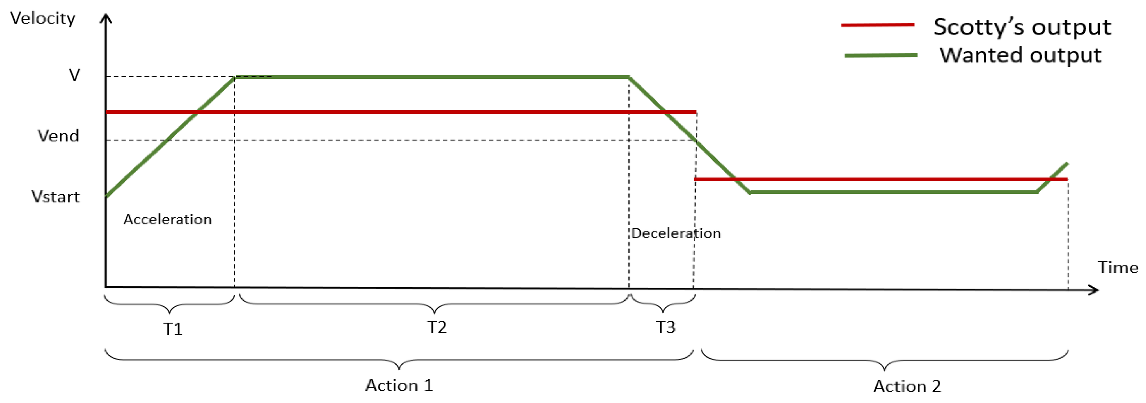
- Scotty is a planner which was developed in MIT.
- The planner searches (heuristically, with A^* or EHC) in the discrete events tree.
- Scotty solves SOCP optimization problem after every step in the search.
- The result plan contains list of events, each event has its time, updated control variables, and the current value of the continuous variables (e.g., position and battery)



Input and output - ScottyActivity

Post processing

- Scotty's plan is not physical – between two different actions, the velocity changes instantly as shown below in red.
- We want to consider (constant) acceleration, as shown below in green.



- Define for each section $1 \leq i \leq N$
  - $X_1(\mathbf{V}) = \int_0^{T_1} V_{start} + a \cdot sign(\mathbf{V} - V_{start}) \cdot t \ dt$
  - $X_2(\boldsymbol{V}) = T_2 \cdot \boldsymbol{V}$
  - $X_3(\boldsymbol{V}) = \int_0^{T_3} \boldsymbol{V} + a \cdot sign(V_{end} - V) \cdot t \ dt$
- Solve $Total\ distance = (X_1 + X_2 + X_3)(\boldsymbol{V})$
- Iterate process 2 times (different initial values)
  - $\forall i \ (V_{start}^i, V_{end}^i) = (0,0)$ - getting $\widetilde{\boldsymbol{V}}^i$
  - $\forall i \ (V_{start}^i, V_{end}^i) = 0.8 \cdot (\widetilde{\boldsymbol{V}}^i, \widetilde{\boldsymbol{V}}^{i+1})$ – getting $\boldsymbol{V}^i$

Simulation in Unity

- We simulated the physical plan with Unity – an engine for developing video games.

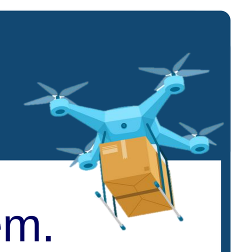
unity

Results

Our scenario:

- 2 drones, Max velocity 15 $\left[\frac{m}{s}\right]$, acceleration $4.75 \left[\frac{m}{sec^2}\right]$ (real specs).
- 4 houses, 2 warehouses, 5 packages.
- House 3 gets packages {3,5} only in a timed window. (3 variants)
- **Feedback:** The 1st plan from scotty after post processing wasn't physical feasible. Drones had intervals with $V = 16.3\backslash 17.3$, max is 15.
- **Solution:** Change max velocity constraint in PDDL from 15 to 13 (the physical constraint).
- This way the velocity won't go above the physical constraint after post processing.
- **Verdict:** plan satisfies all requirements

Summery



- Solving spatial-temporal planning problem.
- Generating PDDL files according to problem (with time intervals).
- Study and use ScottyActivity – advanced planner.
- Physical Plan by solving physical equations.
- Feedback mechanic.
- Simulation with Unity.

January 2024