Morphological Computation and Soft Robotics

ShanghAI lectures, October 31, 2013

Helmut Hauser
Artificial Intelligence Laboratory

ai lab

University of Zurich
What is Morphological Computation?
Morphological computation is a term, which captures conceptually the observation that biological systems take advantage of their morphology to conduct computations needed for a successful interaction with their environments.
What is Morphological Computation?

Morphological computation is a term, which captures conceptually the observation that biological systems take advantage of their morphology to conduct computations needed for a successful interaction with their environments.
Morphological computation is a term, which captures *conceptually* the observation that *biological systems* take advantage of their *morphology* to conduct computations needed for a successful interaction with their environments.
Morphological computation is a term, which captures conceptually the observation that biological systems take advantage of their morphology to conduct computations needed for a successful interaction with their environments.
Morphological computation is a term, which captures conceptually the observation that biological systems take advantage of their morphology to conduct computations needed for a successful interaction with their environments.
Morphological computation is a term, which captures conceptually the observation that biological systems take advantage of their morphology to conduct computations needed for a successful interaction with their environments.
What is Soft Robotics?
What is Soft Robotics?

- use of soft material
What is Soft Robotics?

- use of soft material
What is Soft Robotics?

- use of soft material
- soft actuators

Soft Robotics

OCTOPUS

Havard

Tufts

ECCE

Roboy
What is Soft Robotics?

- use of soft material
- soft actuators
- often bio-inspired or biomimetic approaches
What is Soft Robotics?

- use of soft material
- soft actuators
- often bio-inspired or biomimetic approaches
- safe interaction, energy efficiency, highly resilient

OCTOPUS

Havard

Tufts

ECCE

Roboy
What is Soft Robotics?

- use of soft material
- soft actuators
- often bio-inspired or biomimetic approaches
- safe interaction, energy efficiency, highly resilient

![Images of OCTOPUS, Havard, Tufts, ECCE, Roboy]
How are they connected?

Morphological Computation

Soft Robotics
How are they connected?

Morphological Computation

Soft Robotics

theoretical models for MC
How are they connected?

- High dimensionality

Morphological Computation

Soft Robotics

theoretical models for MC
How are they connected?

Morphological Computation

Soft Robotics

- High dimensionality
- Nonlinearity

theoretical models for MC
How are they connected?

- High dimensionality
- Nonlinearity
- Compliance

Morphological Computation

Soft Robotics

theoretical models for MC
How are they connected?

- High dimensionality
- Nonlinearity
- Compliance
- Noise

Morphological Computation

Soft Robotics

Theoretical models for MC
How are they connected?

- High dimensionality
- Nonlinearity
- Compliance
- Noise

theoretical models for MC
How are they connected?

Morphological Computation

• High dimensionality
• Nonlinearity
• Compliance
• Noise

Soft Robotics

theoretical models for MC
Classical Robot Design

- rigid body parts
- high torque servos
- keep DoFs as low as possible
- fully actuated (at all time)
Classical Robot Design

- rigid body parts
- high torque servos
- keep DoFs as low as possible
- fully actuated (at all time)

To facilitate control
“Nature’s Approach”

- soft body parts
- compliant muscle-tendon system
- large number of DoFs
- underactuation (passive DoFs)
“Nature’s Approach”

- soft body parts
- compliant muscle-tendon system
- large number of DoFs
- underactuation (passive DoFs)

nightmare!
“Nature’s Approach”

- soft body parts
- compliant muscle-tendon system
- large number of DoFs
- underactuation (passive DoFs)

nightmare! to facilitate control
What Kind Of Morphological Computation?

Morphology?

Computation?
What Kind of Morphology?
What Kind of Morphology?

- Form, shape
What Kind of Morphology?

- Form, shape
- Physical parameters like spring constants, damping, friction, etc.
What Kind of Morphology?

- Form, shape
- Physical parameters like spring constants, damping, friction, etc.
- Location of sensors and actuators
What Kind of Morphology?

- Form, shape
- Physical parameters like spring constants, damping, friction, etc.
- Location of sensors and actuators
- Morphology of the environment
What Kind Of Morphological Computation?

Morphology ?

