Aerial Mobility in Unknown Environments

Brett T. Lopez
Postdoctoral Scholar
Aerial Robotic Mobility Group
December 4, 2019
Motivation: human baseline

https://www.youtube.com/watch?v=NxyV-kfio&t=4s
Human feedback loop

Prediction

Video Feed

Feedback Loop

RC Commands
Autonomous feedback loop
Overview

• Background on planning
• World representation & planning strategy
• Collision avoidance with instantaneous pointclouds
• Adding history to world representation
• Model predictive control & robustness
Background - planning

• Classical planning problem (A→B) focused on new algorithms and reduced computation time
• Unrealistic expectation of perception quality
  • Assumptions: Infinite horizon, no uncertainty, sensor data easily transformed into useful form
  • Reality: Finite horizon, state/measurement uncertainty
• Difficult to extend existing planner into receding horizon framework
• Issue: Which world representation and planning technique?
  • Need techniques to integrate real-world perception into planners of varying complexity/timescales
Planning in Unknown Environments

• **Problem:** Navigate through unknown environment as fast as possible

• **Challenge:** World only partially known due to limited perception

• **Two Tasks:**
  1. Transform sensor data into usable world model
  2. Use world model to find path

• **Constraint:** Emphasize fast perception and planning ⇒ minimize reaction time

• **Issue:** Planning & perception designed independently ⇒ slow reaction time
World representations

• Occupancy grid map

• Raw sensor measurements

• Other

Lopez ‘17
Liu ‘17
Florence ‘18
Nam ‘15
Hornung ‘13
Otte ‘09
Oleynikova ‘16
Path planning strategies

• Graph/tree search

• Optimization-based

• Motion primitives

• Hybrid
Hierarchical planning architecture

- **Issue**: Limited computation power ⇒ limit planning horizon or reduce model fidelity

- **Solution**: Long horizon, simple model *global planner* + short horizon, complex model *local planner* ⇒ **Hierarchical planning**

- **Insight**: Global planner guides local planner
Triple Integrator Planner

• **Goal:** Efficient perception/planning pipeline for aggressive obstacle avoidance

• **Approach**
  1. Use *instantaneous* perception data for collision avoidance
  2. Generate motion primitives *online* with *approximate but accurate* vehicle model
  3. Check for collisions *efficiently*

• **Result:** *5ms* computation time is \(\approx 10x\) faster than previous state-of-the-art

Lopez ‘17
World representation revisited

• **Goal:** Low overhead, avoid sensor fusion/costly perception processing

• **Approach:** Construct simple world representation from *instantaneous* point cloud
  • **k-d tree:** convenient data structure for nearest neighbor search
  • 2 transformations: depth image → pointcloud → k-d tree

• **Limitations:**
  • No history of previously seen obstacles ⇒ tradeoff computation time/knowledge
  • Constrained to travel in sensor FOV ⇒ unknown space is occupied space
Minimum-time motion primitives

- **Goal:** Generate collision avoidance maneuvers with minimal computation time

  ⇒ **Motion primitives**

- **Formulation:**
  - Triple integrator model to approximate vehicle dynamics, including attitude
  - Min-time decoupled state/input constrained optimal control problem, with jerk as control input

- **Key idea:** Plan in velocity space to reduce comp. complexity
  - **Closed-form** solution ⇒ 3-D primitives generated online in 4.7µs
  - Primitive generated from current state to desired speed and direction

- **Result:** Bang-(off)-bang solution in jerk ⇒ highly agile maneuvers
State- vs. control-based motion primitives

• **Control-based:** *Satisfy control constraints* but **violate state constraints**

• **State-based:** *Satisfy state constraints* but require solving TPBVP ⇒ time consuming

• **Key Insight:** Planning in velocity space allows TPBVP to be solved with minimal comp. time ⇒ state/control input constraints guaranteed satisfied
  
