UAV Cooperative Control with Stochastic Risk Models

A Geramifard, J. Redding, N. Roy, and J. P. How

Aerospace Controls Laboratory
Laboratory for Information and Decision Systems
Department of Aeronautics and Astronautics
Massachusetts Institute of Technology

June 26, 2011
Outline

1. Introduction
Motivation

▶ DoD missions: execute persistent ISR with heterogeneous UAVs and tasks
• Balance competing objectives
• Rapid response
• Handle uncertainty
• Human operator/automated planner integration

▶ Overall Goal: Develop algorithms that control multiple UAVs to coordinate/cooperate to meet requirements in an optimized and robust way
• Significant work in this area, including our algorithms
Challenges of Cooperative Planning

- Most cooperative control algorithms are **model** based – enable anticipation of likely events & prediction of resulting behavior

- But the models are often **approximated**
  - Planning with stochastic models time consuming ⇒ model simplification

- Typical problems
  - Modeling errors (e.g. incorrect rules, non-representative objective functions, unmodeled uncertainties)
  - Model parameter uncertainties (e.g. vehicle minimum turn radius, fuel burn rate, probability of motor failure . . .)

- Result is **sub-optimal** planner output ⇒ **mismatch** between actual and expected performance
  - Can lead to catastrophic performance degradation
Challenges of Cooperative Planning

- Most cooperative control algorithms are model based – enable anticipation of likely events & prediction of resulting behavior

- But the models are often approximated
  - Planning with stochastic models time consuming ⇒ model simplification

- Typical problems
  - Modeling errors (e.g. incorrect rules, non-representative objective functions, unmodeled uncertainties)
  - Model parameter uncertainties (e.g. vehicle minimum turn radius, fuel burn rate, probability of motor failure ...)

- Result is sub-optimal planner output ⇒ mismatch between actual and expected performance
  - Can lead to catastrophic performance degradation
Challenges of Cooperative Planning

- Most cooperative control algorithms are model based – enable anticipation of likely events & prediction of resulting behavior.

- But the models are often approximated:
  - Planning with stochastic models time consuming \(\Rightarrow\) model simplification

- Typical problems:
  - Modeling errors (e.g. incorrect rules, non-representative objective functions, unmodeled uncertainties)
  - Model parameter uncertainties (e.g. vehicle minimum turn radius, fuel burn rate, probability of motor failure ...)

- Result is sub-optimal planner output \(\Rightarrow\) mismatch between actual and expected performance
  - Can lead to catastrophic performance degradation
Inaccurate Model $\Rightarrow$ Sub-optimal Solution

**Problem**: Find path from top-left (●) to bottom-right (★), while avoiding no-fly-zones (●). Movement noise = 30%.
Inaccurate Model ⇒ Sub-optimal Solution

- **Middle planner** assumes no noise ⇒ path approaches no-fly-zones.
- **Right planner** includes 30% noise ⇒ path maintains distance from the no-fly-zones.
Addressing Sub-optimalities

▶ Modeling errors:
  • Reinforcement learning

▶ Model parameter uncertainty:
  • Adaptive control techniques
  • Maximum-likelihood estimation
  • Data-driven learning methods (e.g. regression)

▶ Need a **framework** to enable these in conjunction with planning
Reinforcement Learning Framework

\[ V^\pi(s) = \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^{t-1} r_t \mid s_0 = s \right], \]

where \( \pi \) is policy that agent follows

**Goal:** Increase the number of UAVs in the persistent surveillance mission is an important research goal \( \Rightarrow \) large state space

**Learning in large state spaces is challenging:**
- Slow
- Memory intensive
- Computationally demanding
Developed template architecture for multi-agent planning and learning – intelligent Cooperative Control Architecture (iCCA) [4]

- **Cooperative planner (parent)**: Generates safe but sub-optimal policies.
- **Learner (child)**: May suggest unsafe actions, but will find optimal policies.
Learner (Natural Actor-Critic) has a \textit{parametric} form for the policy. Planner (Consensus-Based Bundle Algorithm) \textit{initializes} child’s policy.

Given a \textit{deterministic} risk model, each suggested actions of the child is rolled out with the parent’s policy.

Risky actions are replaced with the parent’s policy.
Current paper extended capabilities of that previous work:

1. **Relaxed** requirement that learner have a parametric policy form. Probability of child suggesting an action based on learned value function for a state increases as it experiences that state more.

