The Dynamics Projection Filter (DPF) - Real-Time Nonlinear Trajectory Optimization Using Projection Operators

Sanjiban Choudhury[1] and Sebastian Scherer[1]

***Abstract*— Robotic navigation applications often require on-line generation of trajectories that respect underactuated non-linear dynamics, while optimizing a cost function that depends only on a low-dimensional workspace (collision avoidance). Approaches to non-linear optimization, such as differential dynamic programming (DDP), suffer from the drawbacks of slow convergence by being limited to stay within the trust-region of the linearized dynamics and having to integrate the dynamics with fine granularity at each iteration. We address the problem of decoupling the workspace optimization from the enforcement of non-linear constraints.**

In this paper, we introduce the Dynamics Projection Filter, a nonlinear projection operator based approach that first optimizes a workspace trajectory with reduced constraints and then projects (filters) it to a feasible configuration space trajectory that has a bounded sub-optimality guarantee. We show simulation results for various curvature and curvature-derivatives constrained systems, where the dynamics projection filter is able to, on average, produce similar quality solution 50 times faster than DDP. We also show results from flight tests on an autonomous helicopter that solved these problems on-line while avoiding mountains at high speed as well as trees and buildings as it came in to land.

I. Introduction

A common problem faced in robotics navigation applications is to generate on-line a smooth collision free trajectory that respects the non-linear constraints imposed by the dynamics of an under-actuated system. This problem is difficult because real-time perceptual information requires a fast response. At the same time, the trajectory computed must be dynamically feasible, have a low cost and reach the goal.

Trajectory optimization with non-linear constraints is commonly solved using variants of differential dynamic programming [1] or sequential convex optimization [2]. [3] uses sequential quadratic programming and deals with obstacles through the use of signed distances. More complex dynamics models have been considered by [4], [5], [6]. However the nonlinear nature of the constraint makes this method slow. There has been a lot of success in the field of fast high quality unconstrained optimizations, such as CHOMP [7]. The reason for this is the use of workspace gradients and parameterization invariance.

In this paper, we attempt to make a bridge between fast optimization of objectives that depend only on the low dimensional workspace and ensuring high dimensional constraint satisfaction using projection operators. [8] learns a low dimensional structure automatically, but only for holonomic optimization. The work most similar to ours [9] uses a projection operator, however, the underlying trajectory is still high dimensional, the gradients contain the nonlinear constraint artifacts requiring small step sizes and the projection has no guarantees.

[1]The Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA sanjiban,basti@cmu.edu

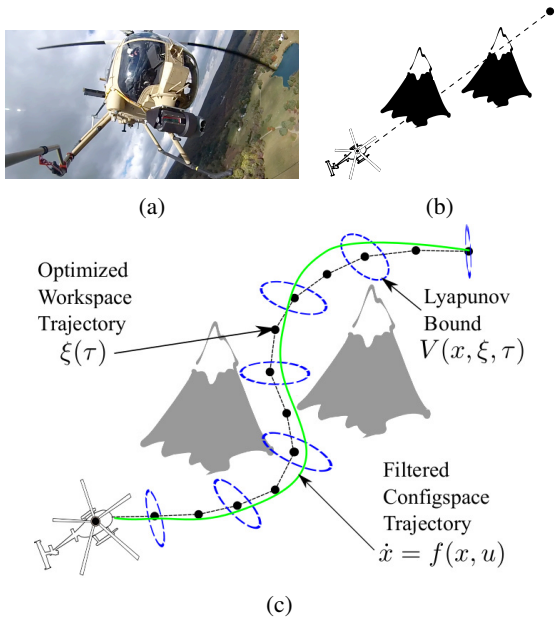


Fig. 1: (a) Boeing's Unmanned LittleBird is guided by the optimization computing trajectories in real-time. (b) A typical scenario where the helicopter has discovered some mountains in the way and has to plan around them (c) An illustration of our approach. We optimize a workspace trajectory $\xi(\tau)$ subject to smoothness constraints, project it to a feasible configuration space $\dot{x} = f(x, u)$ while ensuring bounded suboptimality using Lyapunov bounds $V(x, \xi, \tau)$.

In this paper, we present the *Dynamics Projection Filter*. Our main contributions are as follows:

- We present a *real-time* approach to solving a non-linear trajectory optimization where the cost function only depends on a low-dimensional workspace.(Fig. 1)
- We define a nonlinear projection operator as a control Lyapunov function that takes an optimized *workspace* trajectory and projects it to a *configuration space* trajectory with guarantees on sub-optimality.
- Results on an autonmous helicopter performing missions at high speeds.

