**Control Barrier Functions & Control Lyapunov Functions**

* General idea: if you can find a control barrier function, you can imply forward invariance of a set (with inequality constraints on control input). Once you have this you can combine it with a control lyapunov function to do online quadratic programming
* Pros:
  + Can make the problem just a QP
  + Can handle dynamic obstacles
* Cons:
  + Hard to find right CBF/CLF? Equation (5) and (12) in CDC 2014 paper seems to give a simple form for the barrier function
  + They don’t seem to address disturbances!

Control Barrier Function based Quadratic Programs with Application to Adaptive Cruise Control (CDC 2014)

A Ames, Jessy Grizzle, Paulo Tabuada (Texax A&M, U.Mich, UCLA)

* Control barrier functions and control lyapunov functions with quadractic programs
* If CBF can be found, this implies forward invariance of set (with inequality constraints on control input)
* General CBF construction method is absent since most constraints are imposed on the configuration variables only.
* No time-varying safety constraints
* Seems like the control function must be Lipschitz continuous (in HJ reachability it doesn’t have to be since we use bang-bang control)
* Not sure why there is so much fiddling in the numerical example…

Safety-Critical Control of a 3D Quadrotor with Range-Limited Sensing (DSCC 2016)

* uses control barrier functions. Enforces forward invariance of a set in state space through linear constraint on control input. Combines with control lyapunov functions to do online quadratic programming.
* Hard to generalize to other systems
* Can handle dynamic obstacles

Safety-Critical Control of a Planar Quadrotor (CDC 2016)

Gueofan Wu and Koushil Sreenath (CMU)

* Finds CBF for planar quadrotors
* Uses dynamic obstacle

**Decompose State Space into Convex Regions**

* General Idea: if you can split up the known area into convex regions, you can solve trajectories quickly
* Pros:
  + Breaks up free space to find path through sequence of convex regions
  + Guarantee that there exists a safe stopping control policy over a time horizon
  + Optimal in the sense that receding horizon control policy is based on minimizing trajectory snap
  + Can handle 7D dynamics at up to 2 m/s planning speed
  + Can make guarantees on safety
* Cons:
  + Requires simplified view of surrounding space?
  + Need to pre-compute convex regions? Unsure of this.

“Motion planning with sequential convex optimization and convex collision checking,” (International Journal of Robotics Research, 2014)

J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel

* Breaks workspace into free convex regionals
* Uses Sequetial quadratic programming to converge to a solution super fast
* Requires pre-built map with pre-computed convex regions; difficult to achieve in real-time

Safe Receding Horizon Control for Aggressive MAV Flight with Limited Range Sensing (IROS 2015)

Michael Watterson and Vijay Kumar (UPenn)

* Polyhedral decomposition of visible free space to abstract trajectory planning as problem of finding a path through a sequence of convex regions in configuration space
* Guarantee that there exists a safe stopping control policy over a time horizon
* Optimal in the sense that receding horizon control policy is based on minimizing trajectory snap
* 7D dynamics
* works with lmited sensing range
* average planning/execution speed: 2 m/s
* assumes obstacle within given sensing radius is known completely

High Speed Navigation For Quadrotors With Limited Onboard Sensing (ICRA 2016)

Sikang Liu, Michael Watterson, Sarah Tang, Vijay Kumar (UPenn)

Decompose state space into convex regions?

* Normal long range planning, with dynamic short-range planning that can avoid obstacles in real time
* Detects obstacles and create a map representation of the environment on-the-fly, explicitly incorporating computational demands of translating raw sensor data to a map for trajectory planning
* Propose novel short range planning policy that includes frontier-based method for finding promising paths towards the robot’s final goal and fast convex segmentation of a provided map that allows for real-time generation of optimal trajectories which accommodate the vehicle’s dynamics
* Simulation examples and experimental results that demonstrate complete algorithmic pipeline from perception of obstacles to execution of a desired trajectory
* Questions
  + Can it be done in real time? yes
  + Can it handle safety and/or disturbances? Create safe stopping policy for every short-range planning step
  + Can it handle high-dimensional models?

**Generate a Bunch of Trajectories; Pick the Best**

* General idea:Generating a bunch of possible trajectories in a very cheap way, then pick the best
* Pros:
  + Can work well and quickly for many systems
* Cons:
  + Rely on randomly-sampled trajectories finding collision-free paths, which can be a strong assumption

Stomp: Stochastic trajectory optimization for motion planning (ICRA 2011)

M Kalakrishnan, S CHitta, E Theodorou, P Pastor, S Schaal

* Gradient-free method, samples candidate trajectories and minimizes a cost function by creating linear combinations of the best-scoring candidates

U. Schwesinger, M. Rufli, P. Furgale, and R. Siegwart, “A samplingbased partial motion planning framework for system-compliant navigation along a reference path,” in Intelligent Vehicles Symposium (IV), pp. 391–396, IEEE, 2013.

