NPI: Real-Time Adaptive Entry Guidance

Final Presentation – ESTEC – 08.04.2016

David Seelbinder  david.seelbinder@dlr.de
Prof. Dr. Christof Büskens  bueskens@math.uni-bremen.de
Dr. Stephan Theil  stephan.theil@dlr.de
Agenda

- **Motivation and introduction**
- **Trajectory computation**
  - Parametric sensitivity analysis of nonlinear programs (offline)
  - Fast solution approximation (online)
  - Repeated online trajectory computation
  - Region of convergence
- **Trajectory tracking**
  - Feedback linearization
  - Drag energy dynamics
- **Guidance system overview**
- **Monte Carlo campaign**
  - Perturbed environment
  - Guidance performance results
- **Processor-in-the-loop test**
- **Software tools**
- **Summary out outlook**
Guided Entry Phase

Schiaparelli enters atmosphere
- Time: 0 sec
- Altitude: 123 km
- Speed: 21,000 km/h

Heatshield protection during atmospheric deceleration
- Time of maximum heating: 1 min 12 sec
- Altitude: 45 km
- Speed: 19,000 km/h

Parachute deploys
- Time: 3 min 31 sec
- Altitude: 11 km
- Speed: 1,700 km/h

Front shield separates, radar turns on
- Time: 4 min 3 sec
- Altitude: 7 km
- Speed: 320 km/h

Parachute jettisoned with rear cover
- Time: 5 min 22 sec
- Altitude: 1.2 km
- Speed: 140 km/h

Thruster ignition
- Time: 5 min 23 sec
- Altitude: 1.1 km
- Speed: 290 km/h

Thrusters off; freefall
- Time: 5 min 52 sec
- Altitude: 2 m
- Speed: 4 km/h

Touchdown
- Time: 5 min 53 sec
- Altitude: 0 m
- Speed: 10 km/h

Image credit: ESA
Guided Entry Phase

- **Schiaparelli enters atmosphere**
  - Time: 0 sec
  - Altitude: 135 km
  - Speed: 21,000 km/h

- **Heatshield protection during atmospheric deceleration**
  - Time of maximum deceleration: 3 min 12 sec
  - Altitude: 45 km
  - Speed: 19,000 km/h

- **Parachute deploys**
  - Time: 3 min 28 sec
  - Altitude: 11 km
  - Speed: 1,700 km/h

- **Front shield separates, radar turns on**
  - Time: 4 min 2 sec
  - Altitude: 7 km
  - Speed: 320 km/h

- **Parachute jettisoned with rear cover**
  - Time: 5 min 22 sec
  - Altitude: 1.2 km
  - Speed: 140 km/h

- **Thrusters ignite**
  - Time: 5 min 23 sec
  - Altitude: 1.1 km
  - Speed: 290 km/h

- **Thusters off, freefall**
  - Time: 5 min 52 sec
  - Altitude: 2 m
  - Speed: 4 km/h

- **Touchdown**
  - Time: 5 min 53 sec
  - Altitude: 0 m
  - Speed: 10 km/h

*Image credit: ESA*
Guidance Task

Find an admissible control sequence that steers the vehicle on a feasible trajectory from the current state to the desired terminal state.

Problem: Nonlinear dynamic system with state and control constraints

Req.: Planning of the control sequence and state trajectory over the entire future path

Req.: Mathematical model

Challenges:
1. Constrained nonconvex optimization problem
2. High computational power demand
3. Model dependency
Approaches to Entry Guidance

Atmospheric Entry Guidance

Trajectory Tracking
- Multi-Variable Tracking
- Drag Tracking
- Predictive Drag Tracking
  - Geometric Drag Profile Shaping

Optimal Control
- MPC
- Constr. Numeric Predictor Corrector

Trajectory Computation
- Numeric Predictor Corrector
Approaches to Entry Guidance
Optimization based Guidance

Global Optimization

Local NLP Inf. Horizon

Approx. solution to non-convex problem

Initial guess

approx. solution

approx. problem

Convex opt. problem
Optimization based Guidance

Global Optimization

Initial guess

Local NLP Inf. Horizon

Approx. solution

Convex opt. problem

Linear model + finite horizon

control param. / geometric methods

Reduced Frequency

Approx. solution to non-convex problem

Finite horizon

Linearization of opt. conditions

NMPC

PSA Tracking

Slow Local NLP Inf. Horizon

Tracking

LMPC

Quadratic Programming

QCQP

LCQP

UCQP

Tracking
Optimization based Guidance

Chart 10  •  David Seelbinder  •  ESTEC • April 2016

- Global Optimization
  - Initial guess
  - Local NLP Inf. Horizon
    - Approx. solution
    - Convex opt. problem
      - Linear model + finite horizon
      - control param. / geometric methods
      - Quadratic Programming
        - QCQP
        - LCQP
        - UCQP
        - Tracking
  - Slow Local NLP Inf. Horizon
    - Reduced Frequency
    - Tracking
  - NMPC
    - Finite horizon
    - Linearization of opt. conditions
  - PSA
    - Tracking

Approx. solution to non-convex problem
## Optimization based Guidance

<table>
<thead>
<tr>
<th>Property</th>
<th>NMPC Inf. Horizon</th>
<th>NMPC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Computational power required</strong></td>
<td>--</td>
<td>-</td>
</tr>
<tr>
<td><strong>OBS complexity</strong></td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Respect path constraint</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>Region of attraction</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>Optimality</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>Method universality</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>Online optimization</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>
## Optimization based Guidance