II. Problem Statement

Let $X \subset \mathbb{R}^n$ be the configuration space of the vehicle, $W \subset \mathbb{R}^w$ be the workspace of the vehicle (w is either 2 or 3) and $U \subset \mathbb{R}^m$ be the control space. Let T be the

time horizon of the problem. Let the configuration space trajectory represented by the pair $x : [0, T] \rightarrow X$ and $u : [0, T] \rightarrow U$ be a mapping from time to configuration and control respectively. The trajectories are subject to nonlinear constraints $\dot{x}(t) = f(x(t), u(t))$. Let the function $w : X \rightarrow W$ be the workspace projection of a configuration state

In the optimization problem, the start state is x_0 and the desired end state is x_f. Let $J(x(t)) = \int_0^T c(x(t))\mathrm{dt}$ be a line integral cost. Let $c_f(x(T), x_f)$ be a terminal cost function. The problem we wish to solve is as follows

$$\begin{aligned} \underset{x(t),u(t)}{\text{minimize}} \quad & J(x(t)) + c_f(x(T), x_f) \\ \text{subject to} \quad & x(0) = x_0 \\ & \dot{x}(t) = f(x(t), u(t)) \end{aligned} \tag{1}$$

Problem (1) is fairly generic. However we wish to solve a special case where $J(x(t))$ is only dependent on the workspace component and has the following structure

$$J(x(t)) = J_{obs}(x(t)) + \lambda\, J_{smooth}(x(t)) \tag{2}$$

where λ is a weighting parameter. Similarly $c_f(x(t), x(T)) = \frac{1}{2} \|w(x(T)) - w(x_f)\|$.

$J_{obs}(x(t))$ is an obstacle cost function

$$J_{obs}(x(t)) = \int_0^T c_{obs}(w(x(t))) \left\| \frac{\mathrm{d}}{\mathrm{dt}} w(x(t)) \right\| \mathrm{dt}$$

where $c_{obs}(w(x(t)))$ is inversely proportional to the squared distance to obstacle (Refer to [10] for details)

$J_{smooth}(x(t))$ penalizes high velocities in workspace

$$J_{smooth}(x(t)) = \frac{T}{2} \int_0^T \left\| \frac{\mathrm{d}}{\mathrm{dt}} w(x(t)) \right\|^2 \mathrm{dt}$$

The following assumptions are made for the class of problems we look at in this paper

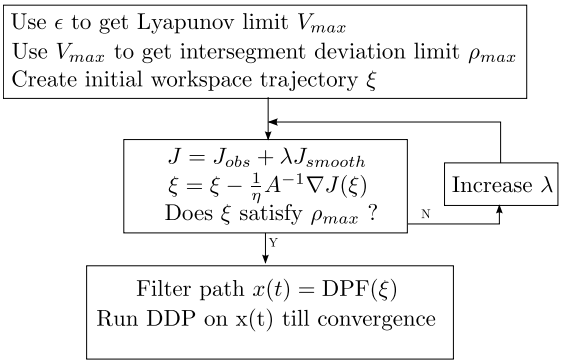
- The workspace dimension w is lower than the configuration space dimension n.
- Since the cost function is only dependent on workspace, the dynamics $f(.)$ is the only term that deal with the full configuration space.
- The dynamics $f(.)$ are that of a mobile robot. $f(.)$ satisfies properties such as global controllability and controllers can be derived that can track feasible reference trajectories.

III. Background: Differential Dynamic Programming

Differential dynamic programming (DDP) [1] is a discrete-time second order trajectory optimization method that uses quadratic approximations of the cost function and dynamics, but also features quadratic convergence. It involves an iterative backward and forward pass along the trajectory. In the backward pass, a quadratic approximation of the value function and a resultant linear control policy is computed. In the forward pass, the control policy is applied to get a new trajectory.

Despite DDP being much more efficient than full Newton's method on the trajectory, it has two bottlenecks in speed. Firstly, the integration granularity in the forward step needs to be very high - otherwise the integration errors accumulate along the trajectory driving the states at the end out of the linearization trust region. Secondly, the control update can be unbounded if the approximation matrices are ill-conditioned - this happens when the control change is orthogonal to the cost function change. Small steps are required to exit this condition. Moreover the method is prone to get stuck in bad local minima because the dynamics might locally restrict a large jump.

