Model-based Programming for Cooperating Vehicles

16.412 Group Project

Seung H. Chung
Robert T. Effinger
Thomas Léauté
Steven D. Lovell

May 12, 2004

Cooperative Model-based Program Architecture

Control Program
- Team mission objectives & coordination constraints
- Team strategies & tactics
- Options & contingencies

System Model
- Terrain map
- Threats and assets
- Vehicle commands and resource models.

Control Sequencer
Selects Feasible Mission Plan from Options

Mission State (Current & Predicted)

Desired Sequence of Mission States (Mission Plan)

Model-based Executive

Estimation and Prediction

Activity and Path Planning

Sensor Observations
Commands
Cooperative Model-based Program Architecture

Control Sequencer
- **Kirk**: searches Temporal Plan Network (TPN) to find a feasible mission plan

Model-based Executive
- **Fast Roadmap Path Planner**: quickly estimate flight distance, path, and time
- **Generative Activity Planner**: generates activity plan from mission plan
- **Kinodynamic Path Planner**: generates a path plan for each motion activity

System Model
- Terrain map
- Threats and assets
- Vehicle commands and resource models.

Control Program
- Team mission objectives & coordination constraints
- Team strategies & tactics
- Options & contingencies

Forest Firefighting with Cooperative UAV’s
Cooperative Forest Fire Fighting Scenario

Legend:
- Seeker UAV
- Water UAV
- No-Fly Zone
- Fire
- Water
- UAV Base
- Refueling Station

Plan Goal: Extinguish All Fires
Vehicles: One Seeker UAV (to image the fires)
One Water UAV (to extinguish fires)
Resources: Fuel & Water

Cooperative Model-based Program Architecture

Control Program
- Team mission objectives & coordination constraints
- Team strategies & tactics
- Options & contingencies

System Model
- Terrain map
- Threats and assets
- Vehicle commands and resource models.

Control Sequencer
- Kirk: searches Temporal Plan Network (TPN) to find a feasible mission plan
- Mission State (Current & Predicted)
- Desired Sequence of Mission States (Mission Plan)

Model-based Executive
- Fast Roadmap Path Planner: quickly estimate flight distance, path, and time
- Generative Activity Planner: generates activity plan from mission plan
- Kinodynamic Path Planner: generates a path plan for each motion activity

Sensor Observations

Commands
Robert Effinger’s Contributions

1. Created Interface between the high-level planner (Kirk) and the Path Planner (dStarLite)
   (*jointly with Dan Lovell)

2. Created framework in Kirk to solve TPN as a dynamic CSP

3. Implemented Dynamic Backtracking within Kirk

---

Interface Between Kirk and dStarLite

Operator inputs detailed mission scenario:

1. Choose which order to image Fire 1 and Fire2
2. Choose which order to drop water on Fire 1 and Fire2
3. UAVs Return to Base together
4. Update lower timebound using path-planner
Solving a TPN as a Dynamic CSP

2 Main Functions:
1.) getNextDecisionStartNodes()
2.) deactivateNestedDecisionNodes()

Dynamic Backtracking Implementation

- Backtracks to the source of the problem
- Keeps variable assignments that aren’t part of the problem.
Cooperative Model-based Program Architecture

Control Program
- Team mission objectives & coordination constraints
- Team strategies & tactics
- Options & contingencies

System Model
- Terrain map
- Threats and assets
- Vehicle commands and resource models.

Control Sequencer
Kirk: searches Temporal Plan Network (TPN) to find a feasible mission plan

Model-based Executive
- Fast Roadmap Path Planner: quickly estimate flight distance, path, and time
- Generative Activity Planner: generates activity plan from mission plan
- Kinodynamic Path Planner: generates a path plan for each motion activity

Sensor Observations

Commands

Mission Plan to Activity Plan

Move (Base, Lake) [130,160] Get Water (Lake) [30,50] Move (Lake, Base) [170,240] Drop Water (Fire0) [30,50]

Drop Water (Fire0) [30,50]

Take Image (Fire0) [250,1000]

Image (Fire0) [30,50]

Move (Base, Fire0) [250,500]

Take Image (Fire1) [30,50]

Move (Fire0, Fire1) [30,50]

Image (Fire1) [30,50]

Get Water (Lake) [30,50]
Mission Plan to Activity Plan

Legend:
- Seeker UAV
- Water UAV
- No-Fly Zone
- Fire
- Lake
- UAV Base
- Refueling Station

Generative Activity Planner

Uses temporal generative planner (augmented LPG) to generate an activity plan from a mission plan.

