Goals:

- Introduce two degree of freedom design for motion control systems
- Describe how to use flatness for real-time motion planning using NTG
- Give examples of implementation on Caltech ducted fan, satellite formations

Reading:

**Real-Time Trajectory Generation Using Flatness**

Nonlinear design
- global nonlinearities
- input saturation
- state space constraints

Linear design

![Diagram](image)

**Approach: Two Degree of Freedom Design**
- Use online trajectory generation to construct feasible trajectories
- Use linear control for local performance
- For many systems, dynamics are differentially flat \(\Rightarrow\) reduce dynamic system to algebraic equivalent and generate feasible trajectories in real time

**Rapid Transition from Hover to Forward Flight**

Caltech Ducted Fan

Real-Time Trajectory Generation

CDS 270-2, 10 Apr 06

R. M. Murray, Caltech CDS
Trajectory Generation Using Differential Flatness

\[ \mathcal{X} = f(x, u) \]
\[ z = h(x, u, \mathcal{G}K, u^{(p)}) \]
\[ |u| < L \]

\[ x = x(z, \mathcal{G}K, z^{(q)}) \]
\[ u = u(z, \mathcal{G}K, z^{(q)}) \]

Complicated (algebraic) constraints

\[ \bar{z}_0 = \begin{bmatrix} z(0) \\ \mathcal{G}(0) \\ \mathcal{G}T(0) \\ M \\ z^{(q)}(0) \end{bmatrix}, \quad z \sim \bar{z}_f = \begin{bmatrix} z(T) \\ \mathcal{G}(T) \\ \mathcal{G}T(T) \\ M \\ z^{(q)}(T) \end{bmatrix} \]

\[ z = \sum \alpha_i \psi^i(t) \]
\[ M\alpha = \begin{bmatrix} \bar{z}_0 \\ \bar{z}_f \end{bmatrix} \]

- Use basis functions to parameterize output \( \Rightarrow \) linear problem in terms of coefficients
Optimal Control Using Differential Flatness

Can also solve constrained optimization problem via flatness

\[
\min J = \int_{t_0}^{T} L(x, u) \, dt + V(x(T), u(T))
\]

subject to

\[
\dot{x} = f(x, u) \quad g(x, u) \leq 0
\]

• Input constraints
• State constraints

If system is flat, once again we get an \textit{algebraic} problem:

\[
x = x(z, \mathcal{K}^{(q)}, z^{(q)})
\]
\[
u = u(z, \mathcal{K}^{(q)}, z^{(q)})
\]
\[
z = \sum \alpha_i \psi^i(t)
\]

\[
\min J = \int_{t_0}^{T} L(\alpha, t) \, dt + V(\alpha)
\]

\[
g(\alpha, t) \leq 0
\]

Finite parameter \textit{optimization} problem

• Constraints hold at all times \(\Rightarrow\) potentially over-constrained optimization
• Numerically solve by discretizing time (collocation)
NTG: Nonlinear Trajectory Generation

Flatness-based optimal control package
• B-spline representation of (partially) flat outputs
• Collocation based optimization approach
• Built on NPSOL optimization pkg (requires license)
• Warm start capability for receding horizon control

Solves general nonlinear optimization problem
\[
\min J = \int_{t_0}^{T} q(x, u) \, dt + V(x(T), u(T)) \\
\dot{x} = f(x, u) \quad lb \leq g(x, u) \leq ub
\]

• Assumes \( x \) and \( u \) are given in terms of (partially) flat outputs
• Constraints are enforced at a user-specified set of collocation points
• Gives *approximate* solution; need to use w/ feedback to ensure robustness (2 DOF)

http://www.cds.caltech.edu/~murray/software/2002a_ntg.html
Trajectory Generation Using Splines for Flat Outputs

Rewrite flat outputs in terms of splines

$$z_j = \sum_{i=1}^{p_j} B_{i,k_j}(t) C_i^j$$ for the knot sequence $t_j$

$$p_j = l_j(k_j - m_j) + m_j$$

Evaluate constrained optimization at collocation points:

$$\min_{\tilde{C} \in \mathbb{R}^M} J(\tilde{z}(t_i)) \quad \text{subject to} \quad lb \leq c(\tilde{z}(t_i)) \leq ub$$