IV. Approach: Dynamics Projection Filter



(a)

Fig. 2: Dynamics Projection Filter Overview

In this section, we present our novel approach - the *Dynamics Projection Filter* (DPF). DPF is a nonlinear projection operator that projects a workspace trajectory to a dynamically feasible configuration space trajectory. DPF guarantees a fixed suboptimality bound in cost increase for this projection subject to constraints that the workspace trajectory must satisfy. This property is used to solve the planning problem in (1) using a set of procedures as shown in Fig. 2. A workspace trajectory is optimized with respect to the cost function in (2) subject to the afore-mentioned constraints. Then DPF is applied to this trajectory to obtain a configuration space trajectory. The configuration space trajectory is further optimized using DDP (Section III).

The main advantage to this approach is that most of the optimization is performed without explicitly dealing with the nonlinear dynamics constraints. The workspace optimization is low-dimensional, at a coarser granularity than full state space optimization and also eliminates the need to repeatedly linearize the constraints and integrate the dynamics. Even though DDP is a much more general approach, we exploit the assumptions made in the problem statement to come up with a computationally cheaper solution.

A. Lyapunov Stabilized Workspace Trajectory

The DPF accepts a workspace trajectory as input, uses a controller to track this trajectory and the outputs the

configuration trajectory traced out by the system. To provide guarantees about the output, the first component required is a control-Lyapunov function (CLF) that stabilizes around a feasible workspace trajectory. Since the controller can observe all states, has a perfect model and is free from any disturbance, approaches such as feedback linearization and backstepping can be applied [11], [12], [13].

Let the workspace trajectory be $\xi : [0,1] \rightarrow \mathbb{R}^w$. A control-Lyapunov stabilized trajectory firstly requires the existence of a feedback control $u = K(x, \xi, \tau)$ where $x \in \mathbb{R}^d$ is the configuration space coordinate of the robot, τ is the index of the workspace trajectory. It also requires the definition of index dynamics $\dot{\tau} = \Gamma(x, \xi, \tau)$. Careful selection of these functions can ensure a function $V(x, \xi, \tau)$ to satisfy the Lyapunov criteria when ξ is dynamically feasible. If $x \in X_\xi$ implies perfect tracking, the Lyapunov criteria is $V(x,.) > 0, \dot{V}(x,.) < 0, \forall x \in X \setminus X_\xi$ and $V(x,.) = 0, \forall x \in X_\xi$ (globally asymptotic stable version).

A further local exponential stability is also required, i.e,

$$\left|\frac{\mathrm{d}V(.,\tau)}{\mathrm{d}\tau}\right| > \alpha V(.,\tau),\ \alpha > 0,\ V(.) < V_{max}.$$

where the rate of exponential stability α determines how fast convergence occurs. This property will be used to guarantee decay properties of the CLF. A detailed derivation of these properties for a fixed wing UAV is provided in Section V.

The output $x(t) = \mathrm{DPF}(\xi)$ is given by

$$\begin{bmatrix} x(t) \\ \tau(t) \end{bmatrix} = \begin{bmatrix} x(0) \\ 0 \end{bmatrix} + \int_0^t \begin{bmatrix} f(x(t), K(x(t), \xi, \tau(t))) \\ \Gamma(x(t), \xi, \tau(t)) \end{bmatrix} \mathrm{dt} \quad (3)$$

In the scope of this work, we consider ξ to be approximated by a set of workspace samples at equal discretization $\Delta\tau$: $\xi \approx (q_1, q_2, \ldots, q_n)^{\mathrm{T}} \in \mathbb{R}^{n \times w}$, with q_0 and q_{n+1} the fixed starting and ending points of the trajectory. This will facilitate concrete expressions on bounds that are parameterization dependent. However, the method remains valid for other parameterizations, such as splines.

Under the assumption that the linear segments between waypoints are dynamically feasible (there exists $u \in U$ which allows perfect tracking) , the Lyapunov function $V(.)$ converges in the straight line portion of ξ. However, at every waypoint it increases by $\Delta V > 0$ because of the angle change at the waypoint. The more jagged ξ is the more the cumulative effect of ΔV will be. The maximum $V(.)$ at anytime determines how much deviation $x(.)$ has from ξ. We first establish that under certain assumptions about $V(.)$, a bounded Lyapunov value implies a bounded intersegment deviation.