The planner must be able to achieve complex sequence of goals (i.e. a mission plan represented as an STN).
LPG as a Generative Activity Planner

- **Dynamic Problem Definition**
  - Update current state as necessary
    - Unavailable vehicles
    - Reachable locations
    - Distance between locations
  - Enables closed loop replanning

- **Dynamic Domain Definition**
  - Add mission plan as domain operators
  - Enables a search for an optimal solution

- **Plan with Flexible Time Bound**
  - Add upper bound to each action and check the temporal consistency

Mapping Mission Plan into Domain Operators

```prolog
(:durative-action take-image-seeker-uav0-fire0
  :parameters ()
  :duration (= ?duration 0)
  :condition (and (at start (not (act0-achieved)))
                (at start (have-image seeker-uav0 fire0)))
  :effect (at end (act0-achieved)))

(:durative-action drop-water-water-uav0-fire0
  :parameters ()
  :duration (= ?duration 0)
  :condition (and (at start (not (act1-achieved)))
                 (at start (act0-achieved))
                 (at start (water-uav0 fire0))
                 (at start (dropped-water water-uav0 fire0)))
  :effect (at end (act1-achieved)))
```
Seung’s Contributions

1. Wrote the scenario in PDDL2.1 (PDDL+ Level-3 subset)
   • Lack duration inequalities and continuous effects requires many workarounds.

2. Generated a plan for the scenario using LPG
   • Reaching the limits of LPG’s capability – required problem description tweaking

3. Develop codes and methods necessary for the proof of concept
   • Gained some insights into a new approach to generative planning

4. Develop/Compile the STN interface necessary for the architecture

Cooperative Model-based Program Architecture

Control Program
- Team mission objectives & coordination constraints
- Team strategies & tactics
- Options & contingencies

System Model
- Terrain map
- Threats and assets
- Vehicle commands and resource models.

Control Sequencer
- Kirk: searches Temporal Plan Network (TPN) to find a feasible mission plan

Model-based Executive
- Fast Roadmap Path Planner: quickly estimate flight distance, path, and time
- Generative Activity Planner: generates activity plan from mission plan
- Kinodynamic Path Planner: generates a path plan for each motion activity

Sensor Observations → Commands
Thomas Leaute’s Contributions

1. Wrote a comprehensive software interface with the Cloud Cap simulator, exchanging TCP messages to read telemetry and send commands

2. Created a Tactic-driven Cooperative Path Planner that uses Mixed Integer Linear Programming to both plan paths and execute a temporally flexible activity plan

3.Drafted a receding horizon planning framework that asks the path planner to re-plan any time necessary

Path Planning within a Receding Horizon Using Mixed Integer Linear Programming

- Only plan within a limited time window
- Use fast roadmap path planner to estimate the remaining cost to go

**Result:**
- Quick kinodynamic path planner that minimizes the total path cost
- Ability to incrementally replan as the knowledge of the environment is updated.
Scheduling within a Receding Horizon Using Mixed Integer Linear Programming

Nodes → time variables $t_i$

Duration constraints → constraints $T_{\text{min}} < t_j - t_i < T_{\text{max}}$

“Go to G” activities → post-conditions $X(t_j) = X_G$

“Stay at P” activities → for all $t_i < t < t_j$: $|X(t) - X_P| < e$

Cooperative Model-based Program Architecture

Control Program
- Team mission objectives & coordination constraints
- Team strategies & tactics
- Options & contingencies

System Model
- Terrain map
- Threats and assets
- Vehicle commands and resource models.

