An Evolutionary Approach to Discovering Execution Mode Boundaries for Adaptive Controllers

Anthony J. Clark
Computer Science Department, Missouri State University, USA

Jared M. Moore
School of Computing and Information Systems, Grand Valley State University, USA

Byron DeVries, Betty H. C. Cheng, and Philip K. McKinley
Department of Computer Science and Engineering, Michigan State University, USA

Adaptability of Autonomous Robots

Internal Uncertainties

- degrading and complex (flexible) components
- changing objectives and control strategies

External Uncertainties

- dynamic environments
- significant damage

Adaptive Control

Model-based

- require a precise model
- perform parameter identification

Data-driven

- (or, model-free)
- input / output data
- “learns” how to adapt

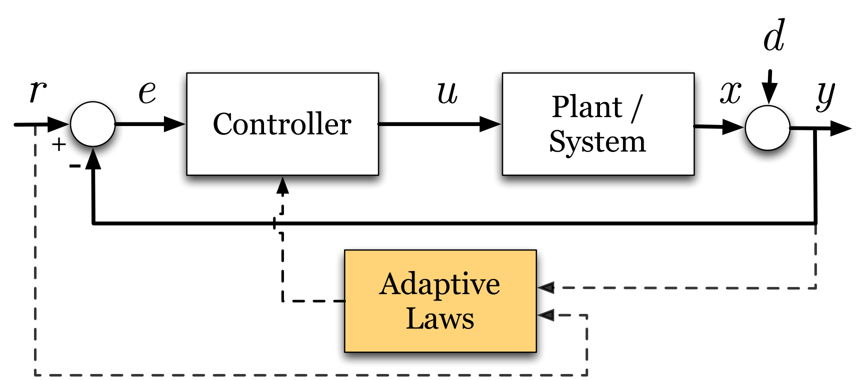
Adaptive Control

Model-based

- require a precise model
- perform parameter identification

Data-driven

- (or, model-free)
- input / output data
- "learns" how to adapt



Limitations of Adaptive Control

- Adaptive controllers can continue to adapt as long as the system remains fundamentally unchanged
- That is, the system responds to inputs in roughly the same manner even after it changes

- For example, cut the tail fin of a robotic fish

Robotic Fish

Applications

- autonomous
  mobile sensors
- biological studies
  (elicit natural behaviors)

Robotic Fish

Research Platform

- benefit from flexible components
- operate in a nonlinear environment
- exhibit complex dynamics
- [Marchese 2014]

Robotic Fish

Research Platform

- benefit from flexible components
- operate in a nonlinear environment
- exhibit complex dynamics

This Study

1. Improve adaptive controllers, AND
2. Find the limits of these adaptive controllers.

- Using evolutionary computation

- From controller's perspective:
  - Reference signals are part of the environment
  - Fin morphology is part of the environment

Enhancing Adaptive Control

Exploit EC to Enhance an MFAC [Cheng 2000]

- differential evolution [Storn 1997]
- evolve MFAC parameters
- controlling a robotic fish
- adapt to:
  - changing fin flexibilities
  - changing fin length
  - changing control demands

Adaptive Neural Network

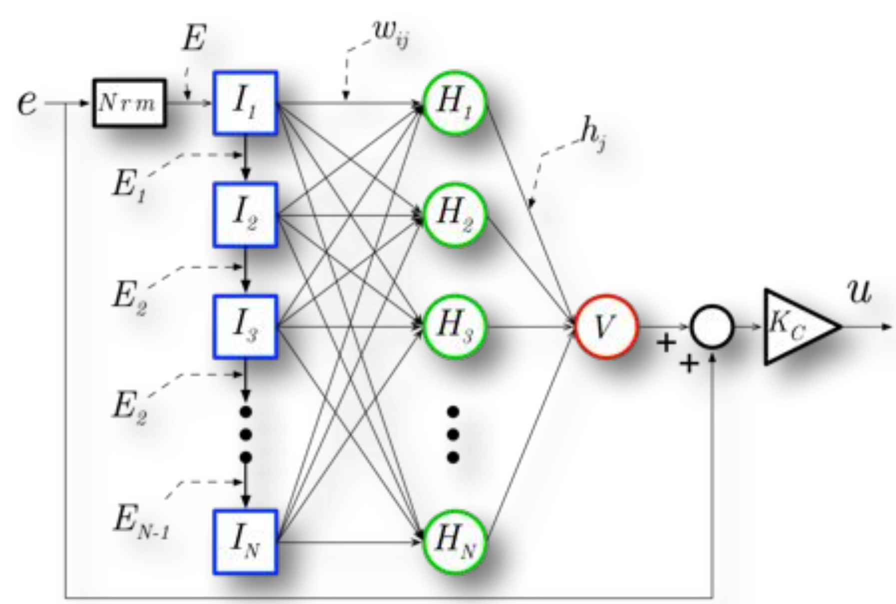
Network Activation

- feed-forward network
- propagated error
- sigmoid activation

Network Update

- minimize error

$$E_s(t) = \frac{1}{2}\, e(t)^2$$



Adaptive Neural Network

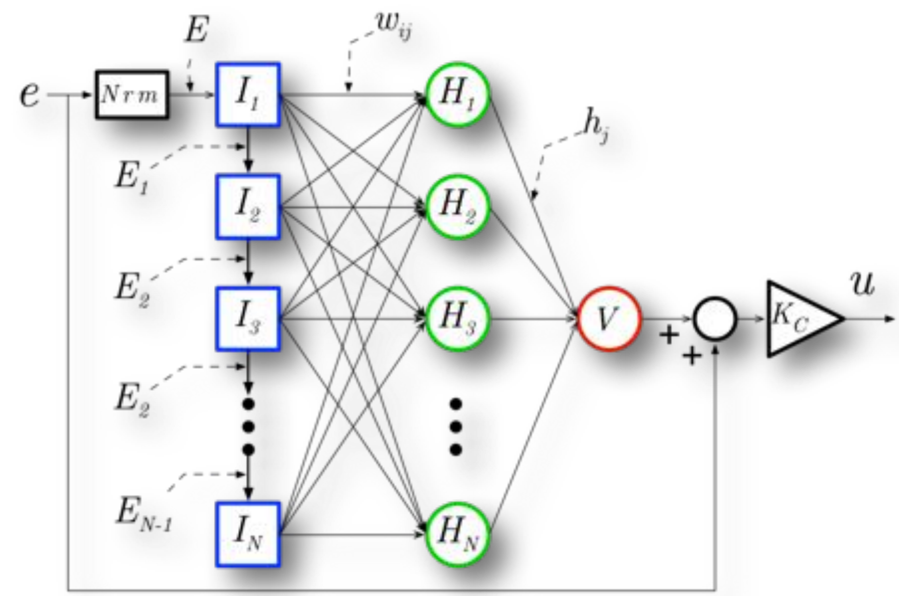
$$
\begin{aligned}
\Delta w_{ij}(n) &\propto \frac{\partial E_s}{\partial w_{ij}}, \\
&= \frac{\partial E_s}{\partial y}\frac{\partial y}{\partial w_{ij}}, \\
&= \frac{\partial E_s}{\partial y}\frac{\partial y}{\partial u}\frac{\partial u}{\partial w_{ij}}, \\
&= \frac{\partial E_s}{\partial y}\frac{\partial y}{\partial u}\frac{\partial u}{\partial o}\frac{\partial o}{\partial w_{ij}}, \\
&= \frac{\partial E_s}{\partial y}\frac{\partial y}{\partial u}\frac{\partial u}{\partial o}\frac{\partial o}{\partial q}\frac{\partial q}{\partial w_{ij}}, \\
&= \frac{\partial E_s}{\partial y}\frac{\partial y}{\partial u}\frac{\partial u}{\partial o}\frac{\partial o}{\partial q}\frac{\partial q}{\partial p}\frac{\partial p}{\partial w_{ij}}. \\
&= \boxed{-\eta\, K_c\, S_f(n)\, e(n)\, q_j(n)\, (1 - q_j(n))\, E_i(n) \sum_{k=1}^{N} h_k(n),}
\end{aligned}
$$