Proposition 1 (Bounded Intersegment Deviation)**.** *Given a desired bound on the Lyapunov function $V(x, \xi, \tau) \leq V_{max}, \forall x, \tau \in \mathrm{DPF}(\xi)$, the intersegment deviation is also bounded, i.e. $\left(1 + \frac{(q_i - q_{i-1})^{\mathrm{T}}(q_i - q_{i+1})}{\|q_i - q_{i-1}\|\|q_i - q_{i+1}\|}\right) \leq \rho_{max}$.*

Proof. Let $\rho_i = \left(1 + \frac{(q_i - q_{i-1})^{\mathrm{T}}(q_i - q_{i+1})}{\|q_i - q_{i-1}\|\|q_i - q_{i+1}\|}\right)$ be the intersegment deviation and V_i be the Lyapunov value at waypoint i. Let the inflation of Lyapunov value be ΔV_i on transitioning to the segment leading to waypoint $i+1$. We define the relation $\Delta V_i \leq h(V_i, \rho_i)$, where a specific form of $h(.)$ depends on the form of $V(.)$ as we show in Section V. Let $l = \min_i \|q_i - q_{i+1}\|$. The exponential stability α implies that along l, the Lyapunov decays from V to $Ve^{-\alpha l}$.

We now find the largest ρ_i that guarantees $V_{i+1} \leq V_i$.

$$\begin{aligned} V_{i+1} &\leq V_i \\ (V_i + \Delta V_i)e^{-\alpha l} &\leq V_i \\ \Delta V_i &\leq V_i(e^{\alpha l} - 1) \\ h(V_i, \rho_i) &\leq V_i(e^{\alpha l} - 1) \\ \rho_i &\leq h^{-1}(V_i(e^{\alpha l} - 1)) \end{aligned} \quad (4)$$

where $h^{-1}(.)$ is an abuse of notation to imply that the relation is invertible.

We assume that $h^{-1}(.)$ monotonically increases with V_i so that the tightest bound is achieved when $V_{i-1} + \Delta V_{i-1} = V_{max}$ and decays to $V_i = V_{max}e^{-\alpha l}$.

$$\begin{aligned} \rho_i &\leq h^{-1}(V_{max}e^{-\alpha l}(e^{\alpha l} - 1)) \\ \rho_i &\leq h^{-1}(V_{max}(1 - e^{-\alpha l})) \end{aligned} \quad (5)$$

□

We now prove that bounded Lyapunov value implies $x(t) = \mathrm{DPF}(\xi)$ is within bounded suboptimality of ξ (by abuse of notation $J(.)$ serves a dual role of cost evaluation of ξ and x)

Proposition 2 (Bounded Suboptimality)**.** *Given a desired bound on the Lyapunov function $V(x, \xi, \tau) \leq V_{max}, \forall x, \tau \in \mathrm{DPF}(\xi)$, $J(x) \leq (1 + \epsilon)J(\xi)$*

Proof. (Sketch) The workspace deviation is bounded, $\|w(x(t)) - \xi(\tau)\| \leq d_{max}$ since $V(x, \xi, \tau) \leq V_{max}$. Under the assumption that DPF always ensures $\dot{\tau} > 0$, the total length is within bounded inflation $\|w(x(t))\| \leq (1 + \gamma_l)\|\xi(\tau)\|$, where γ_l is an inflation constant derived from the dynamics constraints. This implies a bounded smoothness cost factor, $J_{smooth}(x) \leq (1 + \epsilon_s)J_{smooth}(\xi)$ where ϵ_s is a sub-optimality bound. The function $c_{obs}(.)$ is assumed to be Lipschitz continuous which gives $J_{obs}(x) \leq (1 + \epsilon_o)J_{obs}(\xi)$. □

Since the projected trajectory is not guaranteed to be locally optimal, executing a DDP as a post-processing step results in local optimality.

B. Workspace Optimization

The workspace optimization problem is to optimize $J(\xi)$ subject to constraints at each waypoint ρ_i stated in proposition 1. While a wide variety of approaches exist to solve this problem [2], [14], we use CHOMP [7]. CHOMP optimizes the same objective functional but has a lot of added advantages such as using a steepest descent direction with respect to a Riemannian metric that allows fast convergence as well as invariance to parameterization. By selecting a

parameterization only dependent on the cost function granularity and independent of the dynamics, CHOMP is able to make large progress away from obstacles while still retaining smoothness.

Let ξ_i be the trajectory at the i^{th} iteration. Let A be a measure of the acceleration along a trajectory. Then the update rule for CHOMP is

$$\xi_{i+1} = \xi_i - \frac{1}{\eta} A^{-1} \nabla J(\xi_i)$$

where η is the step size.

CHOMP does not explicitly guarantee satisfaction of workspace constraints. However, we take advantage of the fact that minimization of the smoothness objective at each waypoint of ξ is equivalent to minimization of ρ_i and hence contributes to constraint satisfaction. CHOMP is executed till convergence and the constraint is checked for. If it is not met, the smoothness term λ is increased and the process is repeated. Fig. 2 shows the overall process.