2. Risk model can be **stochastic**. Safety ensured by generating multiple Monte-Carlo simulations and replacing risky actions suggested by learner with planner’s policy.
Algorithm Details

- The learner suggests actions with the following probability akin to the $R_{max}$ algorithm [2]:

$$P = \min\{1, \frac{\text{count}(s, a)}{N}\}$$

- Higher values of $N$ suggests slower exploration rate.
- $\text{count}(s, a) =$ number of times the planner picked action $a$ at state $s$.

- Furthermore, the safety of the learner’s suggested action is estimated through $M$ Monte-Carlo simulations.
Safe Exploration

Algorithm 1: Cooperative Natural Actor-Critic (CNAC)

Input: $\pi_p$, $\xi$

Output: $a$

$$a \sim \pi_{AC}(s, a)$$

if not safe($s, a$) ... pure planning methods in the GridWorld example mentioned in Section III, and a multi-UAV mission planning scheme quite simply as they parameterize the policy explicitly.

Hence, if estimated risk exceeds threshold $\psi$, the planner’s model, $\pi_p$, replaces the learner from suggesting that action again, reducing the risk of the state-action pair by the risk analysis element. If they are deemed to be safe, the critic and actor parameters are updated. Algorithm 2 details the process. On every step, the learner updates its parameter depending on the choice of the action suggested by the planner ($\pi_p$). Given the knownness of the state-action pair, the learner updates its parameter depending on the choice of the action suggested by the planner ($\pi_p$).

The initial policy of actor-critic type learners is biased towards the planner’s policy, $\pi_p$, generating the rest of actions.

Algorithm 2: safe

Input: $s, a$

Output: $isSafe$

$\text{risk} \leftarrow 0$

for $i \leftarrow 1$ to $M$ do

$t \leftarrow 1$

$s_t \sim T^p(s, a)$

while not constrained($s_t$) and not isTerminal($s_t$) and $t < H$ do

$t \leftarrow t + 1$

$risk \leftarrow risk + \frac{1}{i}(\text{constrained}(s_t) - \text{risk})$

end while

isSafe $\leftarrow (risk < \psi)$

The learner provides the first action, $a$, in each trajectory, while the planner’s policy, $\pi^p$, generates the rest of actions.

The planner’s model, $T^p$, is used to role out each trajectory.

If estimated risk exceeds threshold $\psi$, learner’s suggested action is replaced with the planner’s policy.
Empirical Results: GridWorld Domain

- True Model: 30% noise for movement
- Planner’s Model: 0% noise for movement
- Reward:
  - reaching goal, = +1
  - entering no-fly-zone = −1
  - all other moves = −0.001
Empirical Results: GridWorld Domain

- Average performance of methods through $10^4$ interactions using 60 trials. Bars highlight 95% confidence intervals.
- CNAC: iCCA + NAC, CSarsa: iCCA + Sarsa

![Graph showing performance of different methods](image)
2 UAVs, 6 Targets

- UAVs (triangles) and Targets (circles)
- Time windows for target visit times in brackets, e.g. [2,3]
- Target visit rewards
- Probability of receiving reward shown in cloud
- Stochastic risk model: 5% noise for traveling an edge
- $\approx 10^8$ state-action pairs

- iCCA and Actor-Critic test cases were run for 60 episodes
- CBBA was run on the deterministic version of the stochastic problem for 10,000 episodes – plot averaged performance
2 UAVs, 6 Targets: Simulation Results

- NAC, Sarsa, CBBA, CNAC, and CSarsa algorithms at the end of the training session in the UAV mission planning scenario.
- Cooperative learners (CNAC, CSarsa) perform very well with respect to overall reward and risk levels when compared with the baseline CBBA planner and the non-cooperative learning algorithms.
NAC, Sarsa, CBBA, CNAC, and CSarsa algorithms at the end of the training session in the UAV mission planning scenario.

Cooperative learners (CNAC, CSarsa) perform very well with respect to overall reward and risk levels when compared with the baseline CBBA planner and the non-cooperative learning algorithms.
2 UAVs, 6 Targets: Simulation Results

- NAC, Sarsa, CBBA, CNAC, and CSarsa algorithms at the end of the training session in the UAV mission planning scenario.

- Cooperative learners (CNAC, CSarsa) perform **very well** with respect to overall reward and risk levels when compared with the baseline CBBA planner and the non-cooperative learning algorithms.
Contributions

► Extensions:
  • Support for stochastic risk models
  • Support for learning methods with no parametric form for the policy (e.g., Sarsa).

► Empirical Results: Provided simulation results showing the benefit of integrating learning and planning in a multi-agent mission planning domain with more than $10^8$ possibilities.