Computation ?
What Kind of Computation?

Computation?

Turing

program

read/write

tape
What Kind of Computation?

- Well defined, powerful concept
What Kind of Computation?

- Well defined, powerful concept
- However, not very biological
What Kind of Computation?

- Well defined, powerful concept
- However, not very biological
- Nature does not wait for the end of a computation
What Kind of Computation?

- Well defined, powerful concept
- However, not very biological
- Nature does not wait for the end of a computation
- Continuous never-ending computation with feedback
What Kind of Computation?

- Well defined, powerful concept
- However, not very biological
- Nature does not wait for the end of a computation
- Continuous never-ending computation with feedback
- Mathematically speaking, we need operators
What Kind of Computation?

input functions \rightarrow \text{mapping} \rightarrow \text{output function}

\text{mathematical operator}
What Kind of Computation?

mathematical operator

mapping

input functions

sensory input streams

mathematical operators

output function

motor output streams
What Kind of Computation?

mathematical operator

mapping

input functions

mathematical operators

sensory input streams

output function

motor output streams

encodes our computation
What Kind of Computation?

- Mathematical operator

  - Input functions
  - Sensory input streams
  - Morphology
  - Mathematical operators
  - Mapping

  - Output function
  - Motor output streams

Encodes our computation
What Kind of Computation?

input functions

mathematical operators

mapping

morphology

output functions

sensory input streams

continuous (analog) computation

motor output streams
First Theoretical Model

- Based on a result by [Boyd and Chua 1985]

First Theoretical Model

- Based on a result by [Boyd and Chua 1985]
- Arbitrary time invariant operators with fading memory

First Theoretical Model

- Based on a result by [Boyd and Chua 1985]
- Arbitrary time invariant operators with fading memory
- Encoding our computational task
First Theoretical Model

- Based on a result by [Boyd and Chua 1985]
- Arbitrary time invariant operators with fading memory
  - Encoding our computational task
  - Nonlinear, dynamic operator
First Theoretical Model

- Based on a result by [Boyd and Chua 1985]
- Arbitrary time invariant operators with fading memory

- Encoding our computational task
- Nonlinear, dynamic operator
**First Theoretical Model**

- Based on a result by [Boyd and Chua 1985]
- Arbitrary *time invariant* operators with *fading memory*

- Encoding our computational task
- Nonlinear, dynamic operator
- Any exp. stable nonlinear dynamic system with one point of equilibrium
First Theoretical Model

- Based on a result by [Boyd and Chua 1985]
- Arbitrary time invariant operators with fading memory

- Encoding our computational task
- Nonlinear, dynamic operator
- Any exp. stable nonlinear dynamic system with one point of equilibrium
- Nonlinear controller
First Theoretical Model

• Based on a result by [Boyd and Chua 1985]

• Arbitrary time invariant operators with fading memory

memory and nonlinearity

**First Theoretical Model**

- Based on a result by [Boyd and Chua 1985]
- Arbitrary **time invariant** operators with **fading memory**

First Theoretical Model

- Based on a result by [Boyd and Chua 1985]
- Arbitrary time invariant operators with fading memory

First Theoretical Model

- Based on a result by [Boyd and Chua 1985]

- Arbitrary **time invariant** operators with **fading memory** can be uniformly approximated by computational devices, which consist of two simple stages:

  ![Diagram](attachment:image.png)

  - **Temporal integration**
  - **Nonlinear combination**

  1. Linear, dynamic
  2. Nonlinear, static
First Theoretical Model

- Based on a result by [Boyd and Chua 1985]

- Arbitrary time invariant operators with fading memory can be uniformly approximated by computational devices, which consist of two simple stages:

![Diagram showing the components of the model: temporal integration, nonlinear combination, linear, dynamic, and nonlinear, static stages.]

\[ B_1 \quad B_2 \quad \cdots \quad B_k \]
Arbitrary time invariant operators with fading memory can be uniformly approximated by computational devices, which consist of two simple stages:

Stage 1

- Has to integrate information over time (fading memory)
- Has to separate signals

Based on a result by [Boyd and Chua 1985]
Arbitrary time invariant operators with fading memory can be uniformly approximated by computational devices, which consist of two simple stages:

Stage 1

Has to integrate information over time (fading memory)

Has to separate signals

Based on a result by [Boyd and Chua 1985]

\[ B_1 \quad B_2 \quad \ldots \quad B_k \]
Arbitrary time invariant operators with fading memory can be uniformly approximated by computational devices, which consist of two simple stages:

Stage 1

Has to integrate information over time (fading memory)

Has to separate signals

Stage 1

B₁

B₂

... Bₖ

linear, dynamic

nonlinear, static

First Theoretical Model
Arbitrary time invariant operators with fading memory can be uniformly approximated by computational devices, which consist of two simple stages:

- **Stage 1**
  - Has to **integrate** information over time (fading memory)
  - Has to **separate** signals

Based on a result by [Boyd and Chua 1985](#),

\[ B_1, B_2, \ldots, B_k \]

- **Linear, dynamic**
- **Nonlinear, static**
Arbitrary time invariant operators with fading memory can be uniformly approximated by computational devices, which consist of two simple stages:

**First Theoretical Model**

Based on a result by [Boyd and Chua 1985]

Stage 1

- Has to **integrate** information over time (fading memory)
- Has to **separate** signals

Has to separate signals
First Theoretical Model

- Based on a result by [Boyd and Chua 1985]

Stage 1

Has to **integrate** information over time (fading memory)

Has to **separate** signals
Arbitrary time-invariant operators with fading memory can be uniformly approximated by computational devices, which consist of two simple stages:

**Stage 1**

- Has to **integrate** information over time (fading memory)
- Has to **separate** signals

linear, dynamic → \( f \) → nonlinear, static
Arbitrary time invariant operators with fading memory can be uniformly approximated by computational devices, which consist of two simple stages:

Stage 1

- Has to integrate information over time (fading memory)
- Has to separate signals

Based on a result by [Boyd and Chua 1985]
First Theoretical Model

• Based on a result by [Boyd and Chua 1985]

• Arbitrary time invariant memory can be uniformly approximated by computational devices, which consist of two simple stages:
  
  **Stage 2**
  
  Has to combine integrated information nonlinearily

- **First Theoretical Model**
  
  - Temporal integration
  - Nonlinear combination
  - Linear, dynamic
  - Nonlinear, static

Stage 2

Has to combine integrated information nonlinearily
First Theoretical Model

nonlinear, dynamic operator

temporal integration

linear, dynamic

nonlinear, static

nonlinear combination
First Theoretical Model

- Temporal integration
- Nonlinear, dynamic operator

- Linear, dynamic
- Nonlinear, static

- Nonlinear combination

- \( B_1 \)
- \( B_2 \)
- \( \cdots \)
- \( B_k \)
First Theoretical Model

nonlinear, dynamic operator

temporal integration

linear, dynamic

nonlinear combination

nonlinear, static
First Theoretical Model

- Temporal integration
- Nonlinear, dynamic operator
- Nonlinear, dynamic emulation
- Linear, dynamic
- Nonlinear, static
First Theoretical Model

nonlinear, dynamic operator

emulation

with the help of morphology

temporal integration

linear, dynamic

nonlinear, static

nonlinear, static
First Theoretical Model

nonlinear, dynamic operator

emulation

temporal integration

nonlinear combination

linear, dynamic

nonlinear, static
First Theoretical Model

nonlinear, dynamic operator

emulation

temporal integration

nonlinear combination

linear, dynamic

nonlinear, static
Remarkable Conclusion

- Nonlinear, dynamic operator

- Temporal integration

- Nonlinear combination

- Linear, dynamic

- Nonlinear, static

- ANN
Remarkable Conclusion

Outsourcing part of the computation to the morphology

nonlinear, dynamic operator

temporal integration

nonlinear combination

linear, dynamic

ANN

nonlinear, static

\[
B_1 \quad B_2 \quad \ldots \quad B_k
\]

\[
\ldots
\]

\[
\ldots
\]

\[
f
\]
Remarkable Conclusion

Outsourcing part of the computation to the morphology

resulting task is easier: finding some static weights

nonlinear, dynamic operator

temporal integration

nonlinear combination

linear, dynamic

ANN

nonlinear, static
Remarkable Conclusion

Outsourcing part of the computation to the morphology

resulting task is easier: finding some **static** weights
Remarks
Remarks

- Theories just tells us that it is theoretically possible to emulate any nonlinear, time invariant operator with fading memory
Remarks