V. Derivations for Fixed Wing UAV

We go into a detailed example because the realization of the dynamics filter is strongly coupled with the system. We select a 2D fixed wing UAV ($n = 3$) for simplicity although our end application is for a $n = 12$ system.

Let the $\mathbb{R}^3$ configuration space be $X = [x, y, \psi]^T$ and the workspace be $\mathbb{R}^2$. The speed of the uav is v and the angular speed (control input) is ω. Then the dynamics of the system are represented by

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} v\cos\psi \\ v\sin\psi \\ \omega \end{bmatrix}$$

where $\mid \omega \mid \leq \omega_{max}$

A. Controller Synthesis

Let the workspace trajectory be $\xi(s) = (x_r(s), y_r(s))$, where s is an arc-length parameterization. The instantaneous direction of the workspace trajectory is $\psi_r(s) = \tan^{-1} \frac{\dot{y}_r(s)}{\dot{x}_r(s)}$ and the curvature of the trajectory is $\kappa(s)$.

We will now define the error dynamics of the system with respect to $\xi(s)$. Let $p = (x, y)$ be the position of the vehicle. The error vector is defined as the position difference from the workspace trajectory frame

$$\begin{bmatrix} e_s \\ e_d \\ \dot{\psi} \end{bmatrix} = R^T(\psi_r(s))(p - \xi(s))$$

where e_s is the along track error, e_d is the cross track error and R is the rotation matrix. The error state space is then composed of e_s, e_d, heading error $\chi = \psi - \psi_r(s)$ and the reference index s. The error dynamics are as follows

$$\begin{bmatrix} \dot{e}_s \\ \dot{e}_d \\ \dot{\chi} \\ \dot{s} \end{bmatrix} = \begin{bmatrix} v\cos\chi - (1 - \kappa(s)e_d)\dot{s} \\ v\sin\chi - \kappa(s)e_s\dot{s} \\ \omega - \kappa(s)\dot{s} \\ k_s e_s + v\cos\chi \end{bmatrix}$$

The following control Lyapunov function [12] is used

$$V(s) = \frac{1}{2\gamma}(e_s^2 + e_d^2) + \frac{1}{2}(\chi - \delta(e_d))^2$$

where γ is a scaling factor and $\delta(e_d) = -\chi^\infty \frac{e^{2ke_d} - 1}{e^{2ke_d} + 1}$ generates a steering angle based on the cross track error [11].

The function $V(s)$ satisfies the conditions $V(s) \geq 0$. If

$$\begin{aligned} \omega \quad &= -k_\omega(\chi - \delta(e_d)) + \kappa(s)\dot{s} + \delta'(e_d)(v\sin\chi - \kappa(s)e_s\dot{s}) \\ &\quad - \frac{e_d v}{\gamma}\left(\frac{\sin\chi - \sin(\delta(e_d))}{\chi - \delta(e_d)}\right) \end{aligned} \tag{6}$$

we get the derivative of the Lyapunov function

$$\dot{V}(s) = -\frac{k_s}{\gamma}e_s^2 + \frac{e_d v}{\gamma}\sin(\delta(e_d)) - k_\omega(\chi - \delta(e_d))^2$$

Since $e_d \sin\delta(e_d) \leq 0$, $\dot{V}(s) \leq 0$. Moreover, since $\dot{V}(s) = 0$ only for $V(s) = 0$, by LaSalle's invariance principle, the system is stable.

B. Exponential stability of controller

Proposition 3 (Exponential Stability)**.** *Given a bound on the Lyapunov value $V(s) < V_{max}$, the controller is exponentially stable , i.e., $\left|\frac{\mathrm{d}V(s)}{\mathrm{d}s}\right| > \alpha V(s), \alpha > 0$.*

Proof. The time derivative $\dot{V}(s)$ was derived as

$$\dot{V}(s) = -\frac{k_s}{\gamma}e_s^2 + \frac{e_d v}{\gamma}\sin(\delta(e_d)) - k_\omega(\chi - \delta(e_d))^2$$

Rewriting as a derivative w.r.t arc length $\frac{\mathrm{d}V(s)}{\mathrm{d}s} = \frac{\dot{V}(s)}{\dot{s}}$

$$\frac{\mathrm{d}V(s)}{\mathrm{d}s} = \frac{1}{\dot{s}}(-\frac{k_s}{\gamma}e_s^2 + \frac{e_d v}{\gamma}\sin(\delta(e_d)) - k_\omega(\tilde{\chi} - \delta(e_d))^2)$$

$$\left|\frac{\mathrm{d}V(s)}{\mathrm{d}s}\right| \geq \frac{1}{\dot{s}_{max}}(\frac{k_s}{\gamma}e_s^2 - \frac{e_d v}{\gamma}\sin(\delta(e_d)) + k_\omega(\tilde{\chi} - \delta(e_d))^2)$$

since $e_d \sin\delta(e_d) \leq 0$.