<table>
<thead>
<tr>
<th>Property</th>
<th>NMPC Inf. Horizon</th>
<th>NMPC</th>
<th>Slow NLP / Tracking</th>
<th>PSA / Tracking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computational power required</td>
<td>--</td>
<td>-</td>
<td>o/+</td>
<td>++</td>
</tr>
<tr>
<td>OBS complexity</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>+</td>
</tr>
<tr>
<td>Respect path constraint</td>
<td>++</td>
<td>++</td>
<td>+/-</td>
<td>o/-</td>
</tr>
<tr>
<td>Region of attraction</td>
<td>++</td>
<td>+</td>
<td>+/-</td>
<td>?/?</td>
</tr>
<tr>
<td>Optimality</td>
<td>++</td>
<td>+</td>
<td>+/-</td>
<td>+/?</td>
</tr>
<tr>
<td>Method universality</td>
<td>++</td>
<td>++</td>
<td>+/o</td>
<td>o/o</td>
</tr>
<tr>
<td>Online optimization</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>
Optimization based Guidance

<table>
<thead>
<tr>
<th>Property</th>
<th>NMPC Inf. Horizon</th>
<th>NMPC</th>
<th>Slow NLP / Tracking</th>
<th>PSA / Tracking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computational power required</td>
<td>--</td>
<td>-</td>
<td>o/+</td>
<td>+/+</td>
</tr>
<tr>
<td>OBS complexity</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>+</td>
</tr>
<tr>
<td>Respect path constraint</td>
<td>++</td>
<td>++</td>
<td>+/-</td>
<td>o/-</td>
</tr>
<tr>
<td>Region of attraction</td>
<td>++</td>
<td>+</td>
<td>+/?</td>
<td>?/?</td>
</tr>
<tr>
<td>Optimality</td>
<td>++</td>
<td>+</td>
<td>+/?</td>
<td>+/?</td>
</tr>
<tr>
<td>Method universality</td>
<td>++</td>
<td>++</td>
<td>+/-</td>
<td>o/o</td>
</tr>
<tr>
<td>Online optimization</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>
Approaches to Entry Guidance

Atmospheric Entry Guidance

Trajectory Tracking
- Multi-Variable Tracking
- Drag Tracking
- Predictive Drag Tracking
  - Geometric Drag Profile Shaping

Trajectory Computation
- Optimal Control
- Numeric Predictor Corrector
  - Constr. Numeric Predictor Corrector
  - MPC

...
Approaches to Entry Guidance

- Optimal solution approximation based on parametric sensitivity analysis to obtain a fast update of the optimal control sequence
- Combination with drag tracking
Trajectory computation

- Parametric sensitivity analysis of nonlinear programs (offline)
- Fast solution approximation (online)
- Repeated online trajectory computation
- Region of attraction
Offline Phase: Problem Formulation and Transcription

Formulation as parametric Optimal Control Problem:

\[ \min J(x, u, p) = g(x(t_0), x(t_f), p) + \int_{t_0}^{t_f} l(x(t), u(t), p) \, dt \]

w.r.t.

\[ \dot{x}(t) = f(x(t), u(t), p) \]

\[ \psi_0(x(t_0), p) = 0 \]

\[ \psi_f(x(t_f), p) = 0 \]

\[ C(x(t), u(t), p) \leq 0 \quad t \in [t_0, t_f] \]
Offline Phase: Problem Formulation and Transcription

- Direct optimization
  - Discretization (Runge-Kutta-4, Linear control interpolation)
  - Transcription into a parametric nonlinear program (using single or multiple shooting)

- Choice of a nominal parameter set $p_0 = 0$

\[
\min_z \quad F(z, p_0) \\
\text{w.r.t.} \quad g_l \leq G(z, p_0) \leq g_u
\]
Offline Phase: Sensitivity Analysis I

Obtain nominal optimal solution $z_0 = z^*(p_0)$
Offline Phase: Sensitivity Analysis I

- Obtain nominal optimal solution $z_0 = z^*(p_0)$
- Obtain parametric sensitivity $\frac{dz}{dp}[p_0]$
Linearization of the Necessary Optimality Conditions

Active constraints: \( G^a(z, p) \)

Lagrange multipliers: \( \eta^a \)

Lagrange function: \( L(z, \eta^a, p) = F(z, p) + (\eta^a)^T G^a(z, p) \)
Linearization of the Necessary Optimality Conditions

Active constraints: \( G^a(z,p) \)

Lagrange multipliers: \( \eta^a \)

Lagrange function: \( L(z, \eta^a, p) = F(z, p) + (\eta^a)^T G^a(z, p) \)

Necessary optimality conditions (KKT):

\[
K(z(p), \eta^a(p), p) = \begin{bmatrix}
    \nabla_z L(z, \eta^a, p) \\
    G^a(z, p)
\end{bmatrix} = \begin{bmatrix}
    \nabla_z F(z, p) + (\eta^a)^T \nabla_z G^a(z, p) \\
    G^a(z, p)
\end{bmatrix} = 0
\]
Linearization of the Necessary Optimality Conditions

Active constraints: \( G^a(z, p) \)

Lagrange multipliers: \( \eta^a \)

Lagrange function: \( L(z, \eta^a, p) = F(z, p) + (\eta^a)^T G^a(z, p) \)

Necessary optimality conditions (KKT):

\[
K(z(p), \eta^a(p), p) = \begin{bmatrix} \nabla_z L(z, \eta^a, p) \\ G^a(z, p) \end{bmatrix} = \begin{bmatrix} \nabla_z F(z, p) + (\eta^a)^T \nabla_z G^a(z, p) \\ G^a(z, p) \end{bmatrix} = 0
\]

