Preliminary Results of Adaptive-Reinforcement Learning Control for Morphing Aircraft

Monish D. Tandale, Jie Rong & John Valasek
Aerospace Engineering Department
Texas A&M University

AIAA Guidance, Navigation, and Control Conference and Exhibit
16 - 19 Aug 2004
Providence, Rhode Island
Overview

- Introduction to Aircraft Morphing
- Simplified Model of a Morphing Air Vehicle
- Reinforcement Learning Module
- Structured Adaptive Model Inversion Control
- Adaptive-Reinforcement Learning Control Architecture Functionality
- Numerical Example
- Conclusions
- Future Work
Big Picture Research Goals

- **WHEN** to reconfigure
- **HOW** to reconfigure
- **LEARNING** to reconfigure
Which Morphing?

- **Morphing for Mission Adaptation**
  - Large scale, relatively slow, in-flight shape change to enable a single vehicle to perform multiple diverse mission profiles

as opposed to:

- **Morphing for Control**
  - In-flight physical or virtual shape change to achieve multiple control objectives (maneuvering, flutter suppression, load alleviation, active separation control)

John Davidson, NASA Langley, AFRL Morphing Controls Workshop – Feb 2004

Tandale, Rong, & Valasek   AIAA-2004-5358-4
Reconfiguration Command Generation

Knowledge Base

System Performance Evaluation

Adaptive Controller

Control Information Distribution

Sensed Information Aggregation

Environment

Synthetic Jets for Virtual Shaping and Separation Control

MultiSensor MEMS Arrays for Flow Control Feedback
Adaptive-Reinforcement Learning Control (A-RLC)

Conceptual Control Architecture for Reconfigurable Aircraft

SAMI
Structured Adaptive Model Inversion (Traditional Control)
Flight controller to handle wide variation in dynamic properties due to shape change

RL
Reinforcement Learning (Intelligent Control)
Learn the morphing dynamics and the optimal shape at every flight condition in real-time
Morphing Air Vehicle
Morphing Vehicle - TiiMY

Shape
- Ellipsoidal shape with varying axis dimensions.
- Constant volume (V) during morphing
- 2 independent variables: $y$ and $z$, dependent dimension $x = \frac{6V}{\pi yz}$

Morphing Dynamics
- Smart material: carbon nano-tubes or shape memory alloy
- Morphing Dynamics: Simple Nonlinear Differential Equations
  - $y$-dimension: $\ddot{y} + 2y\dot{y} = V_y$
  - $z$-dimension: $\ddot{z} + 3z\dot{z} = V_z$
Shape Morphing Animation

TiiMY
Morphing Time Histories
Optimal Shapes at Various Flight Conditions

Optimality is defined by identifying a cost function.

\[ J = J (\text{Current shape, Flight condition}) \]

\[ J = J_y + J_z = (y - S_y(F))^2 + (z - S_z(F))^2 \]

\[ S_y = 3 + \cos\left(\frac{\pi}{2} F\right) \quad \text{and} \quad S_z = 2 + 2e^{-0.5F} \]
6-DOF Mathematical Model for Dynamic Behavior

- **Variables**
  \[ \mathbf{p}_c = [d_x\ d_y\ d_z]^T \quad \mathbf{v}_c = [u\ v\ w]^T \]
  \[ \mathbf{\sigma} = [\phi\ \theta\ \psi]^T \quad \mathbf{\omega} = [p\ q\ r]^T \]

- **Nonlinear 6–DOF Equations**
  - **Kinematic level:**
    \[ \dot{\mathbf{p}}_c = \mathbf{J}_i \mathbf{\nu}_c \quad ; \quad \dot{\mathbf{\sigma}} = \mathbf{J}_a \mathbf{\omega} \]
  - **Acceleration level:**
    \[ m \ddot{\mathbf{v}}_c + \mathbf{\ddot{\omega}} m \mathbf{v}_c = \mathbf{F} + \mathbf{F}_d \]
    \[ I \ddot{\mathbf{\omega}} + \dot{\mathbf{I}} \mathbf{\omega} + \mathbf{\ddot{\omega}} I \mathbf{\omega} = \mathbf{M} + \mathbf{M}_d \]
    - additional dynamics due to morphing