- Theories just tells us that it is theoretically possible to emulate any nonlinear, time invariant operator with fading memory.
- However, for a given task it does not tell us which mass-spring systems nor how many we have to use.
• Theories just tells us that it is theoretically possible to emulate any nonlinear, time invariant operator with fading memory
• However, for a given task it does not tell us which mass-spring systems nor how many we have to use
• Either we use nonlinear optimization schemes or generic networks
Remarks

- Theories just tells us that it is theoretically possible to emulate any nonlinear, time invariant operator with fading memory.
- However, for a given task it does not tell us which mass-spring systems nor how many we have to use.
- Either we use **nonlinear optimization schemes** or **generic networks**.
- We used **generic, random** morphologies → they are not constructed for a specific task.
Remarks

- Theories just tells us that it is theoretically possible to emulate any nonlinear, time invariant operator with fading memory
- However, for a given task it does not tell us which mass-spring systems nor how many we have to use
- Either we use nonlinear optimization schemes or generic networks
- We used generic, random morphologies → they are not constructed for a specific task
- → this implies multitasking
Theories just tells us that it is theoretically possible to emulate any nonlinear, time invariant operator with fading memory. However, for a given task it does not tell us which mass-spring systems nor how many we have to use. Either we use nonlinear optimization schemes or generic networks. We used generic, random morphologies → they are not constructed for a specific task. → this implies multitasking.
• Theories just tell us that it is theoretically possible to emulate any nonlinear, time invariant operator with fading memory.
• However, for a given task it does not tell us which mass-spring systems nor how many we have to use.
• Either we use **nonlinear optimization schemes** or **generic networks**.
• We used **generic, random** morphologies $\rightarrow$ they are not constructed for a specific task.
• $\rightarrow$ this implies **multitasking**.

exploitation of the body
Computational Power
Has to **integrate** information over time (fading memory)

Has to **separate** signals
Computational Power

- compliance is important

Has to integrate information over time (fading memory)

Has to separate signals
Computational Power

- Compliance is important
- The number of mass-spring systems should be high

Has to integrate information over time (fading memory)

Has to separate signals
Computational Power

- compliance is important
- The number of mass-spring systems should be high
- The diversity of mass-spring systems is important (i.e., different physical parameters)

Has to **integrate** information over time (fading memory)

Has to **separate** signals

**soft robots**
Can the Physical Body Do More?

temporal integration

nonlinear combination

ANN
Can the Physical Body Do More?

- Nonlinear, static mapping could also be done by "kernel"
Can the Physical Body Do More?

- Nonlinear, static mapping could also be done by "kernel"
- Idea: body is a physical implementation of such a finite kernel
Can the Physical Body Do More?

- Nonlinear, static mapping could also be done by "kernel"
- Idea: body is a physical implementation of such a finite kernel
- E.g., recurrent network of nonlinear springs and masses
Can the Physical Body Do More?

- Nonlinear, static mapping could also be done by "kernel"
- Idea: body is a physical implementation of such a finite kernel
- E.g., recurrent network of nonlinear springs and masses
Can the Physical Body Do More?

- Nonlinear, static mapping could also be done by "kernel"
- Idea: body is a physical implementation of such a finite kernel
- E.g., recurrent network of nonlinear springs and masses

No theoretical proof anymore!
Remarkable Conclusion

nonlinear, dynamic operator

temporal integration  nonlinear combination

...
Remarkable Conclusion

- Outsourcing big part of the computation to the morphology

- Nonlinear, dynamic operator
  - Temporal integration
  - Nonlinear combination

- Diagram representation
Remarkable Conclusion

- Nonlinear, dynamic operator
- Temporal integration
- Nonlinear combination

Outsourcing big part of the computation to the morphology

Resulting task is easier: linear regression
Remarkable Conclusion

nonlinear, dynamic operator

Outsourcing big part of the computation to the morphology

resulting task is easier: linear regression
Remarkable Conclusion

temporal integration  nonlinear combination

...
Remarkable Conclusion

randomly initialized!