Differentiation of \( K(z(p), \eta^a(p), p) \equiv 0 \) with respect to \( p \) at point \( p_0 \):

\[
\begin{bmatrix} \nabla_z^2 L(z_0, \eta^a_0, p_0) \\ \nabla_z G^a(z_0, p_0) \end{bmatrix} \begin{bmatrix} \frac{dz}{dp}[p_0] \\ \frac{d\eta^a}{dp}[p_0] \end{bmatrix} + \begin{bmatrix} \nabla_p^2 L(z_0, \eta^a_0, p_0) \\ \nabla_p G^a(z_0, p_0) \end{bmatrix} = 0
\]

Literature: [Fiacco] [Büskens]
Offline Phase: Sensitivity Analysis I

- Obtain nominal optimal solution \( z_0 = z^*(p_0) \)
- Obtain parametric sensitivity \( \frac{dz}{dp}[p_0] \)
Offline Phase: Sensitivity Analysis I

- Obtain nominal optimal solution $z_0 = z^*(p_0)$
- Obtain parametric sensitivity $\frac{dz}{dp}[p_0]$
- If $z_0$ fulfills strong second order sufficient conditions: Kuhn-Tucker matrix is invertible

\[
\begin{bmatrix}
\frac{dz}{dp}[p_0] \\
\frac{d\eta^a}{dp}[p_0]
\end{bmatrix}
= - \begin{bmatrix}
\nabla_z^2 L(z_0, \eta^a_0, p_0) & \nabla_z G^a(z_0, p_0)^T \\
\nabla_z G^a(z_0, p_0) & 0
\end{bmatrix}^{-1}
\begin{bmatrix}
\nabla_{zp^2} L(z_0, \eta^a_0, p_0) \\
\nabla_p G^a(z_0, p_0)
\end{bmatrix}
\]
Online Solution Approximation: p-Step

- Approximation of optimal solution for disturbed parameters $p$ using Taylor expansion

$$z^*(p) \approx z_1 := z_0 + \frac{dz}{dp}[p_0] \cdot (p - p_0)$$
Online Solution Approximation: p-Step

- Approximation of optimal solution for disturbed parameters $p$ using Taylor expansion

$$z^*(p) \approx z_1 := z_0 + \frac{dz}{dp}[p_0] \cdot (p - p_0)$$

error in the active constraints

$$\|G^a(z_1, p)\| > 0$$
Offline Phase: Sensitivity Analysis II

\[
\begin{align*}
\text{min}_z & \quad F(z, p_0) \\
\text{w.r.t.} & \quad g_l \leq G(z, p_0) \quad \leq g_u
\end{align*}
\]
Offline Phase: Sensitivity Analysis II

- Additional parameter vector $q$ with nominal value $q_0 = 0$

\[
\min_z F(z, p_0) \\
\text{w.r.t.} \quad g_l \leq G(z, p_0) - q_0 \leq g_u
\]

- \( \frac{dz}{dq} \) can be computed analog to \( \frac{dz}{dp} \)
Online Solution Approximation: \( p \text{-Step and } q \text{-Step} \)

- Approximation of optimal solution for disturbed parameters \( p \) using Taylor expansion

\[
z^*(p) \approx z_1 := z_0 + \frac{dz}{dp}[p_0] \cdot (p - p_0)
\]

error in the active constraints

\[
\|G^a(z_1, p)\| > 0
\]
Online Solution Approximation: p-Step and q-Step

- Approximation of optimal solution for disturbed parameters $p$ using Taylor expansion

$$z^*(p) \approx z_1 := z_0 + \frac{dz}{dp} \cdot [p_0] \cdot (p - p_0)$$

- $\frac{dz}{dq}$ is used to iteratively correct the constraint error and at the same time improve the optimality of the approximation

$$\text{while } \|G^a(z, p)\| > \varepsilon$$

$$q_i = G^a(z_i, p)$$

For $\|q - q_0\| < \delta$ iteration converges against a fixpoint $z_\infty$ at which

$$\|G^a(z_\infty, p)\| = 0$$

[Büskens]
Example

- Scenario: Entry of a small capsule into the Martian atmosphere
- Control: Bank angle $\mu$
- Assumptions: Constant AoA (capsule statically and dynamically stable)
- Dynamic model: Translational motion over a spherical, rotating planet
  (nonlinear, coupled, first order ODE system)
Example

- Scenario: Entry of a small capsule into the Martian atmosphere
- Control: Bank angle $\mu$
- Assumptions: Constant AoA (capsule statically and dynamically stable)
- Dynamic model: Translational motion over a spherical, rotating planet (nonlinear, coupled, first order ODE system)

### State

<table>
<thead>
<tr>
<th>State</th>
<th>$x(t_0)$</th>
<th>$x(t_f)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Altitude</td>
<td>$h_0 = 120000 \text{ m}$</td>
<td>$h_f = 10000 \text{ m}$</td>
</tr>
<tr>
<td>Longitude</td>
<td>$\lambda_0 = 0^\circ$</td>
<td>$\lambda_f = 11.3^\circ$</td>
</tr>
<tr>
<td>Latitude</td>
<td>$\varphi_0 = 25^\circ$</td>
<td>$\varphi_f = 23.3^\circ$</td>
</tr>
<tr>
<td>Velocity</td>
<td>$v_0 = 5440.8 \text{ m/s}$</td>
<td>$v_f \leq 450 \text{ m/s}$</td>
</tr>
<tr>
<td>FPA</td>
<td>$\gamma_0 = -14.5^\circ$</td>
<td>$\gamma_f : \text{free}$</td>
</tr>
<tr>
<td>Heading</td>
<td>$\chi_0 = 97.4^\circ$</td>
<td>$\chi_f : \text{free}$</td>
</tr>
</tbody>
</table>