- **Drag Force**
  - Function of air density, square of velocity along axis, and projected area of the vehicle perpendicular to the axis

*Tandale, Rong, & Valasek*  
AIAA-2004-5358-12
Knowledge Based Control

- Candidates to develop inference mechanism
  - Rules-Based Expert System
    - Model the knowledge of human experts
    - Imitate the natural behaviour of birds

**Question:** How Many Control Theorists Does It Take To Change A Using Reinforcement Learning?

- Possible learning algorithms include Artificial Neural Networks (ANN), Explanation-Based Learning (EBL), and Reinforcement Learning (RL)

- Biologically inspired control process
  - Mimic the behaviour of birds
Reinforcement Learning - 1

- Unsupervised Learning

- Achieves a specific **goal** by learning from interactions with the environment.
  - Considers **state** information
  - Performs sequences of **actions**, observing the consequences
  - Attempts to maximize **rewards** over time
    - These specify what is to be achieved, *not how to achieve it*
  - Constructs a **state value function**
    - Learns an optimal control **policy**

- Memory is contained in the state value function
Reinforcement Learning - 2

- Learning is done repetitively, by subjecting to different scenarios

- Learning is cumulative and lifelong

- Formulations are generally based on Finite Markov Decision Processes (MDP)
  - 3 major candidate algorithms:
    - Dynamic programming
    - Monte Carlo methods
    - Temporal Difference Learning
Q-Learning

- Action value function $Q(s,a)$
  - How well agent performs action $a$ in state $s$ under policy $\pi$
- Q learning is an off-policy temporal difference method

Q-Learning()
Initialize $Q(s,a)$ arbitrarily
Repeat (for each episode):
  Initialize $s$
  Repeat (for each step of episode):
    Choose $a$ from $s$ using policy derived from $Q(s,a)$
    // (e.g., $\epsilon$-greedy policy)
    Take action $a$, observe $r, s'$
    $Q(s,a) \leftarrow Q(s,a) + \alpha [r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$
    $s \leftarrow s'$
  until $s$ is terminal
return $Q(s,a)$
Function Approximation
K Nearest Neighbor Policy Iteration

- The shape of the vehicle is on continuous domains
- Use K-nearest neighbors method to approximate the action-value function $Q(s,a)$
  - Collect a set of state-action pair samples
  - Compute optimal action-values of these samples using Q-learning
  - The action-value of a new state-action pair is the interpolation of those of its K nearest neighbors.
- The whole policy iteration process is called KNN Policy Iteration (KNNPI)

$$
\text{KNNPI}
$$
Collect Sample
Learn $Q_{sample}(s_i, a_i)$
For $Q(s_0, a_0)$, find $\{s_i\}$, the set of K nearest neighboring $s$ to $s_0$

$$
Q(s_0, a_0) = \sum_{n=1}^{K} \frac{Q_{sample}(s_i, a_i)}{\text{Distance}(s_i, a_i, s_0, a_0)}
$$
Structured Adaptive Model Inversion Control
Structured Model Reference
Adaptive Control
Akella, Schaub, Junkins (Texas A&M)

Dynamics
2\textsuperscript{nd} order differential equations

\begin{align*}
  \dot{x} &= v \\
  \dot{v} &= a = \frac{F}{m}
\end{align*}

Exact kinematic relationship between position and velocity

Acceleration level relationships between forces and system parameters
Structured Adaptive Model Inversion
Subbarao, Junkins (Texas A&M)

Features

- Dynamic inversion inner-loop, with an MRAC outer-loop to handle system uncertainties.
- Controls are solved for explicitly:
  \[
  \begin{align*}
  \text{System Model} & \quad \dot{x} = Af(x) + Bu \\
  \text{Reference Trajectory} & \quad x_r, \dot{x}_r \\
  \text{Control Law} & \quad u = B^{-1}(\dot{x}_r - Af(x) - \lambda e) \text{ so that the error dynamics becomes } \dot{e} = -\lambda e
  \end{align*}
  \]
- Undesirable dynamics are cancelled and replaced with user specified desired dynamics.
- Easily applicable to nonlinear systems.
- Error dynamics can be specified.
- Shown to be very effective for a wide variety of systems.
Structured Adaptive Model Inversion Features

Trajectory Tracking for Dynamic Systems

- **Plant:**
  - Nonlinear in states, affine in control, uncertain parameters appear linearly.