* choosing safest trajectories for autonomous vehicles in traffic by generating a bunch of candidate paths then choosing the best

M. W. Mueller, M. Hehn, and R. D’Andrea, “A computationally efficient motion primitive for quadrocopter trajectory generation,” IEEE Transactions on Robotics, vol. 31, no. 6, pp. 1294–1310, 2015.

* Finding good polynomial trajectories to enable quadrotor ball juggling by generating a bunch of paths and choosing the best

P. Krusi, B. B ¨ ucheler, F. Pomerleau, U. Schwesinger, R. Siegwart, and ¨ P. Furgale, “Lighting-invariant adaptive route following using iterative closest point matching,” Journal of Field Robotics, vol. 32, no. 4, pp. 534–564, 2015.

* Finding locally lower-cost trajectories to track a global plan in rough terrain by generating a bunch of paths and choosing the best

**Motion Primitives**

* General idea:
  + Discretize state space into state lattice w/ motion primitives forming edges in the graph. Use standard graph search algorithms like A\* or AD\* to find feasible solution
* Pros:
  + Shown to work in many applications
  + Can sometimes guarantee safety
  + Can use only on-board vision and planning by picking collision-free next maneuvers from motion library
* Cons:
  + Need to discretize workspace and state space
  + Performance is tightly linked to how many motion primitives are generated

M. Pivtoraiko, D. Mellinger, and V. Kumar, “Incremental microuav motion replanning for exploring unknown environments,” in IEEE International Conference on Robotics and Automation (ICRA), pp. 2452–2458, IEEE, 2013. [16]

* Motion primitives, discretize state space, use graph search algorithm, nativate through partially known environment. Requires lots of motion primitives to work well

Motion Primitives and 3D path planning for fast flight through a forest (International Journal of Robotics research, 2015)

A. J. Barry, High-Speed Autonomous Obstacle Avoidance with Pushbroom Stereo. PhD thesis, Massachusetts Institute of Technology, 2016. [17]

Aditya Paranjape, Kevin Meier, Xichen Shi, Soon-Jo Chung, Seth Hutchinson (McGill, U of Illinois)

* motion primitives, simplified dynamics
* very shittily written

B. Landry, R. Deits, P. R. Florence, and R. Tedrake, “Aggressive quadrotor flight through cluttered environments using mixed integer programming,” 2016.

* Flies high-speed through forest using only on-board vision and planning, picking collision-free next maneuvers from motion library. Cites insufficient richness of motion primitive library and discretization in start state as responsible for 50% of experimental failures

**Polynomial Trajectory Optimization without Discrete Sampling**

* General idea:
  + Uses continuous-time basis function to express trajectories, and plan from arbitrary points in the state space to allow greater flexibility for online replanning
* Pros:
  + does not need to be discretized on a state space grid
  + under 50 ms for complete planning cycle, faster than sampling-based methods
  + can replan while maintaining smooth continuous paths
* Cons:
  + Requires simplified model of multicopter dynamics
  + Uses waypoints
  + Tends to be either overly conservative or overly aggressive due to map representation
  + Requires computing dense distance field over each voxel of original map; does not scale well

Continuous-Time Trajectory Optimization for Online UAV Replanning (IROS 2016) [1]

Helen Oleyinikova, Michael Burri, Zachary Taylor, Juan Nieto, Roland Siegwart, Enric Calceran (ETC Zurich)

* Works for partially known or unknown environments
* Use a bunch of polynomials as basis functions, vary one parameter (segment time) to ensure these trajectories are dynamically feasible given simplified model of multicopter dynamics
* Objective function minimizes derivative of position and cost of collision
* Under 50 ms for complete planning cycle (runts in real-time at 4 HZ, mean latency of 40 ms between acquiring depth data to generate feasible collision-free trajectory)
* Faster than smapling-based methods but solve small percentage of problems

**Sample with Waypoints, add Polynomial Smoothing**

* General idea:
  + sample a bunch of points, make a path, smooth it out. Often mixed-integer quadratic program
* Pros:
  + Outperforms traditional rrt methods in control space for execution time
  + Produces high-quality plans that are probabilistically complete
* Cons:
  + Too slow for online applications like real-time avoidance
  + Requires complete map of area

J. Bellingham, A. Richards, and J. P. How, “Receding horizon control of autonomous aerial vehicles,” in Proceedings of the 2002 American Control Conference (ACC), vol. 5, 2002, pp. 3741–3746

* Solves mixed integer linear program over receding horizon to find trajectories that incrementally move towards a goal, incorporating collision avoidance heuristic

Minimum Snap Trajectory Generation and Control for Quadrotors (ICRA 2011)

Daniel Mellinger and Vijay Kumar (UPenn)

* Goes beyond linearized models that are stable under small roll and pitch angles or when a control lyapunov function can be synthesized
* Uses piecewise polynomial trajectories that minimize cost functionals derived from the square of the norm of the snap
* Minimizes control effort indirectly by minimizing snap
* Plays with dynamics: decouples, linearizes, adds constraints, etc.
* Questions
  + Can it be done in real time? Kind of
  + Can it handle safety and/or disturbances? Trades speed for accuracy
  + Can it handle high-dimensional models? 12D