### Path Constraint

<table>
<thead>
<tr>
<th>Path Constraint</th>
<th>Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heat flux</td>
<td>$\dot{Q} \leq 1600 \text{ kW/m}^2$</td>
</tr>
<tr>
<td>Dynamic pressure</td>
<td>$q \leq 17 \text{ kPa}$</td>
</tr>
<tr>
<td>Load factor</td>
<td>$n \leq 15$</td>
</tr>
</tbody>
</table>
Example

**Parametrization**

Initial condition: \( \psi_0 = x(t_0) + p_I \)

Vehicle mass: \( m = m_0 + p_m \)

Lift coefficient: \( C_L = C_{L0} + p_L C_{L0} \)

Drag coefficient: \( C_D = C_{D0} + p_D C_{D0} \)

Parameter: \( p = (p_I, p_m, p_L, p_D)^T \)

Nominal value: \( p_0 = (0, 0, 0, 0, 0, 0, 0, 0)^T = 0 \)
Example

**Parametrization**

Initial condition: \( \psi_0 = x(t_0) + p_I \)

Vehicle mass: \( m = m_0 + p_m \)

Lift coefficient: \( C_L = C_{L0} + p_L C_{L0} \)

Drag coefficient: \( C_D = C_{D0} + p_D C_{D0} \)

Parameter: \( p = (p_I, p_m, p_L, p_D)^T \)

Nominal value: \( p_0 = (0, 0, 0, 0, 0, 0, 0, 0)^T = 0 \)

**Performance index**

- Minimize terminal velocity \( v_f \)
- Avoid flight at max/min lift
- “Smooth” control function

Objective function: \( F(x, u, p) = w_1 v_f^2 + w_2 \int_{t_0}^{t_f} w_e(t) \left[ w_3 (\cos \mu)^2 + w_4 \dot{\mu}^2 \right] dt \)

\( w_1, w_2, w_3, w_4 \): positive, constant

\( w_e(t) \): positive function
Nominal Solution ($p := p_0$)
Perturbed Environment

Example: Estimated conditions at flight time $t_0$:

Perturbed initial conditions:

<table>
<thead>
<tr>
<th>$\bar{x}_0(t_0)$</th>
<th>$\bar{p}_I$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\bar{h}_0 = h_0 + 2000$</td>
<td></td>
</tr>
<tr>
<td>$\bar{\lambda}_0 = \lambda_0 - 0.001$</td>
<td></td>
</tr>
<tr>
<td>$\bar{\varphi}_0 = \varphi_0 + 0.005$</td>
<td></td>
</tr>
<tr>
<td>$\bar{v}_0 = v_0 - 100$</td>
<td></td>
</tr>
<tr>
<td>$\bar{\gamma}_0 = \gamma_0 - 0.0012$</td>
<td></td>
</tr>
<tr>
<td>$\bar{\chi}_0 = \chi_0 - 0.0005$</td>
<td></td>
</tr>
</tbody>
</table>

Perturbed model:

$\bar{p}_m = 9$

$\bar{p}_L = 0.03$

$\bar{p}_D = -0.04$

Perturbed parameter set:

$\bar{p} = (\bar{p}_I, \bar{p}_m, \bar{p}_L, \bar{p}_D)^T$

Approximate optimal control sequence and trajectory for perturbed parameters $\bar{p}$
Solution Approximation for $p := \bar{p}$ (p-Step)
Solution Approximation for $p := \bar{p}$ (q-Step 1)
Solution Approximation for $p := \overline{p}$ (q-Step 2)
Solution Approximation for \( p := \bar{p} \) (q-Step 3)
Solution Approximation for $p := \bar{p}$ (q-Step 4) (Termination)
Parametric Sensitivities on Discrete Points of the Nominal Trajectory

Image 1: Sensitivity of $\mu$ against perturbations in $h$ at $\psi_0^0 = x^*(t_0)$
Parametric Sensitivities on Discrete Points of the Nominal Trajectory

Image 1: Sensitivity of $\mu$ against perturbations in $h$ at $\psi_0^0 = x^*(t_0)$

Image 2: Sensitivity of $\mu$ against perturbations in $h$ at $\psi_0^{t_i} = x^*(t_i)$, $0 \leq i \leq l$
Offline Phase: Sensitivity Catalog

Trajectory computation at a fixed time $t$ requires sensitivity differentials for initial condition $\psi_0^t := x^*(t), t \in [t_0, t_f)$.
Trajectory computation at a fixed time $t$ requires sensitivity differentials for initial condition $\psi_0^t := x^*(t)$, $t \in [t_0, t_f)$.

Sensitivity analysis is repeated at discrete points $t_i \in [t_0, t_f)$, $0 < i \leq l$ of the nominal trajectory $x^*(t)$.
Interpolation of Parametric Sensitivity Differentials

- Interpolate parametric sensitivities between sufficiently close initial conditions (required for all combinations of states/controls and perturbation parameters)
Interpolation of Parametric Sensitivity Differentials

Interpolate parametric sensitivities between sufficiently close initial conditions (required for all combinations of states/controls and perturbation parameters)
Procedure: Repeated Online Trajectory Computation

1. Estimate current state $\bar{x}$ and model perturbations $\bar{p}_m, \bar{p}_L, \bar{p}_D$
Procedure: Repeated Online Trajectory Computation

1. Estimate current state $\bar{x}$ and model perturbations $\bar{p}_m, \bar{p}_L, \bar{p}_D$

2. Determine $\bar{t}$ such that $\|\bar{p}_I\|$ is sufficiently small with $\bar{p}_I = \bar{x} - x^*(\bar{t}, p_0)$
Procedure: Repeated Online Trajectory Computation