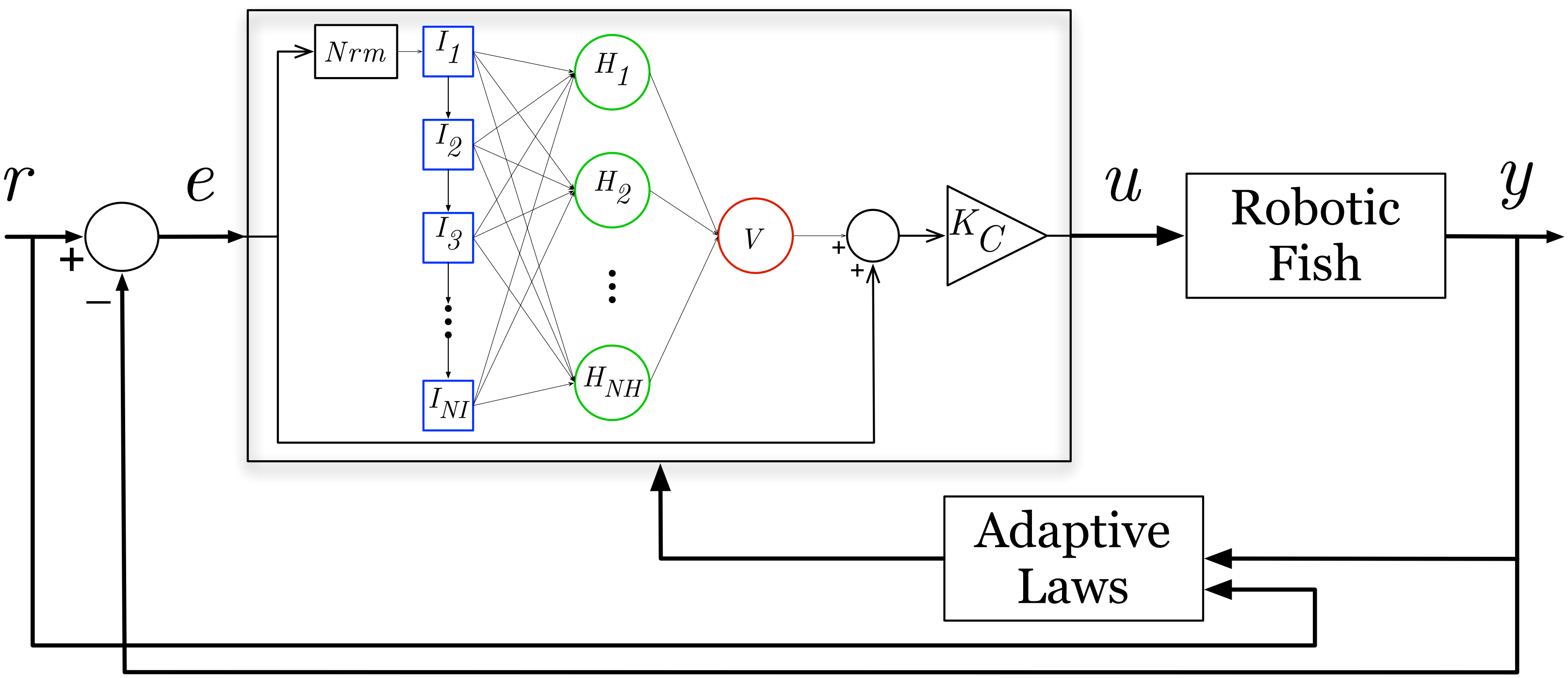
$$
\begin{aligned}
\Delta h_j(n) &\propto \frac{\partial E_s}{\partial h_j}, \\
&= \frac{\partial E_s}{\partial y}\frac{\partial y}{\partial h_j}, \\
&= \frac{\partial E_s}{\partial y}\frac{\partial y}{\partial u}\frac{\partial u}{\partial h_j}, \\
&= \frac{\partial E_s}{\partial y}\frac{\partial y}{\partial u}\frac{\partial u}{\partial o}\frac{\partial o}{\partial h_j}, \\
&= \boxed{-\eta\, K_c\, S_f(n)\, e(n)\, q_j.}
\end{aligned}
$$

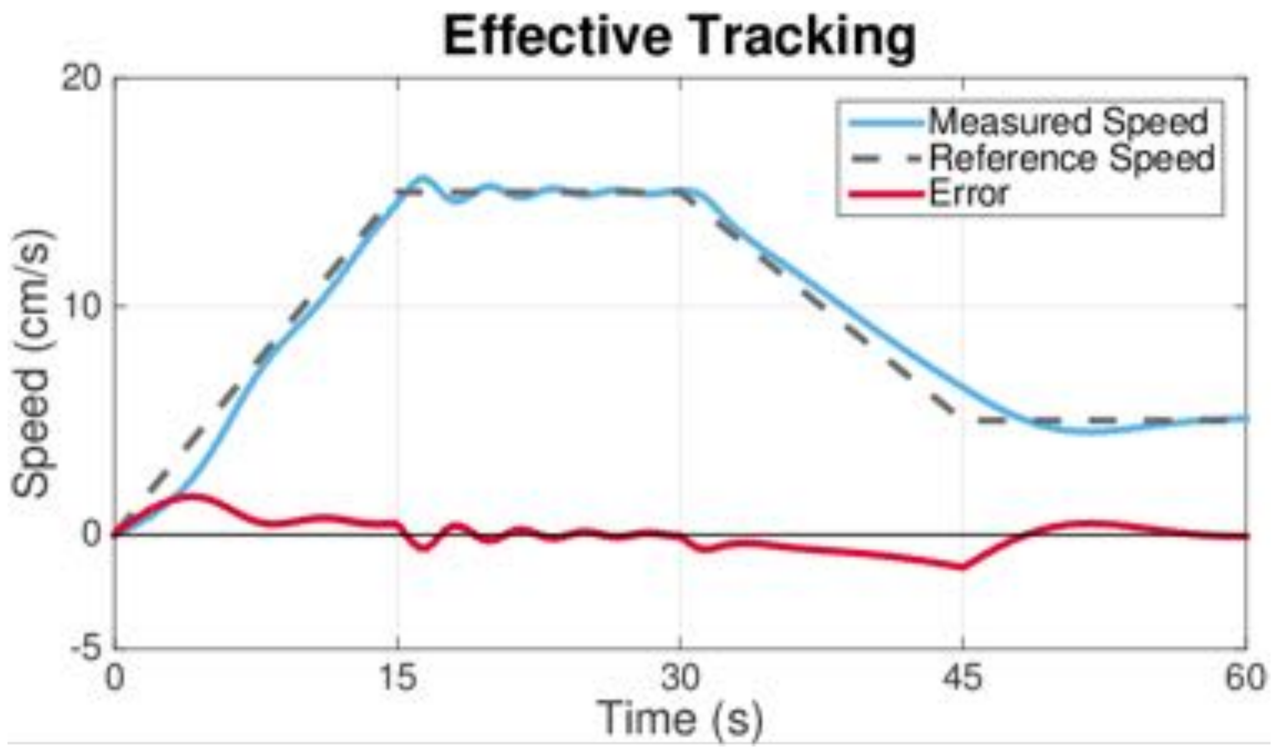
Evolvable Parameters

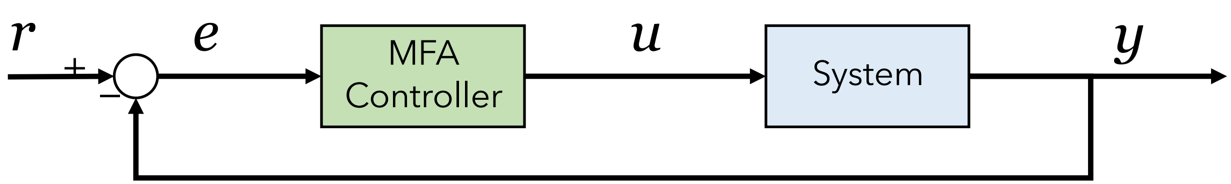
Adaptive Neural Network

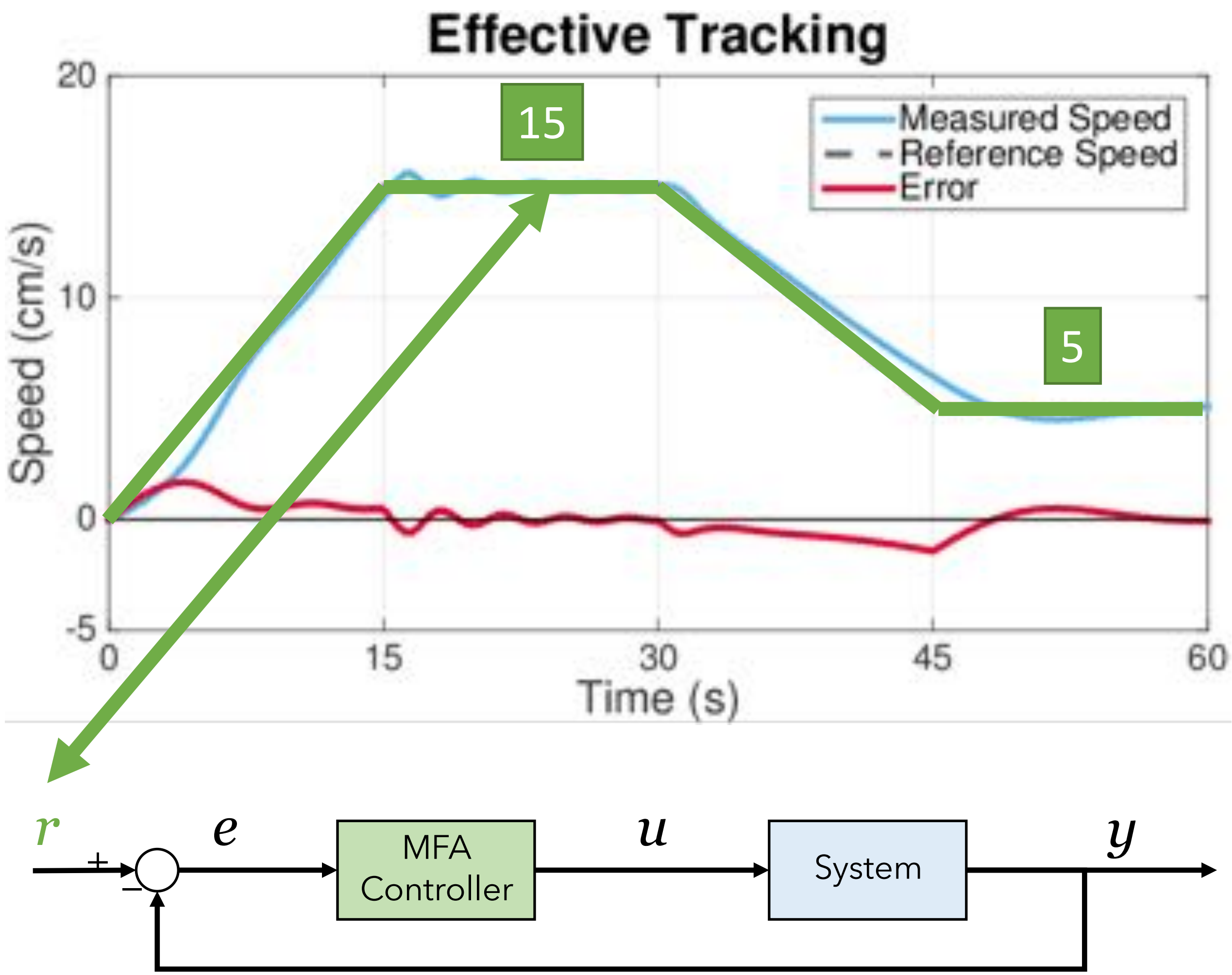
- neural network size/shape
- learning rate
- upper and lower error bounds
- controller gain
- controller update timing