- **Control:**
  - Dynamic Inversion and Sliding Mode Control.
  - Dynamic Inversion requires knowledge of system parameters, which are inherently uncertain.

- **Adaptive Learning Parameters:**
  - Updated in real-time, and used for the Dynamic Inversion

- **Adaptation Mechanism:**
  - Driven by the error between the actual plant trajectory and the reference trajectory

- **Stability Analysis:**
  - Guarantees that the plant trajectory, asymptotically converges to the reference trajectory in the presence of Parametric Uncertainties, and initial condition errors.
Structured Model & Minimal Parameterization

Kinematic Level Model

\[
\dot{\mathbf{p}}_c = J_1 \mathbf{v}_c
\]
\[
\dot{\boldsymbol{\sigma}} = J_a \mathbf{\omega}
\]

Acceleration level Model

\[
m \ddot{\mathbf{v}}_c + \tilde{\omega} m \mathbf{v}_c = \mathbf{F} + \mathbf{F}_d
\]
\[
I \ddot{\mathbf{\omega}} + \dot{\mathbf{I}} \mathbf{\omega} + \tilde{\omega} \mathbf{I} \mathbf{\omega} = \mathbf{M} + \mathbf{M}_d
\]

Attitude Control

\[
\mathbf{I}_a^*(\boldsymbol{\sigma})\ddot{\mathbf{\sigma}} + \mathbf{C}_a^*(\boldsymbol{\sigma}, \dot{\mathbf{\sigma}})\dot{\mathbf{\sigma}} = \mathbf{P}_a^T(\boldsymbol{\sigma})\mathbf{M}
\]

Minimal Parametrization of the Inertia Matrix

\[
\begin{bmatrix}
I_{11} & I_{12} & I_{13} \\
I_{12} & I_{22} & I_{23} \\
I_{13} & I_{23} & I_{33}
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_3
\end{bmatrix}
= 
\begin{bmatrix}
a_1 & 0 & 0 & a_2 & a_3 & 0 \\
0 & a_2 & 0 & a_1 & 0 & a_3 \\
0 & 0 & a_3 & 0 & a_1 & a_2
\end{bmatrix}
\begin{bmatrix}
I_{11} \\
I_{22} \\
I_{33} \\
I_{12} \\
I_{13} \\
I_{23}
\end{bmatrix}
\]
Control Law & Update Law

Using Minimal Parametrization of the Inertia Matrix

\[ I^*_a(\sigma)\ddot{\sigma} + C^*_a(\sigma, \dot{\sigma})\dot{\sigma} = Y_a(\sigma, \dot{\sigma}, \ddot{\sigma})\dot{\theta} \quad \text{unknown parameters} \]

With the control law

\[ M = P_a^{-T}\left\{ Y_a(\sigma, \dot{\sigma}, \dot{\sigma}_r, \ddot{\sigma}_r)\dot{\theta} - C_{da}\dot{\varepsilon} - K_{da}\varepsilon \right\} \]

the closed loop dynamics take the form

\[ I^*_a\dddot{\sigma} + \{C_{da} + C^*_a(\sigma, \dot{\sigma})\}\dot{\sigma} + K_{da}\varepsilon = Y_a(\sigma, \dot{\sigma}, \ddot{\sigma})\ddot{\theta} \]

and, along the adaptive law

\[ \dot{\theta} = -\Gamma Y_a(\sigma, \dot{\sigma}, \dot{\sigma}_r, \ddot{\sigma}_r)^T\dot{\varepsilon} \]

guarantees asymptotic stability of the tracking errors.
Adaptive–Reinforcement Learning Control
A-RLC Architecture
Numerical Example
Numerical Example