temporal integration

nonlinear combination

Σ
randomly initialized!

temporal integration

nonlinear combination

high-dimensional
Remarkable Conclusion

randomly initialized!

temporal integration

nonlinear combination

high-dimensional nonlinear

∑
Remarkable Conclusion

randomly initialized!

temporal integration

nonlinear combination

high-dimensional nonlinear compliant

\[ \sum \]
Remarkable Conclusion

- high-dimensional
- nonlinear
- compliant
Remarkable Conclusion

- high-dimensional
- nonlinear
- compliant

I don’t like that!
Remarkable Conclusion

- high-dimensional
- nonlinear
- compliant

I don’t like that!

Actually, I do like that!
Limitation

temporal integration

nonlinear combination

\[ \sum \]
The theoretical model is limited to time-invariant operators with fading memory.
The theoretical model is limited to time-invariant operators with fading memory. Persistent memory is of interest too, or limit cycles for locomotion.
Limitation

- The theoretical model is limited to **time-invariant operators with fading memory**
- Persistent memory is of interest too, or limit cycles for locomotion
- Another theory is needed!
The theoretical model is limited to time-invariant operators with fading memory.

Persistent memory is of interest too, or limit cycles for locomotion.

Another theory is needed!

Based on feedback linearization from control theory.
Second Theoretical Model

Hauser et al.
"The role of feedback in morphological computation with compliant bodies" Biological Cybernetics, 2012
Second Theoretical Model

\[ u(t) \]

\[ z(t) = G(z(t), z(t)', \ldots, z(t)^{(n-1)}) + u(t) \]

Hauser et al.  
"The role of feedback in morphological computation with compliant bodies" Biological Cybernetics, 2012
Second Theoretical Model

\[ z(t) = G(z(t), z(t)', \ldots, z(t)^{(n-1)}) + u(t) \]

encodes our computation

Hauser et al.  
"The role of feedback in morphological computation with compliant bodies" Biological Cybernetics, 2012
Second Theoretical Model

\[ z(t) = G(z(t), z(t)', \ldots, z(t)^{(n-1)}) + u(t) \]

encodes our computation

Hauser et al.
"The role of feedback in morphological computation with compliant bodies" Biological Cybernetics, 2012
Second Theoretical Model

\[
\begin{align*}
\sum_{z(t)} &= G(z(t), z(t)', \ldots, z(t)^{(n-1)}) + u(t) \\
\end{align*}
\]

encodes our computation

emulation
Learning Setup
Learning Setup

... computation we want to emulate (Black Box)
Learning Setup

computation we want to emulate (Black Box)
Learning Setup

A computation we want to emulate (Black Box)

Target output
Learning Setup
Learning Setup

collect all data points over time

target output
Learning Setup

- collect all data points over time
- noise

- target output
Learning Setup

calculate optimal weights

target output
Learning Setup
Results

Input Dependent Limit Cycle

\[ x'_1 = x_1 + x_2 - \varepsilon x_1 (x_1^2 + x_2^2) \]
\[ x'_2 = -2x_1 + x_2 - x_2 (x_1^2 + x_2^2) \]
Input Dependent Limit Cycle

\[ \begin{align*}
    x'_1 &= x_1 + x_2 - \varepsilon x_1 (x_1^2 + x_2^2) \\
    x'_2 &= -2x_1 + x_2 - x_2 (x_1^2 + x_2^2)
\end{align*} \]
Results

Input Dependent Limit Cycle

\[ x'_1 = x_1 + x_2 - \varepsilon x_1 (x_1^2 + x_2^2) \]

\[ x'_2 = -2x_1 + x_2 - x_2 (x_1^2 + x_2^2) \]
Results

Input Dependent Limit Cycle

\[ x'_1 = x_1 + x_2 - \varepsilon x_1 (x_1^2 + x_2^2) \]
\[ x'_2 = -2x_1 + x_2 - x_2(x_1^2 + x_2^2) \]

![Graph showing the limit cycle with different values of \( \varepsilon \).](image)