For $V(s) < V_{max}$, we can find a limit $e_s < e_{s,max}$ and $e_d < e_{d,max}$. This makes $\dot{s}_{max} = k_s e_{s,max} + v$. Note that $\left|\frac{\sin(\delta(e_d))}{\chi^\infty k e_d}\right|$ is monotonically decreasing function for e_d. Thus $\exists L_\delta, -\sin(\delta(e_d)) \geq L_{ed}\chi^\infty k e_d, e_d \leq e_{d,max}$

$$\left|\frac{\mathrm{d}V(s)}{\mathrm{d}s}\right| \geq \frac{1}{\dot{s}_{max}}(\frac{k_s}{\gamma}e_s^2 + L_\delta\frac{e_d^2 v}{\gamma} + k_\omega(\chi - \delta(e_d))^2)$$

$$\alpha = \frac{2}{k_s e_{s,max} + v}\min(k_s, L_\delta\chi^\infty k v, k_\omega)$$

□

C. Angle constraints on a piecewise linear workspace trajectory

We will now derive a bound on the angle deviation $|\Delta\psi_i|$ between successive segments such that $V_i \leq V_{max}$.

Proposition 4 (Angular Deviation Bound)**.** *The Lyapunov value is bounded $V_i \leq V_{max}$ if the angle deviation satisfies $|\Delta\psi_i| < \sqrt{2V_{max}}(1 - e^{-\frac{\alpha l}{2}})$, where l is the minumum inter-waypoint distance.*

TABLE I: DDP and DPF comparison for 1000 trial runs

Criteria	DDP	DPF	DPF (w/o post DDP)
Cost	337.1 (±161.9)	284.0 (±135.4)	310.1 (±140.3)
Time [s]	10.13 (±4.976)	1.238 (±0.051)	0.209 (±0.029)
Cost Ratio	–	0.906 (±0.370)	1.000 (±0.396)
Speed Up	–	8.198 (±4.035)	49.49 (±25.57)

TABLE II: Analysis of DPF process for 1000 trials

Criteria	CHOMP (Step I)	Filter (Step II)	DDP (Step III)
Cost Ratio Reduction	1.799 (±5.677)	−0.035 (±0.111)	0.130 (±0.302)
Time [s]	0.133 (±0.020)	0.076 (±0.018)	1.029 (±0.041)

TABLE III: DPF performance in autonomous helicopter flights

Criteria	Value
Filter Projection Loss	0.001 (±0.002)
DPF Time [s]	0.066 (±0.086)
Optimization Length [m]	4911 (±3600)

Proof. Let V_i be the Lyapunov value and $\Delta\psi_i$ be the angular deviation at waypoint i. Let the inflation of Lyapunov value be ΔV_i. (It is trivial to see $e_s^2 + e_d^2$ doesnt change). Then

$$\begin{aligned}\Delta V_i = &\frac{1}{2\gamma}(e_s^2+e_d^2)+\frac{1}{2}(\chi-\delta(e_d)+\Delta\psi_i)^2 \\ &-\frac{1}{2\gamma}(e_s^2+e_d^2)+\frac{1}{2}(\chi-\delta(e_d))^2 \\ = &\frac{1}{2}\Delta\psi_i^2+\Delta\psi_i(\chi-\delta(e_d)) \\ \leq &\max_{e_s,e_d,\chi}\frac{1}{2}\Delta\psi_i^2+\Delta\psi_i\sqrt{2(V_i-\frac{1}{2\gamma}(e_s^2+e_d^2))} \\ \leq &\frac{1}{2}\Delta\psi_i^2+\Delta\psi_i\sqrt{2V_i}\end{aligned} \tag{7}$$

Referring to Proposition 1, we see that ρ_i is proportional to ψ_i. Applying (7) to (4), and substituting ψ_i

$$\begin{aligned}\frac{1}{2}\Delta\psi_i^2+\Delta\psi_i\sqrt{2V_i} \leq &\quad V_i(e^{\alpha l}-1) \\ \Delta\psi_i \leq &\quad \sqrt{2V_i}(\sqrt{e^{\alpha l}}-1)\end{aligned} \tag{8}$$