1. Estimate current state $\bar{x}$ and model perturbations $\bar{p}_m, \bar{p}_L, \bar{p}_D$

2. Determine $\bar{t}$ such that $\|\bar{p}_I\|$ is sufficiently small with $\bar{p}_I = \bar{x} - x^*(\bar{t}, p_0)$

3. Set $p := (\bar{p}_I, \bar{p}_m, \bar{p}_L, \bar{p}_D)^T$
Procedure: Repeated Online Trajectory Computation

1. Estimate current state $\bar{x}$ and model perturbations $\bar{p}_m, \bar{p}_L, \bar{p}_D$

2. Determine $\bar{t}$ such that $||\bar{p}_I||$ is sufficiently small with $\bar{p}_I = \bar{x} - x^*(\bar{t}, p_0)$

3. Set $p := (\bar{p}_I, \bar{p}_m, \bar{p}_L, \bar{p}_D)^T$

4. Evaluate sensitivity surfaces at $\bar{t}$ to obtain parametric sensitivities approximations $\frac{\tilde{dz}}{dp} [\bar{t}, p_0], \frac{\tilde{dz}}{dq} [\bar{t}, q_0]$ (corresponding to initial condition $\psi_0^* := x^*(\bar{t}, p_0)$)
Procedure: Repeated Online Trajectory Computation

1. Estimate current state $\bar{x}$ and model perturbations $\tilde{p}_m, \tilde{p}_L, \tilde{p}_D$

2. Determine $\tilde{t}$ such that $||\tilde{p}_I||$ is sufficiently small with $\tilde{p}_I = \bar{x} - x^*(\tilde{t}, p_0)$

3. Set $p := (\tilde{p}_I, \tilde{p}_m, \tilde{p}_L, \tilde{p}_D)^T$

4. Evaluate sensitivity surfaces at $\tilde{t}$ to obtain parametric sensitivities approximations $\frac{d\tilde{z}}{dp}[\tilde{t}, p_0], \frac{d\tilde{z}}{dq}[\tilde{t}, q_0]$ (corresponding to initial condition $\psi_0^\tilde{t} := x^*(\tilde{t}, p_0)$)

5. Execute real-time iteration scheme with arguments
   a. $x^*(t, p_0), t \in [\tilde{t}, t_f]$ (remaining nominal trajectory)
   b. $\frac{d\tilde{z}}{dp}[\tilde{t}, p_0], \frac{d\tilde{z}}{dq}[\tilde{t}, q_0]$ (interpolated parametric sensitivities)
   c. $p := (\tilde{p}_I, \tilde{p}_m, \tilde{p}_L, \tilde{p}_D)^T$ (perturbation parameters)
Convergence Region

What are the maximal perturbations that can be compensated? 

Convergence region of the real-time iteration scheme?

- If the functions $F$ and $G$ are three times continuously differentiable and strong sufficient conditions of optimality hold:
  \[ \exists U(p_0) \text{ of } p_0 \text{ such that } \forall p = p_0 + \Delta p \in U \text{ it holds that } ||G^a(z_\infty, p)|| = 0 \]

- Extend of $U$: No analytic answer for nonlinear problems
- $U$ is defined for a fixed set of active constraints $G^a$
- If $G^a$ changes for a perturbation $p \Rightarrow p \notin U$
Convergence Region

What are the maximal perturbations that can be compensated?

Convergence region of the real-time iteration scheme?

If the functions $F$ and $G$ are three times continuously differentiable and strong sufficient conditions of optimality hold:

$\exists U(p_0)$ of $p_0$ such that $\forall p = p_0 + \Delta p \in U$ it holds that $\|G^a(z_\infty, p)\| = 0$

- Extend of $U$: No analytic answer for nonlinear problems
- $U$ is defined for a fixed set of active constraints $G^a$
- If $G^a$ changes for a perturbation $p \Rightarrow p \notin U$

Investigate convergence region!
(for selected perturbations)
Convergence Region: Single State Error (vs. Norm. Energy)
Convergence Region: Single State Error (vs. Time)
Convergence Region: Some Conclusions

- The convergence region $U(p_0)$ is large enough to cover expected errors at the entry interface point.

- The choice of the reference point $x^*(\bar{t}, p_0)$ is important! Metric $|.|$ desirable such that:
  
  $$\bar{p}_I = \min_{\bar{t} \in [t_0, t_f]} |\bar{x} - x^*(\bar{t}, p_0)| \Rightarrow \bar{p}_I \in U$$

- $U$ shrinks after peak deceleration such that successful trajectory computation cannot be guaranteed.
Convergence Region : Some Conclusions

- The convergence region $U(p_0)$ is large enough to cover expected errors at the entry interface point.

- The choice of the reference point $x^*(\bar{t}, p_0)$ is important! Metric $|.|$ desirable such that:
  $$\bar{p}_I = \min_{\bar{t} \in [t_0, t_f]} |\bar{x} - x^*(\bar{t}, p_0)| \Rightarrow \bar{p}_I \in U$$

- $U$ shrinks after peak deceleration such that successful trajectory computation cannot be guaranteed.