$B_{i,k_j} =$ basis functions
$C_i^j =$ coefficients
$z_i =$ flat outputs
Application: Caltech Ducted Fan

Flight Dynamics

\[ m \ddot{x} = -D \cos \gamma - L \sin \gamma + F_{Xb} \cos \theta + F_{Zb} \sin \theta \]
\[ m \ddot{z} = D \sin \gamma - L \cos \gamma - m g_{eff} + F_{Xb} \sin \theta + F_{Zb} \cos \theta \]
\[ J \ddot{\theta} = M_a - \frac{1}{r_s} I \Omega \dot{x} \cos \theta + M_T \]

\[ \alpha = \theta - \gamma, \quad \text{angle of attack} \]
\[ \gamma = \tan^{-1} \frac{\dot{z}}{x}, \quad \text{flight path angle} \]

Trajectory Generation Implementation

- System is approximately flat, even with aerodynamic forces
- More efficient to over-parameterize the outputs; use \( z = (x, y, \theta) \)
- Input constraints: max thrust, flap limits, flap rates

\[ L = \frac{1}{2} \rho V^2 S C_L (\alpha) \]
\[ D = \frac{1}{2} \rho V^2 S C_D (\alpha) \]
\[ M_a = \frac{1}{2} c \rho V^2 S C_M (\alpha) \]
Implementation using NTG Software Library

Features
• Handles constraints
• Very fast (real-time), especially from warm start
• Good convergence

Weaknesses
• No convergence proofs
• Misses constraints between collocation points
• Doesn’t exploit mechanical structure (except through flatness)

http://www.cds.caltech.edu/~murray/software/2002a_ntg.html
Example 1: Trajectory Generation for the Ducted Fan

Caltech Ducted Fan
- Ducted fan engine with vectored thrust
- Airfoil to provide lift in forward flight mode
- Design to emulate longitudinal flight dynamics
- Control via dSpace-based real-time controller

Trajectory Generation Task: point to point motion avoiding obstacles
- Use differential flatness to represent trajectories satisfying dynamics
- Use B-splines to parameterize trajectories
- Solve \textit{constrained} optimization to avoid obstacles, satisfy thrust limits
NTG Convergence Properties

Numerical Studies using Caltech Ducted Fan

- 6461 test cases
- 500 initial guess for spline coefficients
- Total of > 3M runs

- Count # of cases that converge for given # of initial guesses

- Comparison between quasi-collocation ($x, y, \theta$) and full collocation (states and inputs)
Trajectory Generation for Non-Flat Systems

If system is not fully flat, can still apply NTG

\[ \mathcal{S} = f(x, u) \]

\[ z = z(x, u, \mathcal{S}, u^{(q)}) \]

\[ x = x(z, \mathcal{S}, z^{(q)}) \]

\[ u = u(z, \mathcal{S}, z^{(q)}) \]

\[ y = h(x, u) \]

\[ (x, u) = \Gamma(y, \mathcal{S}, y^{(q)}) \]

\[ 0 = \Phi(y, \mathcal{S}, y^{(p)}) \]

When system is not flat, use quasi-collocation

- Choose output \( y = h(x, u) \) that can be used to compute the full state and input
- Remaining dynamics are treated as constraints for trajectory generation
- Example: chain of integrators

\[ \mathcal{S}_1 = x_2 \]
\[ \mathcal{S}_2 = u \]
\[ y_1 = x_1 \]
\[ y_2 = x_2 \]
\[ x_1 = y_1 \]
\[ x_2 = y_2 \]
\[ u = \mathcal{S}_2 \]

\[ \{ \text{Solve using NTG with } lb = ub \} \]

Can also do full collocation (treat all dynamics as constraints)

\[ (x, u) = \sum \alpha_i \psi_i(t) \]
\[ \mathcal{S}(t_i) = f(x(t_i), u(t_i)) \]

Each equation gives constraints at collocation points \( \Rightarrow \) highly constrained optimization
Effect of Defect on Computation Time

Defect as a measure of flatness
- Defect = number of remaining differential equations
- Defect 0 $\Rightarrow$ differentially flat

Sample problem: 5 states, 1 input
- $x_1$ is possible flat output
- Can choose other outputs to get systems with nonzero defect
- 200 runs per case, with random initial guess