Applying the result to (5)

$$\Delta\psi_i \leq \sqrt{2V_{max}}(1-e^{\frac{-\alpha l}{2}})$$

□

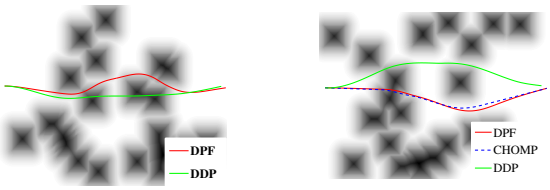
VI. Results

A. Fixed Wing UAV

The model we used for the fixed wing has $v = 20^{\mathrm{m}}/_{\mathrm{s}}$, $\omega_{max} = 0.8496^{\mathrm{rad}}/_{\mathrm{s}}$. The dynamics filter used a controller with the following parameters: $\gamma = 700$, $k_s = 1$, $k_w = 0.98$, $k = 0.08$, $\chi^\infty = 0.785$. The lyapunov limit V_{max} was chosen from $e_{d,max} = 40\mathrm{m}$. DDP integrates the dynamics at $0.02s$ which corresponds to 500 points if $T = 10s$. For the same length, CHOMP chose $N = 100$ waypoints such that the obstacle cost can be measured to an acceptable accuracy.

To highlight the difference between DDP and DPF, we show an example in Fig. 3 of a 'Wall Baffle' - two obstacle placed in such a way that the vehicle has to avoid them in opposite directions. DDP gets stuck in a local minima while DPF is 18 times faster and has a lower cost (almost the global minima).

We benchmark tested the two algorithms for 1000 planning problems in a random poisson obstacle field (1 obstacle every $2744\mathrm{m}^2$). Table I shows a summary of results. On average DPF was more than 8 times faster while still being more optimal on average. Table II shows the breakdown of the DPF performance in terms of cost ratio reduction of each step($\frac{-\Delta c}{c^*}$) and time taken. CHOMP is the fastest and most effective component while DDP takes the most time and offers little benefit. Hence we show that without DDP, we reach speed-up of 50 while still producing similar quality. We show success and failure cases in Fig. 4.



(a) (b)

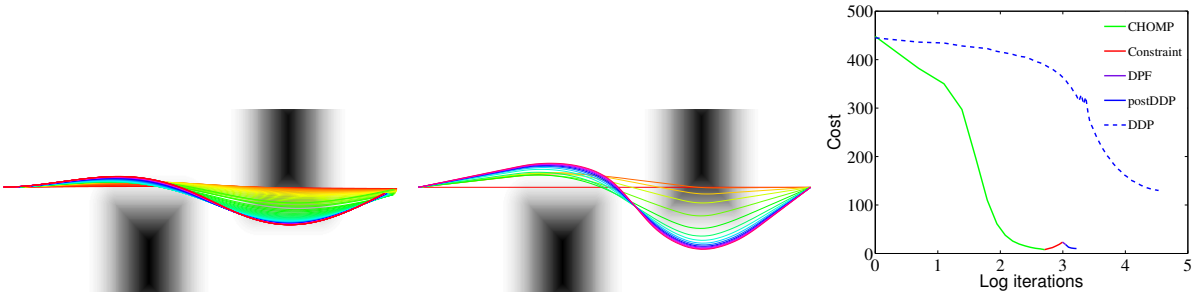
Fig. 4: Success and failure cases from 1000 trials in a poisson obstacle field. (a) DDP fails while DPF succeeds. DDP is stuck in a bad local minima due to obstacles on both sides. DPF takes larger steps to move to one of the sides and then is able to satisfy constraints (b) DPF fails while DDP succeeds. CHOMP moves in a direction by taking a large step where even though it can converge, it cannot satisfy the DPF constraints and hence the filtered path passes through obstacles. DDP taking smaller steps moves to the other side into a good minima.

B. Autonmous Helicopter

The dynamics projection filter has been stress tested on an autonomous helicopter over a period of 4 months flying various missions (more results in [15]). The DPF for this complex helicopter model used a controller that mimicked the closed loop response of a helicopter tracking waypoints. The constraints of the system is described in details in [15]. We did not run the post-processing DDP because it provided little benefit and had a very high run time. Table III shows the average performance during the missions. The DPF suffered minimal projection loss from the infeasible path and ran in real time optimizing paths kilometeres in length. Fig 5 shows some scenarios faced during actual missions. The DPF was always able to produce high quality collision free paths.