![Plot showing oscillatory behavior of \( x_1 \) and \( x_2 \) over time.](image)
**Results**

Input Dependent Limit Cycle

\[
x'_1 = x_1 + x_2 - \varepsilon x_1 (x_1^2 + x_2^2)
\]

\[
x'_2 = -2x_1 + x_2 - x_2 (x_1^2 + x_2^2)
\]
Input Dependent Limit Cycle

\[ x'_1 = x_1 + x_2 - \varepsilon x_1 (x_1^2 + x_2^2) \]
\[ x'_2 = -2x_1 + x_2 - x_2 (x_1^2 + x_2^2) \]
Input Dependent Limit Cycle

\[ x'_1 = x_1 + x_2 - \varepsilon x_1 (x_1^2 + x_2^2) \]
\[ x'_2 = -2x_1 + x_2 - x_2 (x_1^2 + x_2^2) \]
Input Dependent Limit Cycle

\[ x'_1 = x_1 + x_2 - \varepsilon x_1 (x_1^2 + x_2^2) \]
\[ + x_2 - x_2 (x_1^2 + x_2^2) \]

one set of weights for all three \( \varepsilon \)
Results

Input Dependent Limit Cycle

- $x_1$ and $x_2$ vs. time [s]
- $\varepsilon = 5$
- $\varepsilon$ values: 0.2, 1, 5

Graphs show the behavior of $x_1$ and $x_2$ over time with different $\varepsilon$ values.
Results

Input Dependent Limit Cycle

![Graph showing input dependent limit cycle with time [s] and x1, x2 axes, and different values of \( \varepsilon \).]
Results

Input Dependent Limit Cycle
Discussion
Discussion

\[ \sum \text{constant} \]
Discussion

squeezing

constant
Discussion

squeezing

constant
Discussion

squeezing

constant
Discussion

squeezing

constant

\[ \sum \]
System is able to sense through its morphology!
System is able to sense through its morphology!
System is able to sense through its morphology!
System is able to sense through its morphology!
Application in Soft Robotics
Application in Soft Robotics

Kohei Nakajima
Tao Li
Application in Soft Robotics

soft (passive) silicone structure

Kohei Nakajima
Tao Li
Application in Soft Robotics

water tank

soft (passive) silicone structure

Kohei Nakajima

Tao Li
Application in Soft Robotics

10 bending sensors

soft (passive) silicone structure

water tank

Kohei Nakajima

Tao Li
Application in Soft Robotics

- Rotational motor
- Water tank
- Soft (passive) silicone structure
- 10 bending sensors

Kohei Nakajima
Tao Li
APPLICATION IN SOFT ROBOTICS
APPLICATION IN SOFT ROBOTICS
Application in Soft Robotics
Application in Soft Robotics

motor signal
Application in Soft Robotics

- exploitation of the body
Application in Soft Robotics

- exploitation of the body
- nonlinear and memory

motor signal
Application in Soft Robotics

- exploitation of the body
- nonlinear and memory
- noise comes from the sensors
Application in Soft Robotics

- exploitation of the body
- nonlinear and memory
- noise comes from the sensors
- sensors, water and even motor is part of the morphology
Application in Soft Robotics

- exploitation of the body
- nonlinear and memory
- noise comes from the sensors
- sensors, water and even motor is part of the morphology
- able to produce robust limit cycle
APPLICATION IN SOFT ROBOTICS
Take Home Messages
Take Home Messages

1. Outsourcing computation to the physical body facilitates computation/control
Take Home Messages

1. Outsourcing computation to the physical body facilitates computation/control

2. High-dimensionality, nonlinearity, compliance, and noise are your friends!
Take Home Messages

1. Outsourcing computation to the physical body facilitates computation/control

2. High-dimensionality, nonlinearity, compliance, and noise are your friends!

3. Computational exploitation is a way to control soft robots (without “knowing” their body)
Thank you very much for your Attention!

Rolf Pfeifer       Rudolf M. Füchslin     Auke Ijspeert     Wolfgang Maass

Kohei Nakajima     Tao Li
Bibliography


EU project OCTOPUS, http://www.octopusproject.eu/
EU project LOCOMORPH, http://www.locomorph.eu/