Computation time related to defect through power law
- SQP scales cubically $\Rightarrow$ minimize the number of free variables

\[
\begin{align*}
\dot{x}_1 &= 5x_2 \\
\dot{x}_2 &= \sin x_1 + x_2^2 + 5x_3 \\
\dot{x}_3 &= -x_1x_2 + x_3 + 5x_4 \\
\dot{x}_4 &= x_1x_2x_3 + x_2x_3 + x_4 + 5x_5 \\
\dot{x}_5 &= -x_5 + u
\end{align*}
\]

Dramatic speedup through reduction of differential constraints

Petit, Milam, Murray
NOLCOS, 2001
Example 2: Satellite Formation Control

Goal: reconfigure cluster of satellites using minimum fuel

Reconfiguration  Stationkeeping  Deconfiguration

Dynamics given by Hill’s equations (fully actuated $\Rightarrow$ flat)

\[
\begin{align*}
\ddot{q}_1 &= \frac{\mu q_1}{|q|^3} - \frac{3J_2 \mu R_e^2 q_1 \left(q_1^2 + q_2^2 - 4q_3^2\right)}{2|q|^7} + u_1^I \\
\ddot{q}_2 &= \frac{\mu q_2}{|q|^3} - \frac{3J_2 \mu R_e^2 q_2 \left(q_1^2 + q_2^2 - 4q_3^2\right)}{2|q|^7} + u_2^I \\
\ddot{q}_3 &= \frac{\mu q_3}{|q|^3} - \frac{3J_2 \mu R_e^2 q_3 \left(3q_1^2 + 3q_2^2 - 2q_3^2\right)}{2|q|^7} + u_3^I
\end{align*}
\]
Satellite Formation Results

Station-keeping optimization

- Maintain a given area between the satellites (for good imaging) while minimizing the amount of fuel
- Idea: exploit natural dynamics of orbital equations as much as possible
- Input constraints: $\Delta V < 20 \text{ m/s/year}$

Results

- Use NTG to optimize over 60 orbits (~3 days), then repeat
- Results: at $45^\circ$ inclination, obtain $10.4 \text{ m/s/year}$

<table>
<thead>
<tr>
<th>$i$ (deg)</th>
<th>$S$ (m$^2$)</th>
<th>$\Delta V$ (m/s/year)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0$</td>
<td>$100$</td>
<td>$25.6$</td>
</tr>
<tr>
<td>$45$</td>
<td>$100$</td>
<td>$47.8$</td>
</tr>
<tr>
<td>$90$</td>
<td>$100$</td>
<td>$21.4$</td>
</tr>
</tbody>
</table>

Projected area of satellites

100 m$^2$
Example 3: MVWT Control Design

Control design technique

1. LQR design of state space controller $K$ around reference velocity
2. Choose $P$, $Q$, $R$ using Kalman’s formula
3. Implement as a receding horizon control with input and state space constraints

- RHC controller respects state space constraint

\[
\begin{align*}
m \ddot{\xi} &= -\eta \ddot{\xi} + (F_s + F_p) \cos \theta \\
m \ddot{\eta} &= -\eta \ddot{\eta} + (F_s + F_p) \sin \theta \\
J \ddot{\theta} &= -\psi \ddot{\theta} + (F_s - F_p) r_j
\end{align*}
\]
Summary: Real-Time Trajectory Generation

Flatness is a key property for efficient motion planning
- Allows conversion of dynamics into algebra ⇒ much faster algorithms

NTG software package implements required calculations
- Allows solution of general constrained optimization, w/ parameterized outputs
- Gives approximate results ⇒ need to use in feedback context (not open loop)

Growing collection of applications
- Caltech ducted fan, satellite formation control
- Underwater vehicles, wheeled mobile robots, RoboFlag, Alice, …

\[
\min J = \int_{t_0}^{T} q(x, u) \, dt + V(x(T), u(T)) \\
\dot{x} = f(x, u) \quad lb \leq g(x, u) \leq ub
\]
Homework and Project Ideas

Homework
• Download NTG and implement the point to point motion control problem for Alice or a RoboFlag vehicle.

Project ideas:
• For multi-vehicle applications, need to distribute the computation across multiple computers
• Use spread to implement a distributed trajectory generation capability