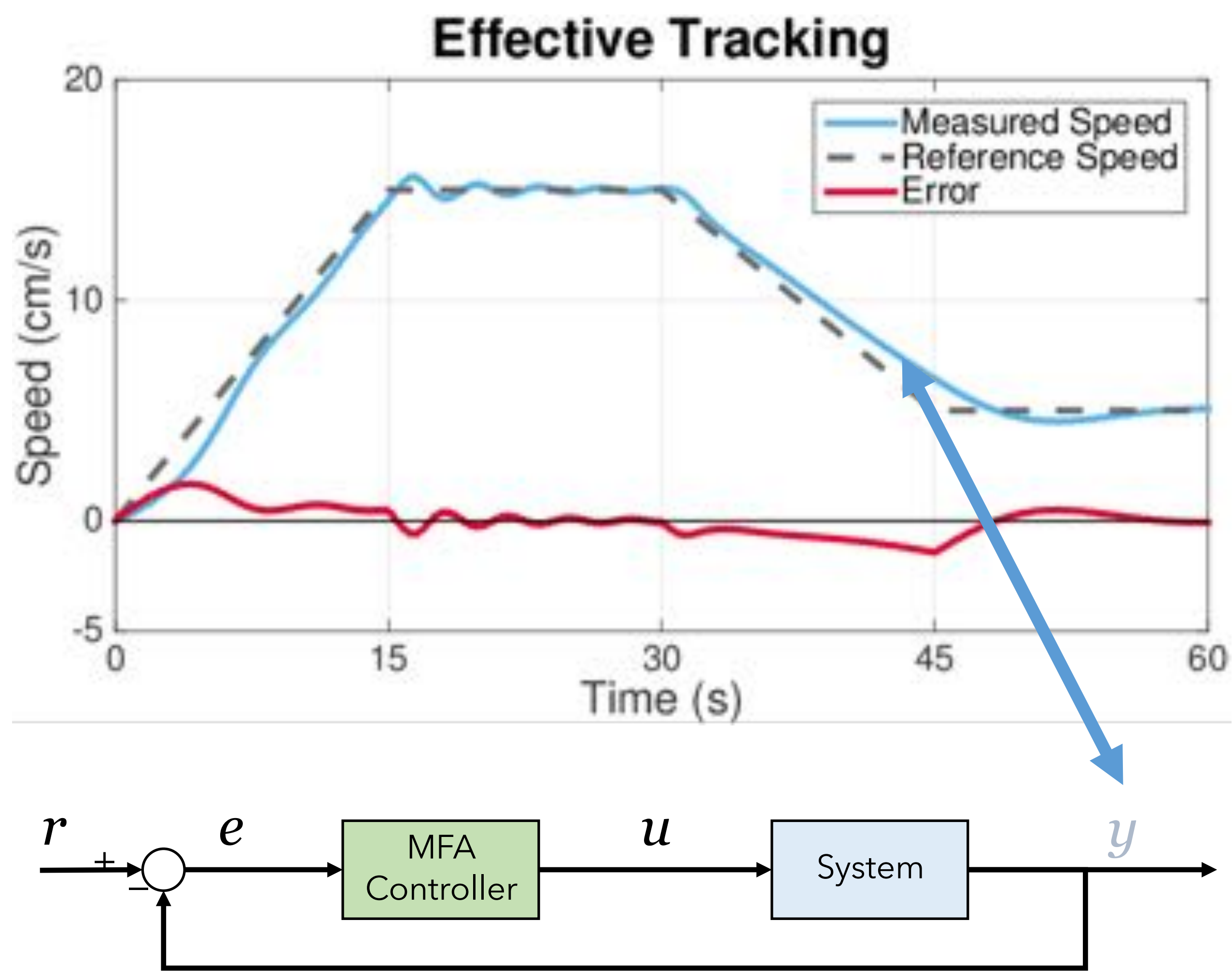


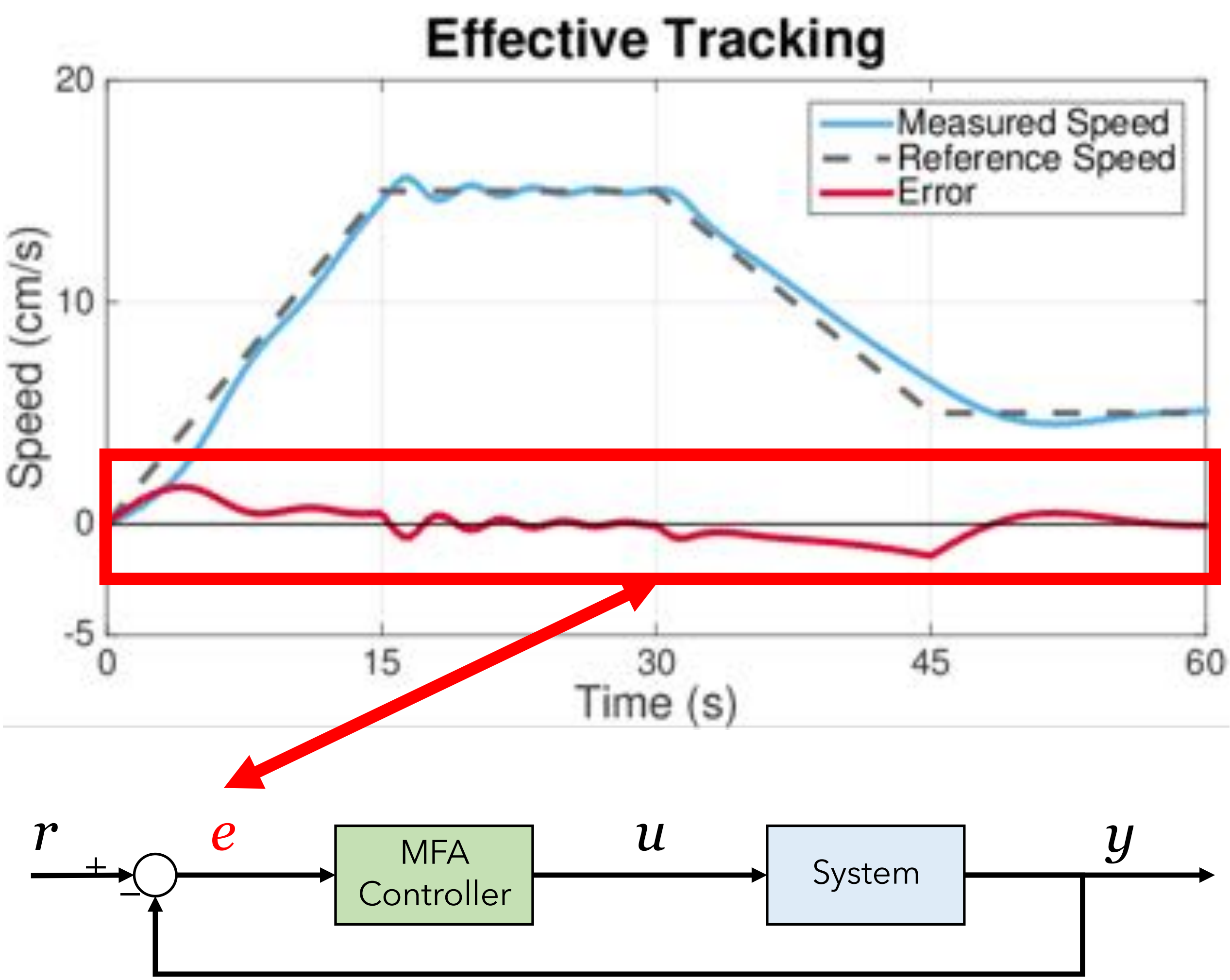

r
e
Nrm
I_1
I_2
I_3
I_{NI}
H_1
H_2
H_{NH}
V
+
+
K_C
u
Robotic Fish
y
+
−
Adaptive Laws