Additional guidance strategy required for “low” velocity flight.
Trajectory tracking

- Feedback linearization
- Drag-energy dynamics
Feedback Linearization 1

Feedback linearization can be applied to nonlinear systems of the form

\[ \dot{x} = f(x) + g(x)u \]

with state vector \( x \in \mathbb{R}^n \), control \( u \in \mathbb{R}^p \), output \( y \in \mathbb{R}^m \)

\[ y = h(x) \]

Goal: Input-Output linearization

Method: Transform \( \dot{x} \) into a new system whose states are the output \( y \in \mathbb{R}^m \) and its first \((n-1)\) ‘time’ derivatives.

\[
\begin{align*}
y &= h(x) \\
\dot{y} &= L_f h(x) \\
\ddot{y} &= L_f^2 h(x) \\
& \quad \vdots \\
y^{(n-1)} &= L_f^{n-1} h(x) \\
y^{(n)} &= L_f^n h(x) + L_g L_f^{n-1} h(x)u
\end{align*}
\]

Taking Lie derivatives of the output

\[
\begin{align*}
L_f h(x) &= \frac{d}{dx} h(x) f(x), \\
L_g h(x) &= \frac{d}{dx} h(x) g(x).
\end{align*}
\]
Feedback Linearization 2

Derivatives give the transformation $T$ and the transformed system $z$

\[
z = T(x) = \begin{bmatrix} z_1(x) \\ z_2(x) \\ \vdots \\ z_n(x) \end{bmatrix} = \begin{bmatrix} y \\ \dot{y} \\ \vdots \\ y^{(n-1)} \end{bmatrix} = \begin{bmatrix} h(x) \\ L_f h(x) \\ \vdots \\ L_f^{n-1} h(x) \end{bmatrix}
\]

\[
\begin{align*}
\dot{z}_1 &= L_f h(x) = z_2(x) \\
\dot{z}_2 &= L_f^2 h(x) = z_3(x) \\
&\quad \vdots \\
\dot{z}_n &= L_f^n h(x) + L_g L_f^{n-1} h(x) u
\end{align*}
\]

The feedback law

\[
u = \frac{1}{L_g L_f^{n-1} h(x)} (-L_f^n h(x) + v)
\]

- Creates a linear input-output map from $\nu$ to $z_1 = y$
- System $z$ is a cascade of $n$ integrators
- Outer-loop control $\nu$ can be chosen using linear system methods
Downrange Control

- Control
  - Lift \rightarrow \int \rightarrow \text{Vertical Velocity} \rightarrow \int \rightarrow \text{Altitude (Density)} \rightarrow \text{Drag} \rightarrow \int \rightarrow \text{Horizontal Velocity} \rightarrow \int \rightarrow \text{Range}

- Drag is the deciding factor!
  - Entry capsule cannot control drag directly
  - Control lift via bank angle rotation

Wingrove 1993, Survey of Atmospheric Re-Entry Guidance and Control Methods, AIAA
Downrange Control

- Entry capsule cannot control drag directly
- Control lift via bank angle rotation

Drag is the deciding factor!

\[ D[\alpha, r, V_{rel}] = \frac{1}{2} \cdot \rho[r] \cdot V_{rel}^2 \cdot C_D[\alpha, M[V, c[r]]] \cdot S \]

\( D \) drag
\( \rho \) atmospheric density
\( C_D \) drag coefficient
\( \alpha \) angle of attack
\( M \) Mach number
\( c \) speed of sound
\( S \) aerodynamic reference area

Chose drag as system output.

Wingrove 1993, Survey of Atmospheric Re-Entry Guidance and Control Methods, AIAA
Application of FBL to Entry Guidance

- Controller design in energy domain
- Separation of dynamics: Consider only longitudinal motion in v-r plane!

\[
\begin{align*}
    h' &= -\frac{V \sin \gamma}{D} \\
    v' &= \frac{D + g \sin \gamma}{Dv} \\
    \gamma' &= -\frac{L}{D v^2} \cos \mu + \left(\frac{g}{D v^2} - \frac{1}{h + r_p D}\right) \cos \gamma
\end{align*}
\]

Assumptions:
- Spherical, nonrotating planet
- No side slip
- Constant angle of attack

Independent variable:
\[
E = \frac{1}{2} v^2 - \left(\frac{GM}{r} - \frac{GM}{r_p}\right)
\]

Energy as independent variable causes a system order reduction compared to the time domain: A minimal energy domain representation needs only retain either $h$ or $v$!
Drag-Energy Derivatives

➢ Take output derivatives and apply feedback linearization mechanism

\[ D(r, V_{rel}) = \frac{1}{2} \cdot \rho(r) \cdot V_{rel}^2 \cdot C_D(M(V, c(r))) \cdot S \]

\[ D' = \frac{1}{2} S v \left( v \rho(h) C_d'(M) \frac{dM}{dEn} + C_d(M) \left( v \frac{d}{dEn} \rho'(h) + 2 \rho(h) \frac{d}{dEn} \right) \right) \]

\[ D'' = \frac{1}{2} S \left( v \rho(h) C_d''(M) \left( \frac{dM}{dEn} \right)^2 + C_d'(M) \left( v \rho(h) \frac{d^2 M}{dEn^2} + 2 \frac{dM}{dEn} \left( v \frac{d}{dEn} \rho'(h) + 2 \rho(h) \frac{d}{dEn} \right) \right) \right) + \]

\[ C_d(M) \left( v \left( \frac{d^2 h}{dEn^2} \rho'(h) + 2 \rho(h) \frac{d^2 v}{dEn^2} + v \left( \frac{d}{dEn} \right)^2 \rho''(h) \right) + 4 v \frac{d}{dEn} \frac{d}{dEn} \rho'(h) + 2 \rho(h) \left( \frac{d}{dEn} \right)^2 \right) \]

\[ D'' = a + bu \]
2nd Drag-Energy Derivative

\[ D'' = a(D, \dot{D}, E) + b(D, \dot{D}, E)u(D, \dot{D}, E) \]