Control Sequencer
Kirk: searches Temporal Plan Network (TPN) to find a feasible mission plan

Mission State (Current & Predicted) → Desired Sequence of Mission States (Mission Plan)

Model-based Executive

Fast Roadmap Path Planner: quickly estimate flight distance, path, and time

Generative Activity Planner: generates activity plan from mission plan

Kinodynamic Path Planner: generates a path plan for each motion activity

Sensor Observations → Commands
Dan Lovell’s Contribution

- Cost map generating algorithm that uses D*-lite
  - Generate a cost map for a given region
  - Using D*-lite, can respond to dynamic environment using local repair of the path

Generating Cost Maps with D*-lite

To create Cost To Go Map, execute many D*-lite searches to locations in map while saving old state

- Previous consistent nodes remain consistent.
  - Searches to consistent locations terminate immediately.

- Previous priority queue needs to be reordered for each new search location.
  - Reordering priority queue ensures the next search creates consistent nodes.

- How do you prevent too many queue reorderings?
  - Order of searches might matter
  - Change heuristic function or tie breaking so search paths wander more, covering more area near the destination.
Generating Cost Maps with D*-lite

• The order in which you search points in the cost map matters
  • Large changes in euclidean distance between start locations could require much of the priority queue to be reordered
  • Small Changes could cause you to minimally reorder queue but reorder many times since each search won’t cover as much new terrain

Cooperative Forest Fire Fighting Scenario

Legend:
- Seeker UAV
- Water UAV
- No-Fly Zone
- Fire
- Water
- UAV Base
- Refueling Station

Step 1: Seeker UAV goes to confirm Fire 1
Step 2: Seeker UAV goes to confirm Fire 2
Step 3: Water UAV commanded to go and extinguish Fire 1
Cooperative Forest Fire Fighting Scenario

Legend:
- Seeker UAV
- Water UAV
- No-Fly Zone
- Fire
- Water
- UAV Base
- Refueling Station

Step 4: Generative Planner discovers WaterUAV needs to get fuel and water before going to Fire1
Step 5: WaterUAV drops water on Fire1
Step 6: WaterUAV commanded to go and extinguish Fire2

Step 7: Generative Planner discovers WaterUAV needs to get water before going to Fire2
Step 8: WaterUAV drops water on Fire2
Step 9: SeekerUAV takes pictures of the fire damage
Step 10: UAVs return to Base (and get fuel if needed)
Conclusions

- **Model-based Programming**: Encodes mission strategies or tactics at a simple, intuitive level in terms of intended state evolutions.

- **Dynamic Backtracking**: Quickly generates an optimal mission plan using best-first order or anytime depth first order.

- **Activity Planning on Complex Sequence of Goals**: Refines complex sequence of goals (mission plan) into an actionable activity plan using existing temporal generative planner.

- **Strategy-driven Cooperative Path Planning**: Incorporates a kinodynamic cooperative path planner with the novel ability to design control trajectories specified in temporally flexible plans.

- **Receding Horizon Continuous Planners**: Quickly adapts the optimal plan as the environment and problem are perturbed using road map path planning.

Mission Description in RMPL

Operator inputs detailed mission scenario:

1. Create an overall strategy describing vehicle actions and intended movements.

2. Coordination and timing constraints.

3. Non-Deterministic Choices

4. Concurrent and Sequential Actions
Cooperative Model-based Program Architecture

**Control Program**
- Team mission objectives & coordination constraints
- Team strategies & tactics
- Options & contingencies

**System Model**
- Terrain map
- Threats and assets
- Vehicle commands and resource models.

---

**Control Sequencer**

*Kirk:* searches Temporal Plan Network (TPN) to find a feasible mission plan

**Model-based Executive**

**Generative Activity Planner:** generates activity plan from mission plan

**Kinodynamic Path Planner:** generates a path plan for each motion activity

**Mission State**
(Current & Predicted)  
Desired Sequence of Mission States (Mission Plan)

**Sensor Observations**

**Commands**

Cooperative UAV's (Simulated using Cloud Cap Autopilot)

---

**Cooperative Model-based Program Architecture**

Given high-level mission goals and strategies, selects an optimal sequence of activities from a temporal network with flexible time bounds. (Robert)

Searches a grid map for the shortest paths to specified goals (using D* Lite algorithm)
- Provides the control sequencer and activity planner with lower time bounds on flight duration
- Allows the kinodynamic path planner to plan beyond its receding horizon (Dan)

Uses temporal generative planner (augmented LPG) to generate an activity plan from mission plan. (Seung)

Uses Mixed Integer Linear Programming to generate path plans and schedule activities. (Thomas)