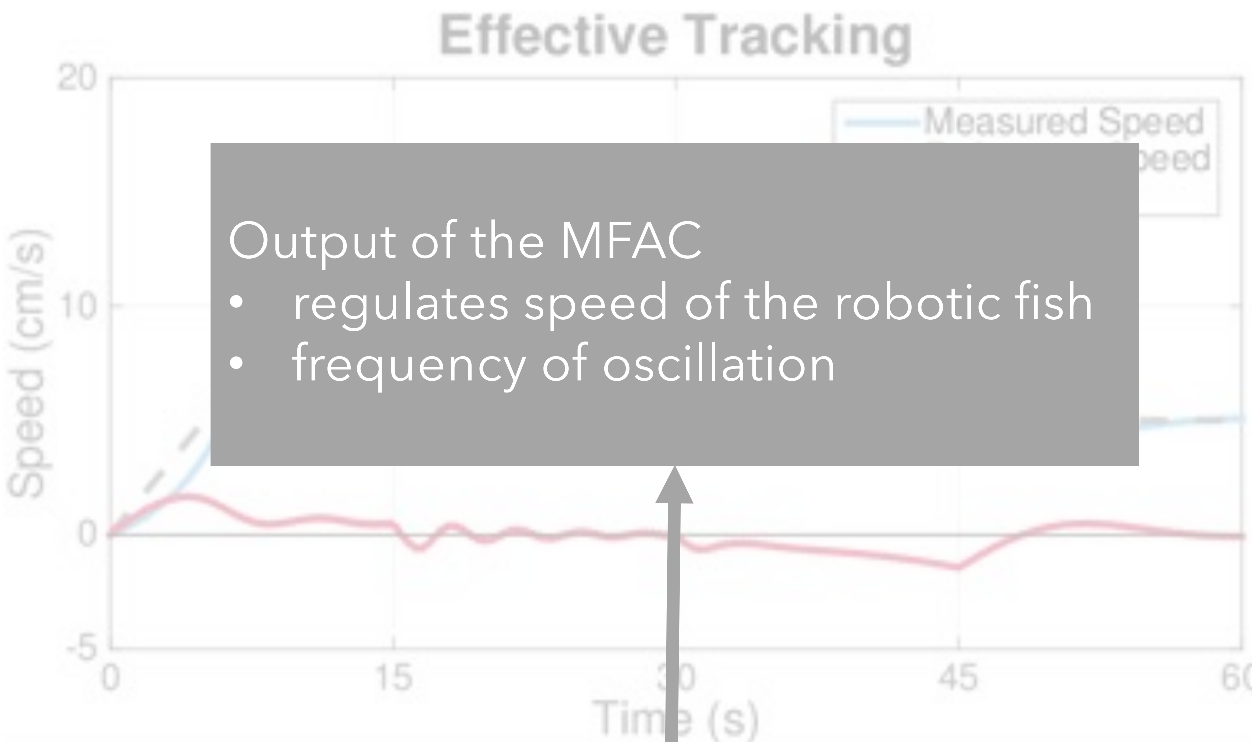

Effective Tracking
Measured Speed
Reference Speed
Error
Speed (cm/s)
Time (s)
20
10
0
-5
0
15
30
45
60


r
+
−
e
MFA Controller
u
System
y


Effective Tracking
20
10
0
-5
Speed (cm/s)
0
15
30
45
60
Time (s)
Measured Speed
Reference Speed
Error
15
5
r
e
MFA
Controller
u
System
y
+
−

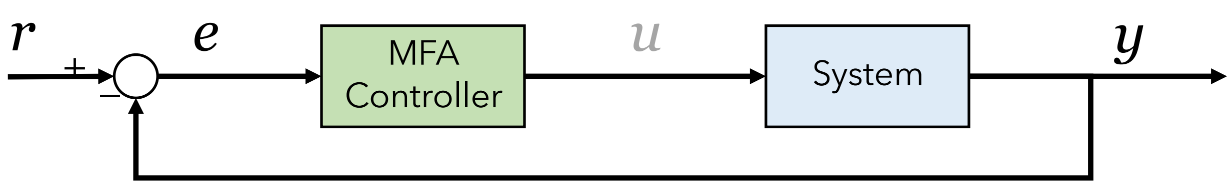

Effective Tracking
Speed (cm/s)
Time (s)
20
10
0
-5
0
15
30
45
60
Measured Speed
Reference Speed
Error
r
e
u
y
+
−
MFA Controller
System


Effective Tracking
Speed (cm/s)
20
10
0
-5
0
15
30
45
60
Time (s)
Measured Speed
Reference Speed
Error
r
+
−
e
MFA Controller
u
System
y



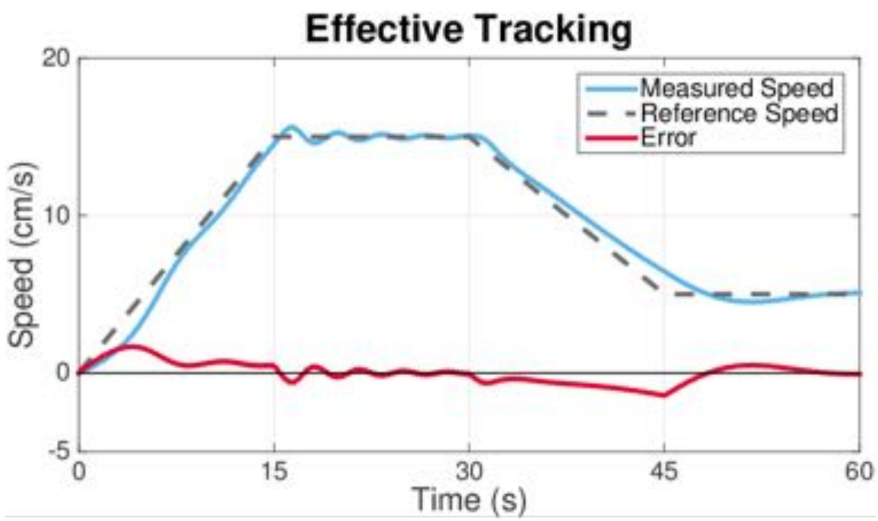
Output of the MFAC

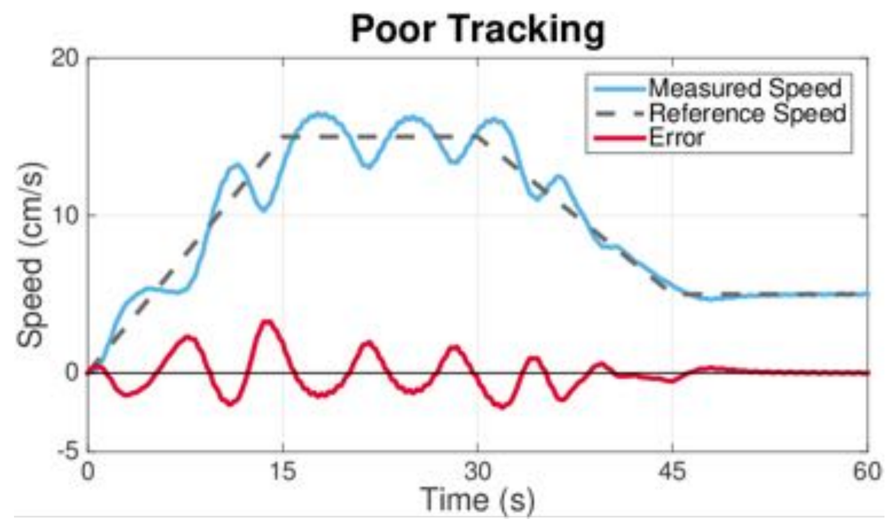
- regulates speed of the robotic fish
- frequency of oscillation



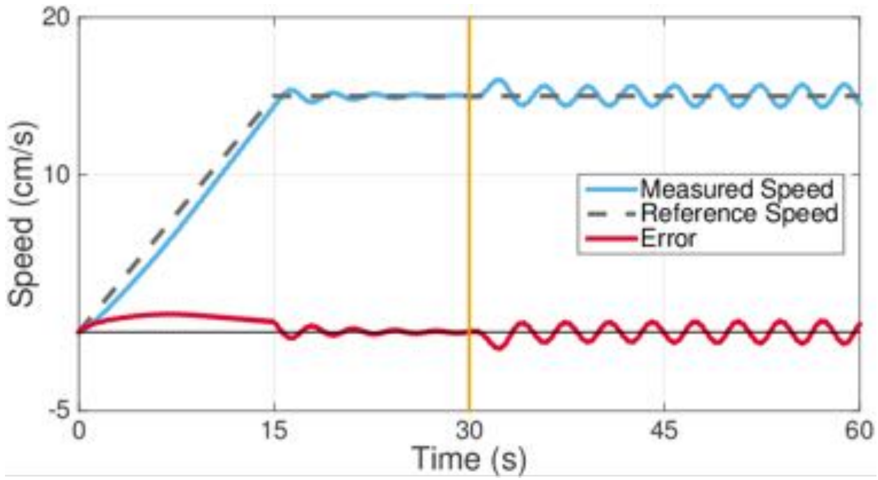
Anthony J. Clark – Missouri State University