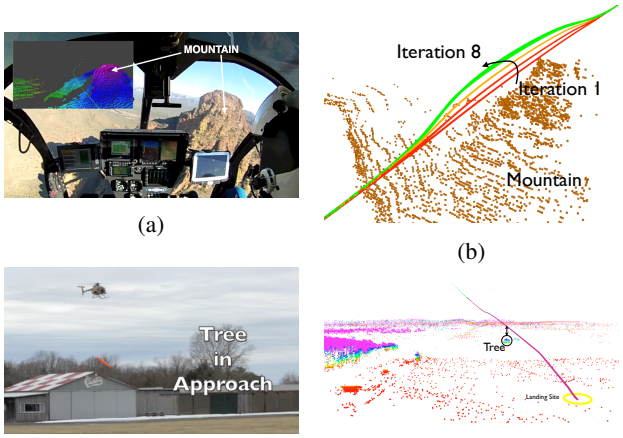
VII. Conclusion

In this paper, we have presented an approach for real-time trajectory optimization while respecting non-linear constraints. The method is based on optimizing in workspace and then projecting the result to configuration space. We show that by systematic design of the projection operator and



(a) (b) (c)

Fig. 3: A difficult 'Wall Baffle' environment. The global minima is between two bad local minimas. The cost map is shown in gray scale (darker means higher cost). (a) DDP takes small steps initially due to ill-conditioned matrices. It eventually converges to a local mimina in the configuration space. (b) DPF takes large steps to first converge in terms of the cost function and then ensure constraints are met. The filtered trajectory is in the valley of the global minima and post DDP ensures convergence. (c) The cost history during iterations for both algorithms. A semi-log plot is used as DDP takes far more iterations. CHOMP converges to minima by the 15^{th} iteration ($t = 0.023$ s), satisfies constraints by 20^{th} ($t = 0.03$), filters ($t = 0.11$) and post DDP takes 4 iterations to converge ($t = 1.55s$). On the other hand, DDP takes 93 iterations and 29.08s.



(a) (b) (c) (d)

Fig. 5: DPF solves real-time nonlinear optimization to guide a fully autonomous helicopter. (a) Avoiding an unmapped mountain in Mesa, AZ. View of the mountain from the helicopter cockpit after the helicopter has avoided it. (b) DPF converges in 8 iterations and takes 0.5s. (c) Avoiding trees on approach during landing. (d) DPF converges within 3 iterations and takes 0.2s.

enforcing simple constraints on the workspace optimization, the suboptimality of the solution can be guaranteed. We present derivations and results with a simplistic model of a fixed wing UAV. We have tested this approach on an autonomous helicopter and present results where the optimization allowed the helicopter to avoid mountains and trees while operating at high speeds. In future work, we wish to overcome the Lyapunov constraint and substitute it with softer requirements.

VIII. Acknowledgement

This work would not have been possible without the dedicated efforts of the entire AACUS TALOS team and was supported by ONR under contract N00014-12-C-0671.

References

[1] D. Mayne, "A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems," *International Journal of Control*, vol. 3, no. 1, pp. 85–95, 1966.

[2] J. Schulman, J. Ho, A. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization." in *Robotics: Science and Systems*, vol. 9, no. 1. Citeseer, 2013, pp. 1–10.

[3] R. Lampariello, D. Nguyen-Tuong, C. Castellini, G. Hirzinger, and J. Peters, "Trajectory planning for optimal robot catching in real-time," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 3719–3726.

[4] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of complex behaviors through contact-invariant optimization," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, p. 43, 2012.

[5] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 4906–4913.

[6] T. Erez and E. Todorov, "Trajectory optimization for domains with contacts using inverse dynamics," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 4914–4919.

[7] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "Chomp: Covariant hamiltonian optimization for motion planning," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[8] P. Vernaza and D. D. Lee, "Learning dimensional descent for optimal motion planning in high-dimensional spaces." in *AAAI*, 2011.

[9] J. Hauser, "A projection operator approach to the optimization of trajectory functionals," in *IFAC world congress*, 2002.

[10] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 489–494.

[11] A. Micaelli, C. Samson, *et al.*, "Trajectory tracking for unicycle-type and two-steering-wheels mobile robots," 1993.

[12] L. Lapierre, R. Zapata, and P. Lepinay, "Combined path-following and obstacle avoidance control of a wheeled robot," *The International Journal of Robotics Research*, vol. 26, no. 4, pp. 361–375, 2007.

[13] D. Jung and P. Tsiotras, "Bank-to-turn control for a small uav using backstepping and parameter adaptation," *Jung*, 2008.

[14] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "Stomp: Stochastic trajectory optimization for motion planning," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 4569–4574.

[15] S. Choudhury, S. Arora, and S. Scherer, "The planner ensemble and trajectory executive: A high performance motion planning system with guaranteed safety," in *AHS 70th Annual Forum, Montreal, Quebec, Canada*, May 2014.