  • Sample over speed and direction

![Diagram showing control-based primitives violating FOV constraint and state-based primitives trivially satisfying FOV constraint](image)
Primitive sorting

• **Goal:** Construct cost function *independent* of world model

• **Idea:** Calculate primitive cost using *heading* information and primitive *length*

• **Cost of primitive MP<sub>i</sub>:**

\[ J_i = \phi_i + c_1 \theta_i + c_2 (d_{\text{perception}} - d_i)^2 \]

- **Stage Cost**
- **Terminal Cost**

• **Stage cost:** Prioritize “similar” primitives

• **Terminal cost:** Prioritize “long” primitives in direction of goal
Efficient collision checking

• **Issue:** Collision checking typically expensive
  - Existing methods finely sample path or further process sensor data ⇒ *slow*

• **Key idea:** Estimate **next possible time** $t^*$
  based on top speed and closest obstacle

• **Procedure:**
  1. Estimate $t^*$
  2. Evaluate primitive at $t^*$
  3. Repeat until collision or sensing horizon reached

• Check small # of points along primitive
  ⇒ *Low computation time*
Indoor Flight Experiments [Lopez ‘17]

4m/s Slalom
Outdoor tests: good day

Wall seen ~3.5m away
Outdoor tests: good day
Adding history to world model

• Advantages of instantaneous pointclouds:
  • Robust to state estimation uncertainty
  • No temporal fusion \(\Rightarrow\) little required computation

• **Issue:** No history of previously observed obstacles \(\Rightarrow\) **myopic**

• **Possible solutions:**
  • Primitive/trajectory history
  • Pointcloud history without fusion
  • **Sliding occupancy map**

Florence ‘18
Lopez ‘17
Ryll ‘19
More outdoor test: good day

• **World model**: Instantaneous pointcloud + local map
More outdoor test: good day

- **World model**: Instantaneous pointcloud + local map

Pole enters sensing range
Outdoor tests: bad day

Ad-hoc design of planning/control architecture ⇒ no performance guarantees
Model Predictive Control

• **MPC**: Repeatedly solve constrained, multivariable optimal control problem

\[ u^* = \arg \min_{u} J \]
subject to \[ \dot{x} = f(x) + b(x)u \]
\[ x \in X, \quad u \in U, \]
\[ x(t_0) = x(t_0), \quad x(t_f) \in X_f \]

• **Insight**: Works well with accurate model of dynamics ⇒ robustness issues
MPC with tracking controller

- MPC + tracking controller \( \Rightarrow \text{compensate} \) for model error & disturbance

- Standard architecture in many fields (e.g., robotics, aerospace, etc.)

- Issue: Ad-hoc feedback controller design \( \Rightarrow \text{no performance guarantees} \)
  - Tube MPC \( \Rightarrow \) performance guarantees
Tube MPC

• **MPC**: Open-loop execution of optimal control solution
  • Little robustness to unmodeled dynamics/disturbances

• **Tube MPC**: Generate open-loop reference that is tracked by ancillary controller
  • Controller bounds tracking error ⇒ **tube** around desired trajectory

• **Issue**: Constructing controller/tube non-trivial for nonlinear systems

• Existing methods overly conservative, expensive to compute, & not generalizable

• Very active area of research for nonlinear systems
Planning needs for subterranean challenge

• Less emphasis on speed, more emphasis on autonomy
  • Frontier exploration
  • Active search

• Perception-degraded environment
  • Planning needs resiliency to state estimation errors/failures

• Accurate SLAM
  • Return to previously visited areas to improve SLAM accuracy

• Multi-robot coordination
Summary

• Plethora of planning approaches
  • Select based on available sensors, computation, & domain

• Coupled perception & planning design for max performance
  • Perception/localization-aware planning

• Resiliency to perception/localization errors & failures

• Testing in the wild crucial!
Questions?

**Presenter:** Brett T. Lopez

**Contact:** brett.t.lopez@jpl.nasa.gov