\[
\begin{align*}
\alpha &= \frac{1}{4D^2 v \rho h (h + \rho M)^2} \left( 2D^2 v^5 \rho(h) (h + \rho M) c'(h)^2 \left( \frac{d\rho}{dE} \right)^2 Cd''(\frac{v}{c(h)}) + 4D^2 v^4 c(h) \rho(h) (h + \rho M) c'(h) \left( \frac{d\rho}{dE} \right) \left( \frac{d\rho}{dE} \right) Cd'(\frac{v}{c(h)}) - \frac{dv}{dE} Cd'(\frac{v}{c(h)}) \right) - \\
&\quad - 2v^3 c(h)^2 \left( Cd'(\frac{v}{c(h)}) \right)^2 D^2 v \rho(h) (h + \rho M) c'(h) \left( \frac{d\rho}{dE} \right)^2 c'(h) \rho(h) (h + \rho M) \left( 6D^2 \frac{d\rho}{dE} \frac{dv}{dE} + v \sin(y) \frac{dD}{dE} + D \cos^2(y) \left( g(h + \rho M) - v^2 \right) \right) + 2D^2 v (h + \rho M) \left( \frac{d\rho}{dE} \right)^2 \rho'(h) \right) - \\
&\quad - D^2 v \rho(h) (h + \rho M) \left( \frac{dv}{dE} \right)^2 \left( \frac{dv}{dE} \right) + \\
&\quad - \rho(h) (h + \rho M) \left( 2 \rho'(h) \left( \frac{d\rho}{dE} \right)^2 + v \sin(y) \frac{dD}{dE} + D \cos^2(y) \left( g(h + \rho M) - v^2 \right) \right) + 2D^2 v (h + \rho M) \left( \frac{d\rho}{dE} \right)^2 \rho''(h) \right) + \\
&\quad - \rho(h) (h + \rho M) \left( 4 \rho(h) \left( \frac{d\rho}{dE} \right)^2 - g \sin(y) \left( \frac{dD}{dE} + D \frac{dv}{dE} \right) + g \cos^2(y) \left( v^2 - g(h + \rho M) \right) \right) \\
\beta &= \frac{L}{D} \cos \beta \quad \text{State dependent control transformation!}
\end{align*}
\]

\[
\begin{align*}
b &= \frac{SuT \cos(y)}{2v c(h)^2} \left( v^3 \rho(h) c'(h) \left( \frac{v}{c(h)} \right) + g v c(h) \rho(h) \frac{dv}{dE} \left( \frac{v}{c(h)} \right) + c(h)^2 \left( 2g \rho(h) - v^2 \rho'(h) \right) \right)^2 \frac{D^2}{D} \cos \beta
\end{align*}
\]
Drag Tracking Law

Bijective transformation $T_E: [r, v, y] \rightarrow [D, \dot{D}, E]$ \hspace{1cm} (Drag – Energy Space)

System order: 2
Output relative degree: 2 \hspace{1cm} \rightarrow \text{No internal dynamics!}

Linearized plant should have error dynamics of 2nd order linear system.

\[
u = \frac{1}{b} (-a + \ddot{D}_{ref} - \omega^2 \Delta D - 2\xi \omega \Delta \dot{D})\]

Obtain control law:
Two-Degree of Freedom Guidance System
Guidance System Overview

- Two-degree-of-freedom design
  - Fast inner tracking loop (20 Hz)
  - Slow outer trajectory loop (0.05 Hz)

- Trajectory computation outputs
  • near optimal discrete $u^*, x^*$ for the entire remaining process
  • optimal bank angle profile $\mu_{ref}$ and drag profile $D_{ref}$ obtained from $u^*, x^*$

- Drag tracking controller based on Mease et. al.
PSA + Drag Tracking

Drag Tracking

✓ Strength
  • Fast feedback based directly on physical measurement
  • Robust against atmospheric disturbances
  • Performant during high-drag flight

○ Weakness
  • No planning / prediction
  • Based on separated and simplified dynamics
  • Over-sensitive during low-drag flight
PSA + Drag Tracking

Drag Tracking

✓ Strength
  • Fast feedback based directly on physical measurement
  • Robust against atmospheric disturbances
  • Performant during high-drag flight

○ Weakness
  • No planning / prediction
  • Based on separated and simplified dynamics
  • Over-sensitive during low-drag flight

PSA Update

○ Weakness
  • Strongly reliant on dynamic model and perturbation model
  • Not applicable after peak deceleration

✓ Strength
  • Near-optimal solution for states and control over entire flight path
  • Large initial correction space
  • Unified solution for DR and CR
PSA + Drag Tracking

Drag Tracking

- **Strength**
  - Fast feedback based directly on physical measurement
  - Robust against atmospheric disturbances
  - Performant during high-drag flight

- **Weakness**
  - No planning / prediction
  - Based on separated and simplified dynamics
  - Over-sensitive during low-drag flight

PSA Update

- **Weakness**
  - Strongly reliant on dynamic model and perturbation model
  - Not applicable after peak deceleration

- **Strength**
  - Near-optimal solution for states and control over entire flight path
  - Large initial correction space
  - Unified solution for DR and CR

PSA update and drag tracking compliment each other well!
Monte Carlo Campaign
Monte Carlo Campaign

- Large EIP state errors (uniform error distribution)

<table>
<thead>
<tr>
<th>State</th>
<th>Pert.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_0$</td>
<td>+- 3 km</td>
</tr>
<tr>
<td>$\lambda_0$</td>
<td>+- 0.3265°</td>
</tr>
<tr>
<td>$\varphi_0$</td>
<td>+- 0.1632°</td>
</tr>
<tr>
<td>$v_0$</td>
<td>+- 200 m/s</td>
</tr>
<tr>
<td>$\gamma_0$</td>
<td>+- 1°</td>
</tr>
<tr>
<td>$\chi_0$</td>
<td>+- 1°</td>
</tr>
</tbody>
</table>