Tracking Behavior

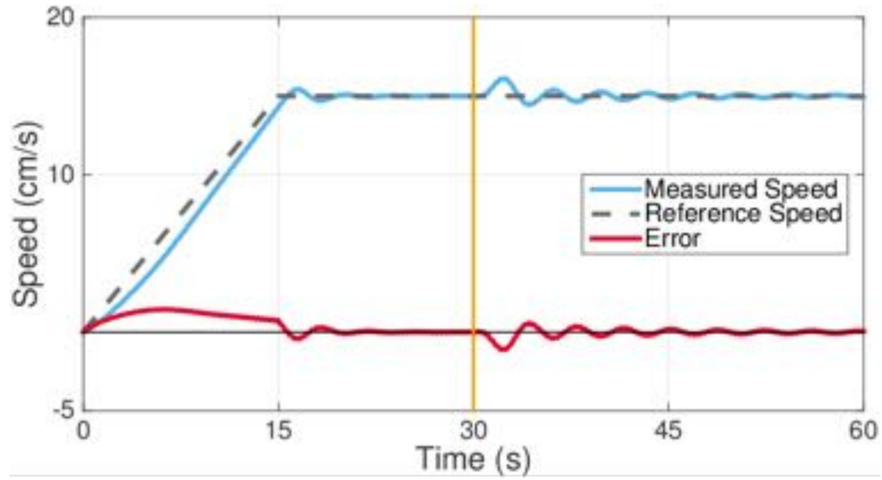




Adaptation

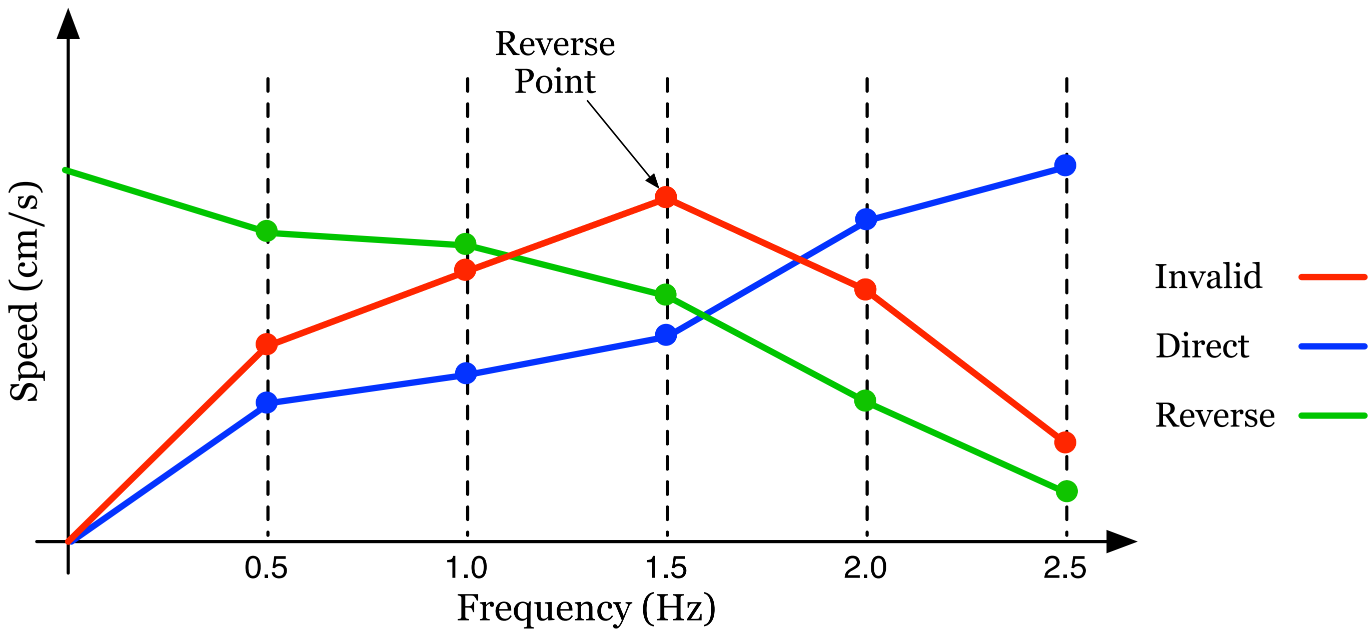


20
10
-5
0
15
30
45
60
Speed (cm/s)
Time (s)
Measured Speed
Reference Speed
Error

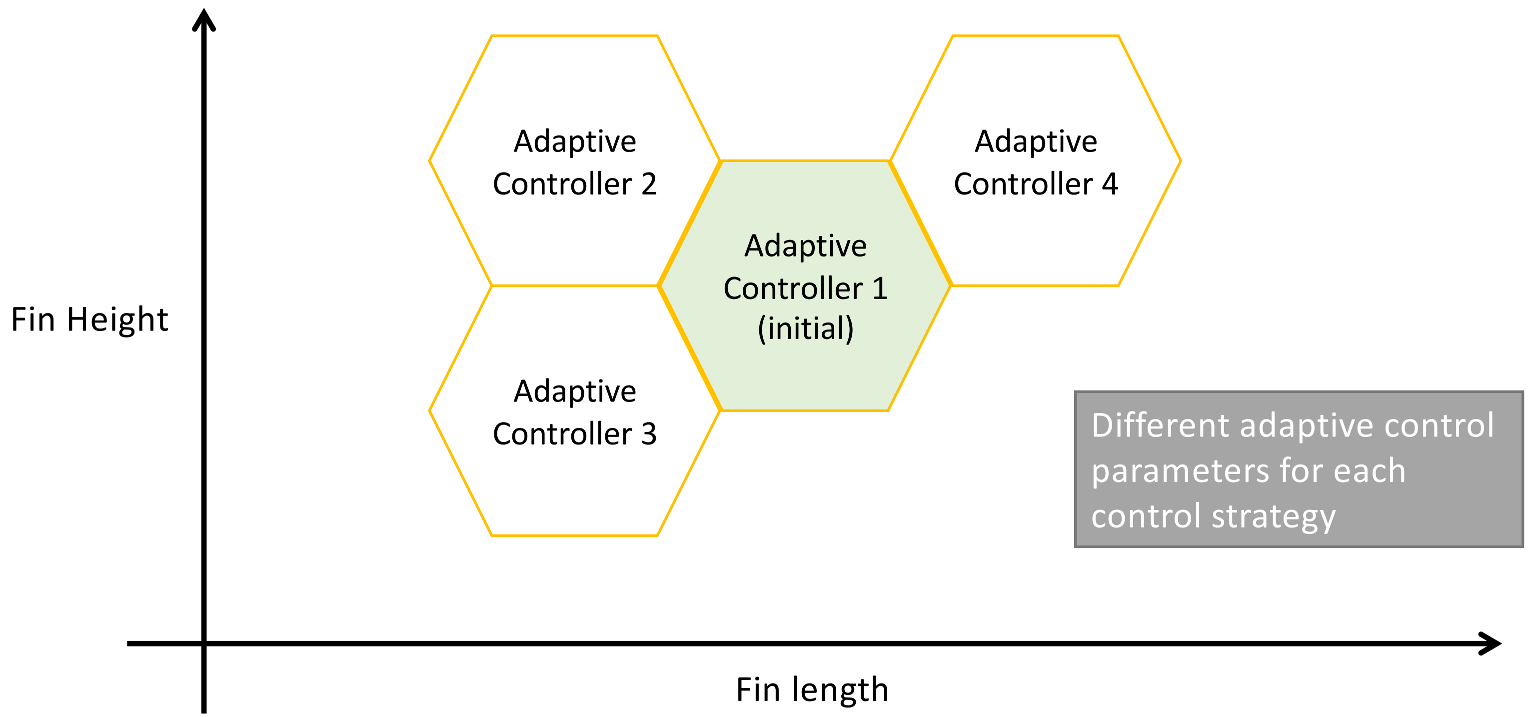


20
10
-5
0
15
30
45
60
Speed (cm/s)
Time (s)
Measured Speed
Reference Speed
Error