Polynomial Trajectory Planning for aggressive quadrotor flight in dense indoor environments (2013)

Charles richter, adam bry, Nicholas roy (MIT)

* RRT-based methods to generate visibility graph, then fitting high-order polynomials through way-points
* Better than traditional RRT methods in control space in terms of execution time
* uses polynomial path segments (stable for high-order polynomials)
* Used mixed-integer semidefinite programming
* Does not mention safety
* Requires full map of the environment
* Produces high-quality plans that are probabilistically complete
* Not real-time, too slow for real-time avoidance
* Assumes an octomap representation of the environment is available
* Smoothed trajectory may deviate from original path, allowing collisions. Adding intermediate waypoints would pull the trajectory closer, but adding more waypoints is costly and no guarantee on how many you need

Incremental micro-uav motion replanning for exploring unknown environments (ICRA 2013) [8]

* called "CHOMP"
* trajectory optimization-based motion planner
* uses two-way objective function with smoothness and collision cost
* performs gradient descent with positions of discrete waypoints as parameters
* step is then multiplied by riemennian metric to ensure smooth, incremental updates

Real-time visual-inertial mapping, re-localization and planning onboard mavs in unknown environments (IROS 2015)

M.Burri, H.Oleynikova, M.W. Achtelik, R. Sigwart

* Requires full map of the environment
* Produces high-quality plans that are probabilistically complete
* Almost real-time, too slow for real-time avoidance

**UNSORTED**

Real-time Obstacle Avoidance for Fast Mobile Robots (IEE transactions on systems, man, and cybernetics, 1989)

J Borenstein, Y Koren

* Robot running at max speed of .78 m/s
* Creates “virtual force field” to measure uncertainty of safety around the robot

Receding Horizon Path Planning with Implicit Safety Guarantees (ACC 2004)

Tom Schouwenaars, Jonathan How, Eric Feron (MIT)

* Used mixed integer linear programming
* Uses “loiter” circles to gather more info/enforce safety
* “quasi real time” = mostly 1 second iterations
* Planning horizon= 6 time steps

Robust Constrained Receding Horizon Control for Trajectory Planning (AIAA conference 2005)

Yoshiaki Kuwata, Tom Schouwenaars, Arthur Richards, Jonathan How (MIT)

High-speed Flight in an Ergodic Forest (arXiv, 2012)

* given map of entire forest
* shows velocity below which there exists a safe trajectory

M. Hehn and R. D’Andrea, “Real-time trajectory generation for interception maneuvers with quadrocopters,” in 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2012, pp. 4979–4984

* Generates minimum time trajectories using optimal control techniques
* Only validated in scenarios where environment is completely known a priori, assumes obstacle-free

Incremental micro-uav motion replanning for exploring unknown environments (ICRA 2013) [8]

* called "CHOMP"
* trajectory optimization-based motion planner
* uses two-way objective function with smoothness and collision cost
* performs gradient descent with positions of discrete waypoints as parameters
* step is then multiplied by riemennian metric to ensure smooth, incremental updates

M. Shomin and R. Hollis, “Fast, dynamic trajectory planning for a dynamically stable mobile robot,” in Proc. IEEE/RSJ Intl. Conf. on Intell. Robots and Syst., Chicago, IL, Sept 2014, pp. 3636–3641.

R. Deits and R. Tedrake, “Efficient mixed-integer planning for uavs in cluttered environments,” in Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 42–49

* Mixed integer quadratic programming. Impractical for real-time planning

Online Generation of Collision-Free Trajectories for Quadrotor Flight in Unknown Cluttered Environments (ICRA 2016)

Jing Chen, Tianbo Liu, Shaojie Shen (Hong Kong University)

* Instead of using convex regions to represent free-space, uses efficient operations in octree-based environment representation for online generation of collision-free flight corrider
* Uses efficient quadratic programming to find smooth trajectories with guaranteed safety
* Works better in confined environments compared to RRT\* methods

**Pavone’s paper**

Control structure

- u\_nom + k(x\*, x)

- appears to only work for two sets of the same dynamics

- should be flexible wrt planning method, but seems to use MPC for some reason

Offline

- Compute RTT set (RCI)

- worst-case disturbance

- finite time horizon (could be useful for our RTT work)

- independent of nominal trajectory

- unaugment target and augment obstacles

- online update of nominal trajectory allows smaller time horizon and therefore smaller "bubble" --> accounts for realized disturbance (Problem MPC)

Online:

- MPC ("Problem MPC")

- geodesic computation

Numerical results

- Problem MPC: 1s

- tracking controller: 5ms

- MPC iteration: 0.35s

- Disturbance: 0.1 m/s^2

- Tracking error: 0.76m semi major, 0.44m semi minor