- Guidance input
  - true state, lift and drag falsified with white noise
  - Perturbation parameters are estimated using an extended Kalman filter
Monte Carlo Campaign: Perturbed Environment

- Atmosphere perturbations
  - Random temperature profile between warm and cold conditions
  - Random sinusoidal density perturbations of up to 50% amplitude

- Aerodynamic coefficients perturbed by up to 10%

<table>
<thead>
<tr>
<th>Param</th>
<th>Pert.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rho$</td>
<td>Temp. +/- 50%</td>
</tr>
<tr>
<td>$c_L$</td>
<td>+/- 10 %</td>
</tr>
<tr>
<td>$c_D$</td>
<td>+/- 10 %</td>
</tr>
<tr>
<td>wind</td>
<td>+/- 200 m/s</td>
</tr>
<tr>
<td>mass</td>
<td>+/- 20 kg</td>
</tr>
</tbody>
</table>

![Graphical representation of Rho and Wind Speed](image)
Monte Carlo Campaign: Results

(|μ| + 3σ) hori. dist.: **12.3 km**

(|μ| + 3σ) alt. error: **2.4 km**

(3.5 DoF, 2500 MC cases)

<table>
<thead>
<tr>
<th>Result</th>
<th>Mean (μ)</th>
<th>Std. Dev. (σ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eucl. dist.</td>
<td>4.1 km</td>
<td>2.6 km</td>
</tr>
<tr>
<td>Hori. dist.</td>
<td>4 km</td>
<td>2.7 km</td>
</tr>
<tr>
<td>DR error</td>
<td>- 2.6 km</td>
<td>3.8 km</td>
</tr>
<tr>
<td>CR error</td>
<td>- 0.4 km</td>
<td>1.1 km</td>
</tr>
<tr>
<td>Alt. error</td>
<td>- 0.4 km</td>
<td>0.7 km</td>
</tr>
<tr>
<td>Vel. error</td>
<td>4 m/s</td>
<td>6 m/s</td>
</tr>
</tbody>
</table>
Parachute Opening Zone
Parachute Opening Zone
LEON2 Processor-in-the-Loop

Knowledge for Tomorrow
Processor-in-the-loop

- Test on RASTA-101 with 80 MHz LEON2 processor
- GNC c-code from autocoding from Embedded Matlab
- TASTE toolset: Onboard SW interface definition, communication setup and target compilation
- Problem sizing
  - dense NLP formulation
  - grid length 70
  - ~25 MB sensitivity data
Processor-in-the-loop

- Test on RASTA-101 with 80 MHz LEON2 processor
- GNC c-code from autocoding from Embedded Matlab
- TASTE toolset: Onboard SW interface definition, communication setup and target compilation
- Problem sizing
  - dense NLP formulation
  - grid length 70
  - ~25 MB sensitivity data

Trajectory computation time: < 1 sec.
Developed Software Tools
Sensitivity Analysis Framework (SAF)

Software tools to enable or support:
- Transcription of the OCP(p) into a NLP(p)
- Solution of multiphase OCP(p) (WORHP, IPOPT)
- High precision derivative computation (ADOL-C)
- Analysis, visualization and processing of the optimal solution and the sensitivity differentials
- Analysis and test of the real-time iteration scheme

SAF components:
- OCP transcription layer (C++)
- Analysis toolbox (Matlab)
- Mars entry simulator (Embedded Matlab/Simulink)
- Mars guided entry GNC (Embedded Matlab, autocoding capable)
SAF Transcription Layer

- Supports formulation of multi-phase parametric optimal control problems
- Uses automatic derivative computation with ADOL-C
- Enables parametric sensitivity computation using WORHP
- Generic multiple shooting transcription
- Automatic grid adaption
- Automatic problem scaling
- Generic interface is adaptable to most NLP solvers (interfaces existing for WORHP and IPOPT)
Sensitivity Analysis Toolbox

- Modular, object oriented design
- No commercial software quality, but
- Fairly tested and documented R&D tools for expert users
- User manual and architectural design guide provided
Summary and Outlook
Summary

- Sensitivity analysis of discretized optimal control process
- Parametrization of dynamic model and initial conditions
- Repeated online trajectory computation
  - Parametric sensitivities + interpolation
  - Taylor expansion and iterative constraint correction
- Two degree of freedom guidance system
  - PSA real-time iteration
  - Drag tracking
- Promising results in 3.5 DoF Monte Carlo campaign
- Real-time capability proven by PIL test on LEON2 processor
Tackling the Difficult Problems

Non-convexity and path constraints
✓ Offline analysis and trajectory design
✓ Reference trajectories and sensitivity catalog
✓ Path constr. can be represented in drag domain

Computational load
✓ Cascaded loop structure schedules and staggers expensive tasks
✓ PSA: fast online adaption relies on precomputed optimal solution derivative
✓ Drag feedback law

Model dependency
✓ Disturbance estimation and online model adaption
✓ Drag guidance law based directly on physical measurement
Tackling the Difficult Problems

Non-convexity and path constraints
✓ Offline analysis and trajectory design
✓ Reference trajectories and sensitivity catalog
✓ Path constr. can be represented in drag domain

Computational load
✓ Cascaded loop structure schedules and staggers expensive tasks
✓ PSA: fast online adaption relies on precomputed optimal solution derivative
✓ Drag feedback law

Model dependency
✓ Disturbance estimation and online model adaption
✓ Drag guidance law based directly on physical measurement

What could be done next?
- Embedded NLP solver is critical enabling technology
- NLP solver as FPGA?
- Hybrid: Online optimization and sensitivity computation + PSA update
Thank you for your attention!

david.seelbinder@dlr.de

This research was conducted under No. 4000107257/12 NL/GLC/al of the ESA Networking and Partnering Initiative.