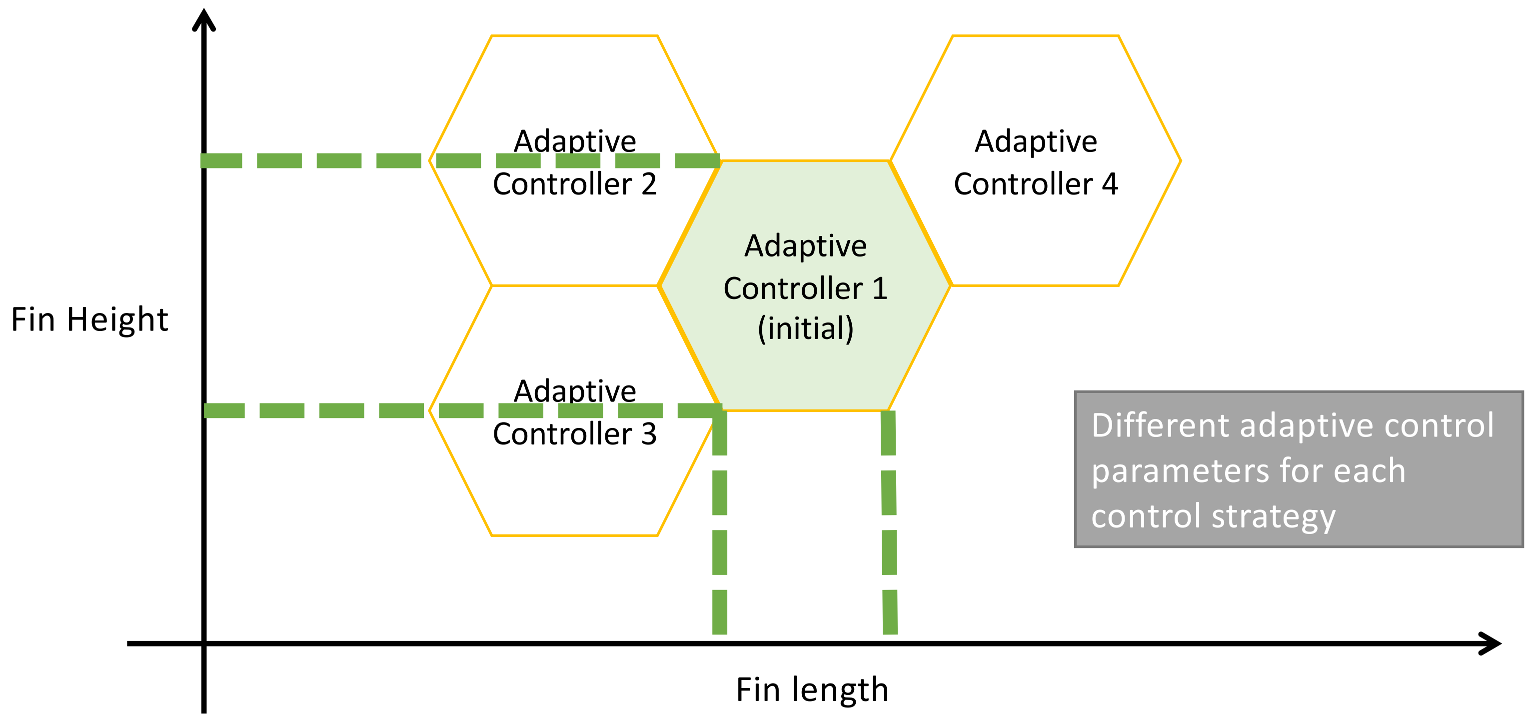
Limitations of Adaptation



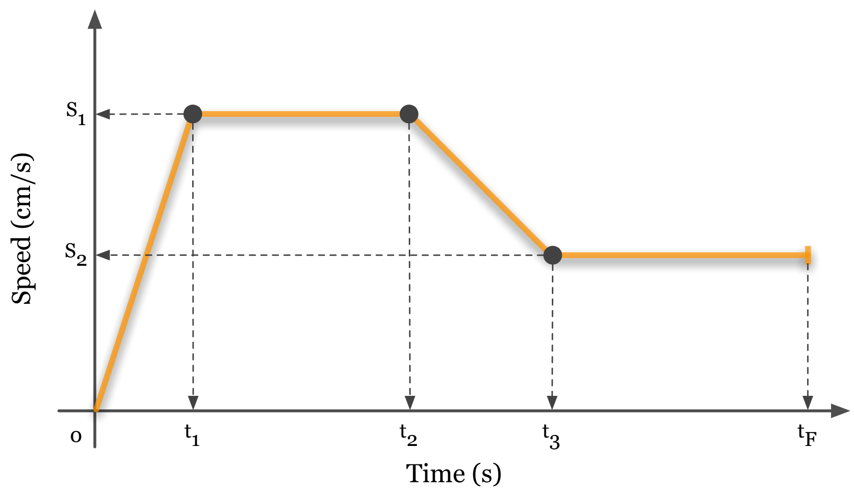
Representation of Execution Modes



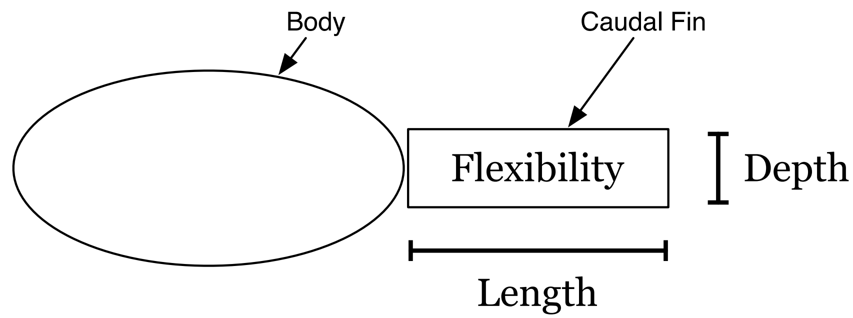
Representation of Execution Modes



Scenarios

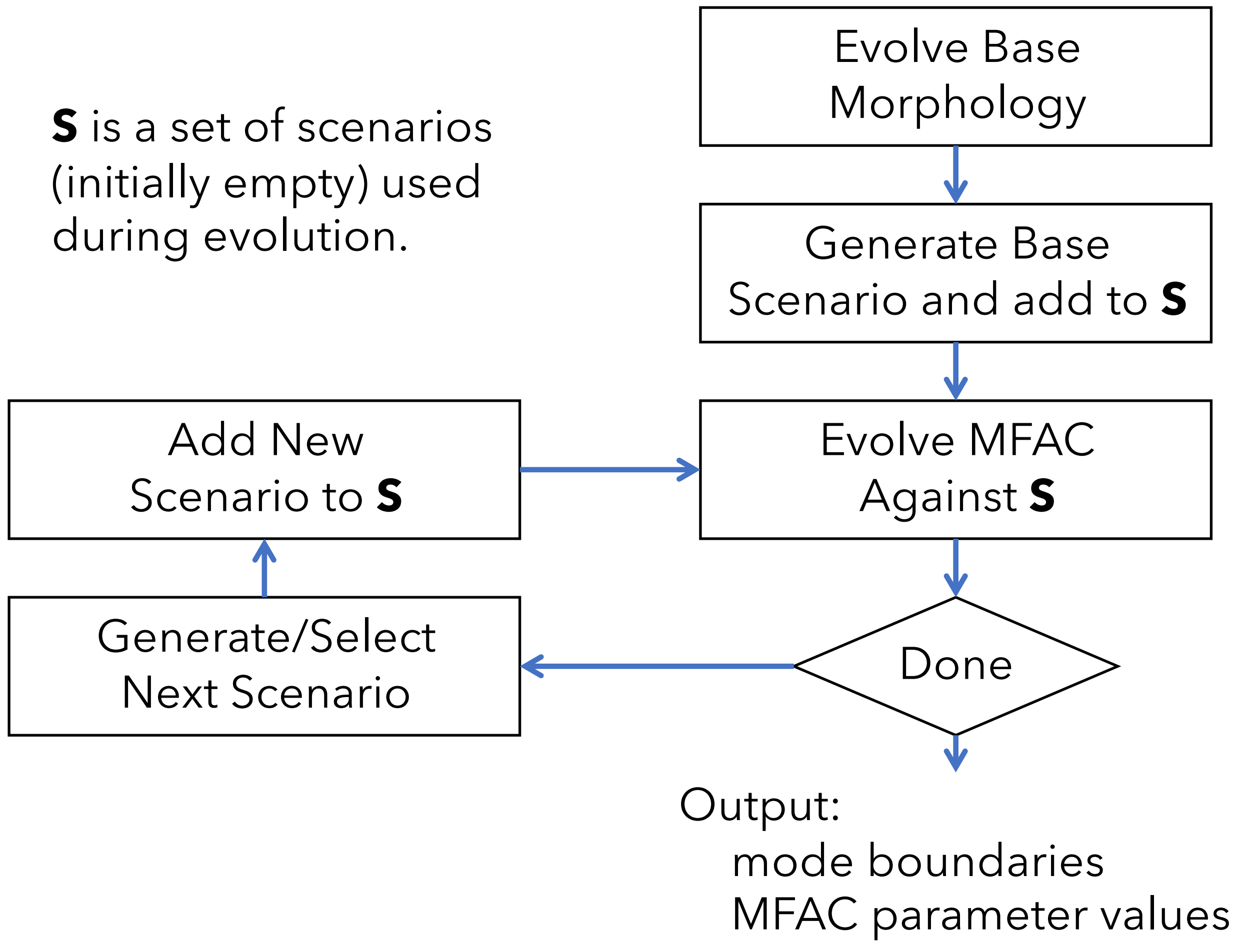


Input Reference Signal



Robotic Fish Diagram

S is a set of scenarios (initially empty) used during evolution.