- **Objective**
  - Demonstrate **optimal** shape morphing for **multiple** specified **flight conditions**

- **Method**
  - For every flight condition, **learn** **optimal policy** that commands voltage producing the optimal shape
  - **Minimize total cost** over the entire flight trajectory
  - Evaluate the learning performance after 200 learning episodes

**RL Module is Completely Ignorant of Optimality Relations and Morphing Control Functions:**
It Must Learn On Its Own, From Scratch
Episodic Learning

- Unsupervised learning episode
- Single pass through 100 meter flight path in 200 seconds
- Reference trajectory is generated arbitrarily
- The flight condition changes twice during each episode
- Shape change iteration after every 1 second

**Exploration-exploitation dilemma:**
- Explorative early, exploitative later
- \( \varepsilon \)-policy with decreasing \( \varepsilon \)

Limited training examples
- Only 6 discrete flight conditions:

2000 samples for KNNPI

\[ a = \arg \max_a Q(s, a) \]

\[ a = \text{rand}(a_i) \]
Learning Animation

Action-Value Functions for Flight Condition 3

Q_plot

Tandale, Rong, & Valasek    AIAA-2004-5358-30
Demo
Comparison of True Optimal Shape and Learned Shape

- y-dimension
  - True: \dash", Learned: solid
  - Time (sec): 0 to 200

- z-dimension
  - True: \dash", Learned: solid
  - Time (sec): 0 to 200

- \( \frac{dy}{dt} \)
  - Time (sec): 0 to 200

- \( \frac{dz}{dt} \)
  - Time (sec): 0 to 200
Morphing Control Voltages

![Graph showing morphing control voltages over time.](graph.png)
Trajectory Tracking Performance
Time Histories of the Linear States

- $d_x$ (m)
- $d_y$ (m)
- $d_z$ (m)

- $u$ (m/sec)
- $v$ (m/sec)
- $w$ (m/sec)
Time Histories of the Angular States

- $\phi$ (deg)
- $\theta$ (deg)
- $\psi$ (deg)
- $\rho$ (deg/sec)
- $\theta$ (deg)
- $q$ (deg/sec)
- $r$ (deg/sec)

**Graphs:**
- Actual Trajectory
- Reference

**Axes:**
- Time (sec)

Sources:
- Tandale, Rong, & Valasek
- AIAA-2004-5358-36
Time Histories of the Adaptive Parameters

![Graph showing time histories of adaptive parameters. The top graph plots the norm of the inertia vector, and the bottom graph plots mass over time. The graphs show estimated and true values, with the estimated values represented by a solid line and the true values by a dashed line.](image)
Trajectory Tracking Controls
Conclusions

- Shape Changes for Mission Morphing can be treated as piecewise constant parameter changes
  - SAMI is a favorable method for trajectory tracking control

- Reinforcement Learning successfully learns the optimal control policy using the $\epsilon$-greedy policy
  - Takes care of the exploration exploitation dilemma

- Morphing dynamics (stiffness, frequency, damping ratio) and cost function have a major effect on learning times and learning performance.
  - Slowest element in system is critical

- Flight condition transitions, morphing dynamics, learning performance, and adaptive control must be balanced to achieve synergy and therefore good performance.
  - Coordination and timing is everything
Future Research 1

- **Directly learn** voltage inputs required to achieve certain position states with time dependency.
  - Skips mathematical modeling and computer simulation steps
  - Avoids modeling and simulation errors
Future Research 2

- Modify the morphing dynamics to represent SMA actuators.
  - Hysteretic behavior

- Investigate control methodologies to handle faster shape changes
  - Linear Parameter Varying (LPV) control

- Modify the Reinforcement Learning module to incorporate continuous states and actions.
  - Function approximators such as Radial Basis Functions

- Modify the simulation to include a more advanced aircraft model
  - Wing-Body, Wing-Body-Empennage, etc.
Questions?