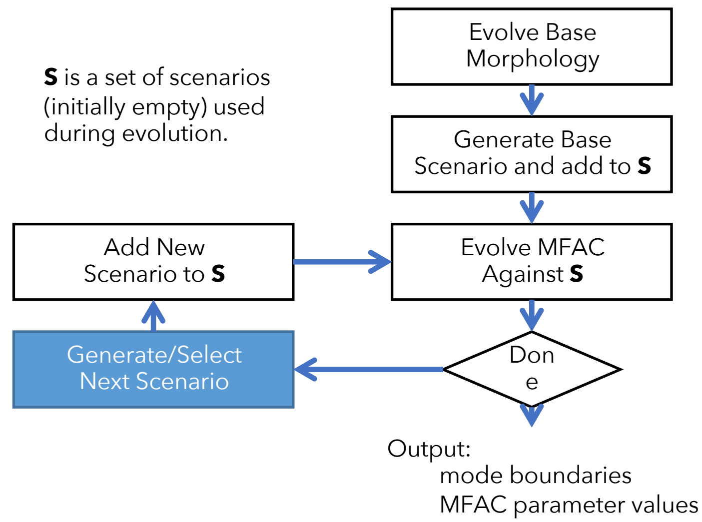
Boundary Selection Method

1. Select a scenario parameter
   i.e., fin length, height, flexibility

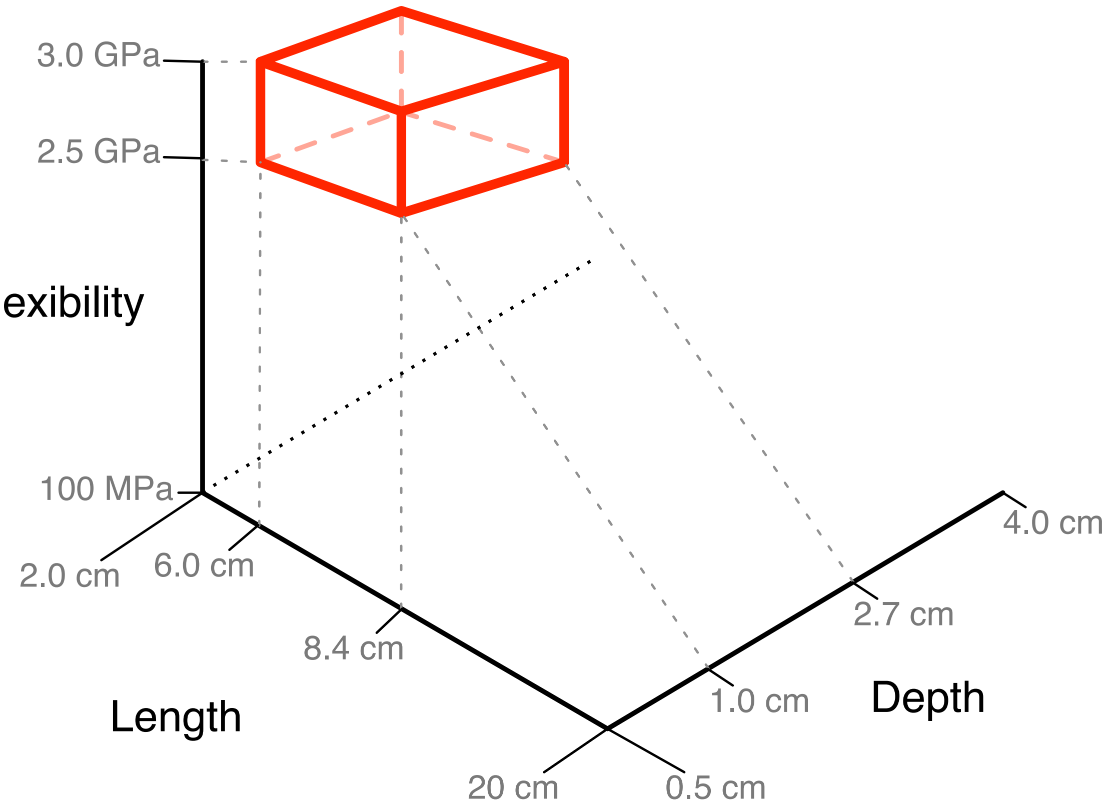
2. Select a direction
   (increase value or decrease value)

3. Increase/decrease parameter until the system becomes infeasible

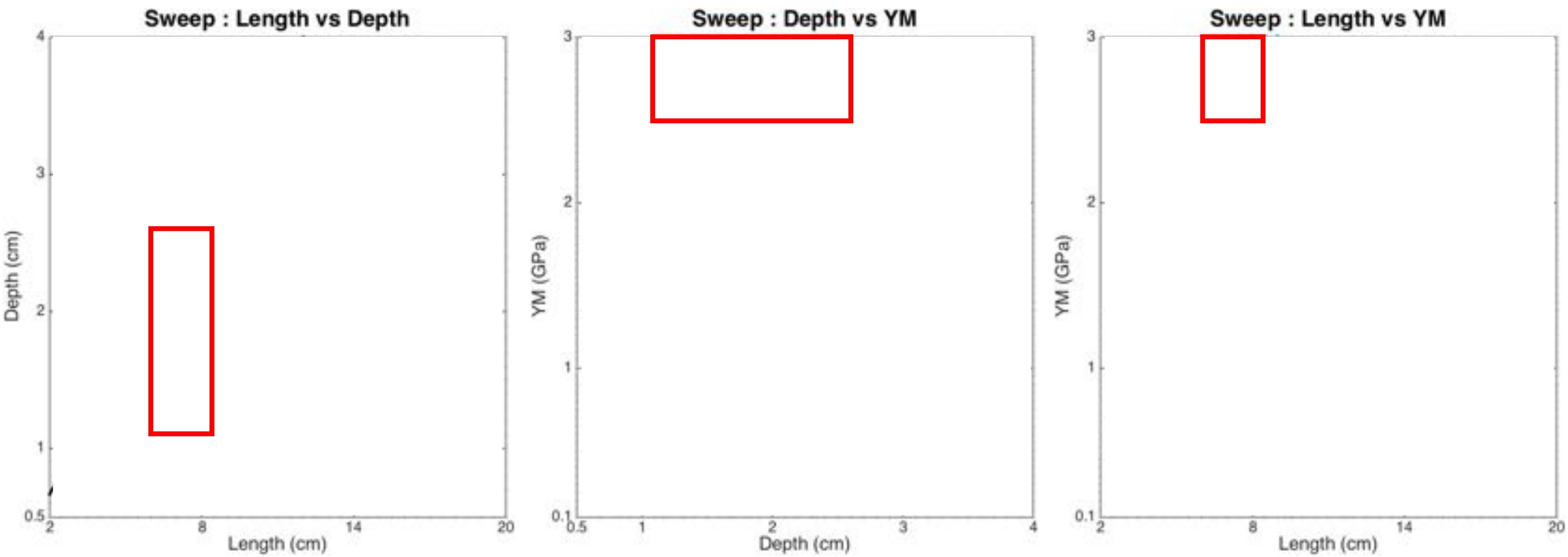
4. Add scenario to **S**



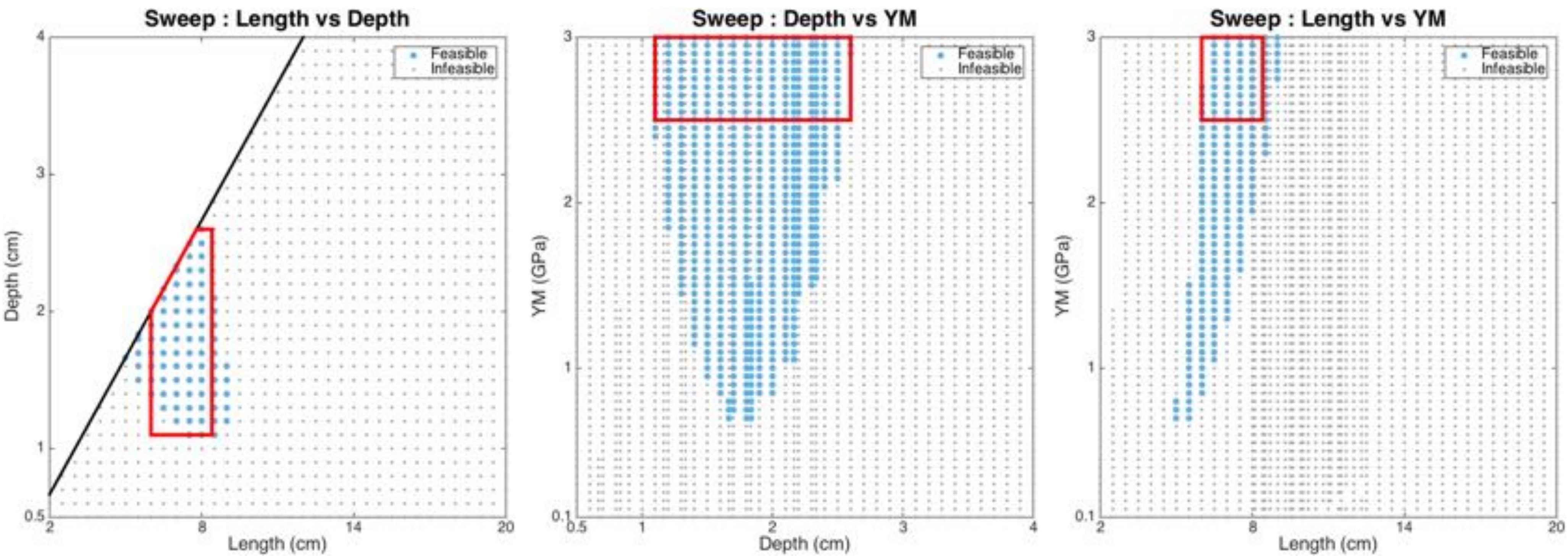
Boundary Scenarios



2D Views of Cuboid



“Ground-Truth”



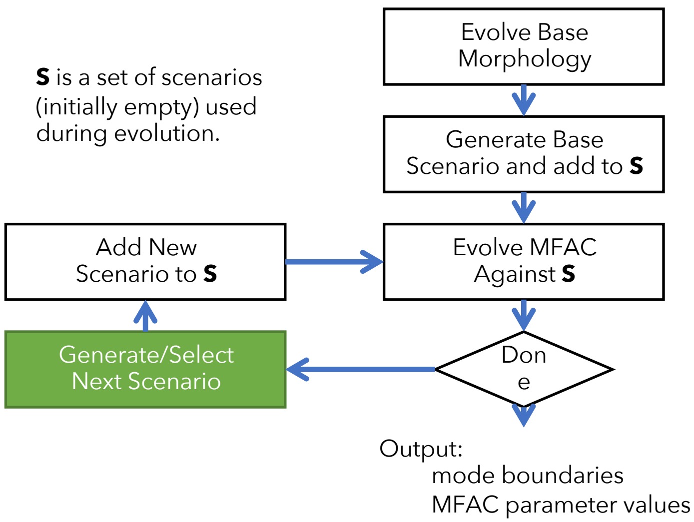
Volume Selection Method

1. Randomly generate 25 scenarios

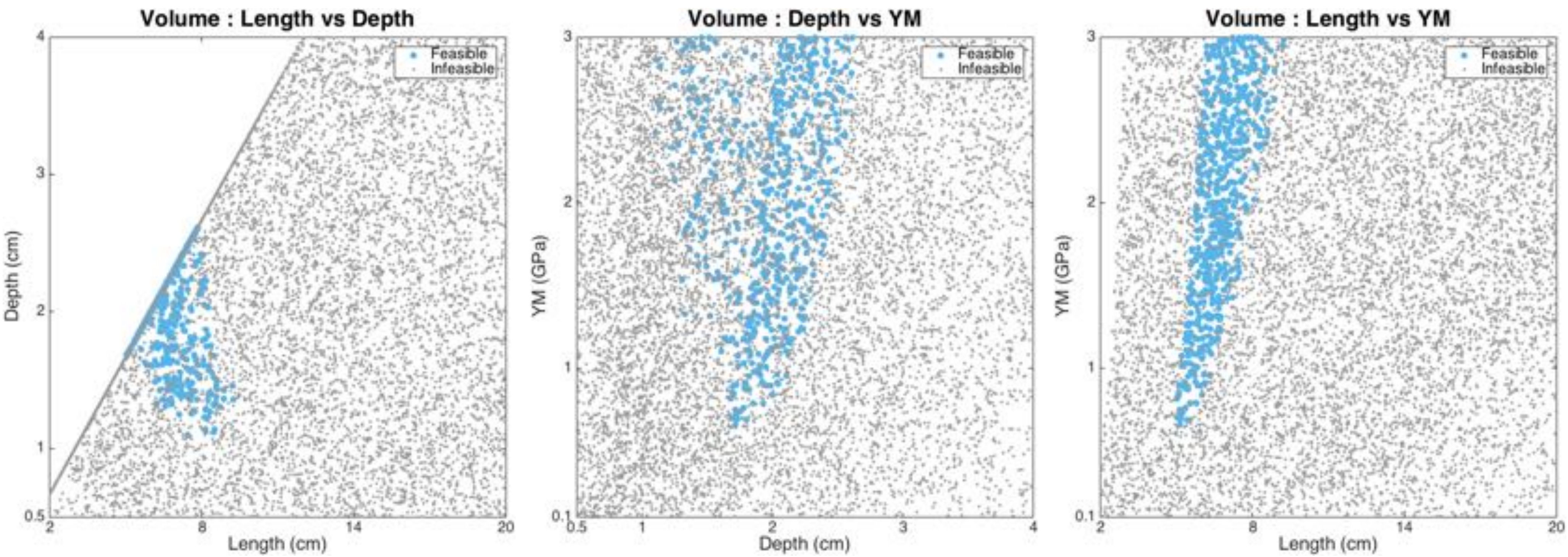
2. Evaluate all against the current best MFAC

3. Select the feasible scenario that produces the most error

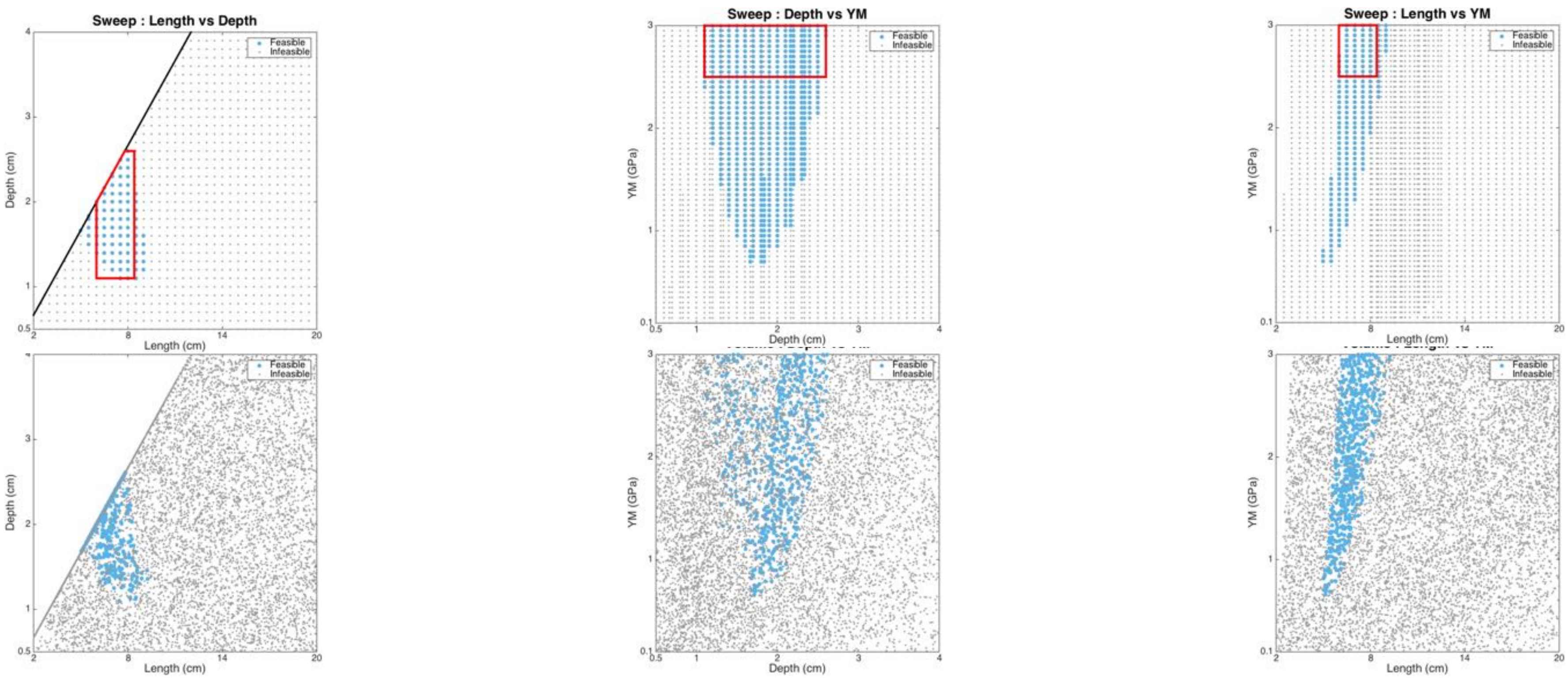
4. Add scenario to ***S***



Volume Scenarios



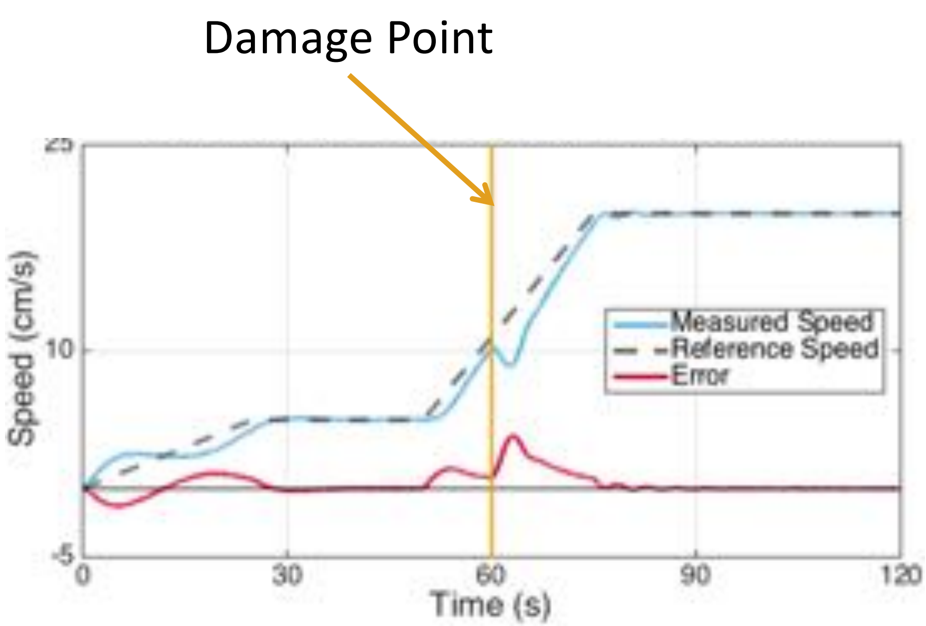
Volume Scenarios



Mean-Absolute-Error Comparison

Scenario Name	Boundary	Volume
Base	2.76 %	**2.60 %**
Min Length	9.30 %	**7.63 %**
Max Length	2.74 %	**2.73 %**
Min Depth	6.23 %	**4.87 %**
Max Depth	3.12 %	**2.92 %**
Random Boundary	4.70 %	**4.54 %**
Random Volume	3.19 %	**3.14 %**

Adapting to Damage



Fin length

- 8.0 → 6.4 cm

Fin Depth

- 2.6 → 2.1 cm

Fin Flex

- 3.0 → 2.1 GPa

Summary

- Automatically discover limits of an adaptive controller
- While at the same time optimizing the controller against "good" scenarios
- These limits define an execution mode
- Our future work involves combining this technique with self-modeling processes to account for automated switching between modes

The authors gratefully acknowledge the contributions and feedback on the work provided by the BEACON Center at Michigan State University.

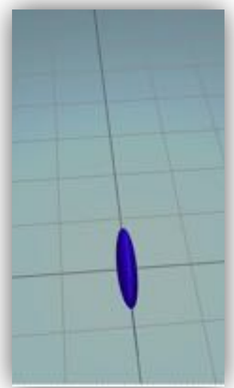
This work was supported in part by National Science Foundation grants CNS-1059373, DBI-0939454, and CNS-1305358, the Ford Motor Company, General Motors Research, and a grant from the Air Force Research Laboratory.

MICHIGAN STATE UNIVERSITY

BEACON An NSF Center for the Study of Evolution in Action

NSF

Thank You.
Questions?



Anthony J. Clark – Missouri State University