Bridging the Reality Gap: Transferring Robots from Simulation to Reality

Josheta Srinivasan[1 and 3], Eduardo J. Izquierdo[1 and 2], Derek Whitley[1]

INDIANA UNIVERSITY CENTER OF EXCELLENCE FOR WOMEN & TECHNOLOGY

[1]Cognitive Science Program, IU College of Arts and Science; [2]Department of Informatics, IU Luddy School of Informatics, Computing, and Engineering; [3]Department of Intelligent Systems Engineering, IU Luddy School of informatics, Computing, and Engineering

LUDDY SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

Introduction

Ever worried that we might be living in a simulation? Fret not, a recently proposed rebuttal[1] empirically proves that the richness of reality is downright impossible to simulate. This raises a paradox that is at the heart of this research:

The Reality Gap : the gap between simulation and real-life.

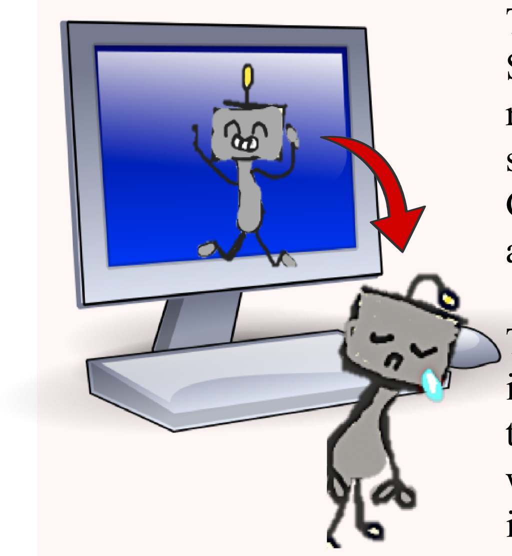

Fig 1: Illustration of the Reality Gap

This gap poses a significant problem in robotics. Simulations remain essential in training many robot behvaiours. They are cheaper, faster and soemtimes more effective. Yet, given the Reality Gap, robots trained in simulation don't perform as well in real-life as they do in simulation.

The most common approach to solve this issue is often some variant of adding noise to the training. Robots can then be trained to cope with noisy environments, which the real world is. Yet, a systematic study of just how one should do so is less explored. Where should we add noise? How much? What type?

Research Question

How does the **amount, location and type of noise added** to a simulator that optimizes a robot called the Single-Legged Walker affect it's **ability to cross The Reality Gap**?

Preliminary Hypothesis

Walkers trained with noise would develop behaviours that aren't highly specialized to their training environment as the added noise keeps changing the environmental parameters.

As such, we hypothesized that

1. Walkers trained with noise would better transfer their behvaiour to reality compared to those trained without.
2. Walkers trained with noise would be more robust (i.e. retain behavioural performance despite noisy input) than those trained without.
3. More robust walkers would be better at crossing the Reality Gap.

What makes a Robot?

We need to answer this to be able to model the Walker in simulation, only after which can we start optimization.

A Dynamical System[2] is all about how 'something' changes over time. To define a dynamical system, we need to (1) define what this 'something' is: i.e a **collection of states** and (2) how it changes over time: **rules that map some input to change in said states**.

This research views a robot as a coupling of two dynamical systems: the Environment and the Walker robot. As such, modelling the Walker required a mathematical model of:

- **The Environment:** A collection of 'Environmental states'
- **The Walker:** A collection of 'Internal states'
- **Sensory mapping:** a rule that maps changes in Environmental states to changes in Internal states.
- **Motor mapping:** a rule that maps changes in Internal states to changes in Environmental states

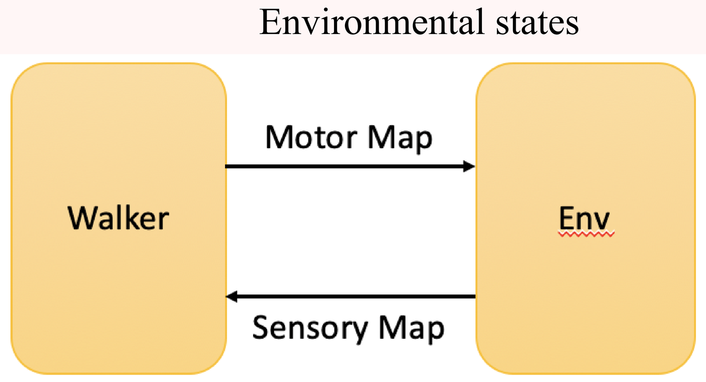

Fig 2: Illustration of the coupled Dynamical System used to model and simulate the walker

Meet the Single-Legged Walker!

The Walker was then modelled in simulation and comprised of:

Brain: Implements the rules that map (1) sensory inputs to changes in internal states and (2) changes in internal states to motor ouputs. I used a Continuous-Time Recurrent Neural Network to create the brain of the Walker.

Body: In simulation, this is a collection of internal states such as leg angle, leg length, foot state etc. (i.e. it is a bunch of code.) In real-life, this is a mechanical construction.

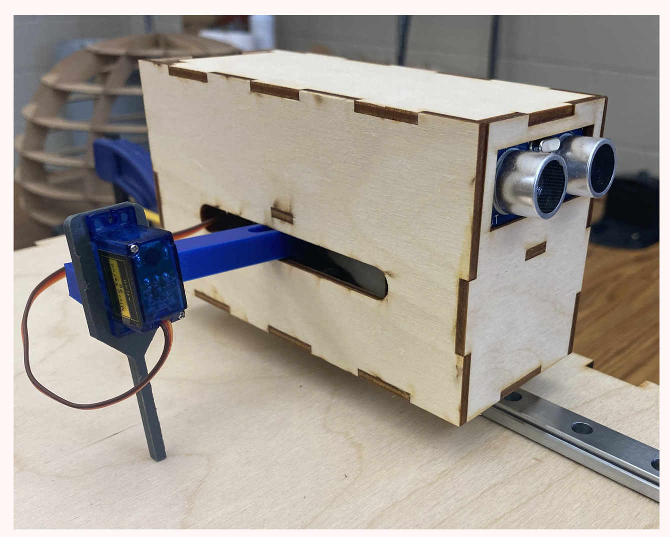

Fig 3: Mechanical Construction of the Walker

```python
import math
import numpy as np

# Constants
MaxLegForce = 0.05
ForwardAngleLimit = math.pi/6.0
BackwardAngleLimit = -math.pi/6.0
MaxVelocity = 6.0
MaxTorque = 0.5
MaxOmega = 1.0

class LeggedAgent:

    def __init__(self):
        self.cx = 0.0
        self.cy = 0.0
        self.vx = 0.0
        self.footstate = 0
```

Fig 4: Snippet of code that simulates body of Walker

Environment: In simulation, this is a collection of environmental states such as force exerted on feet, etc. (again a bunch of code). In real-life, this is simply everything around the robot walker.

Preliminary Methodology

Preliminary Conditions

1. Control: Optimization without any noise (NN: No Noise)
2. Noise to Omega - Leg Angular Velocity (WNO: With Noise Omega)
3. Noise to the parameters of the Neural Network (WNP: With Noise Parameter)

Optimization in Simulation

- an Evolutionary Algorithm[3] was used for optimzation: Essentially mimics biological evolution - Neural Networks are innitialized and ones that don't perform well are weeded out over time: Survival of the Fittest style.
- **10** evolutionary runs for each condition --> **10** x **3** total optimized robots

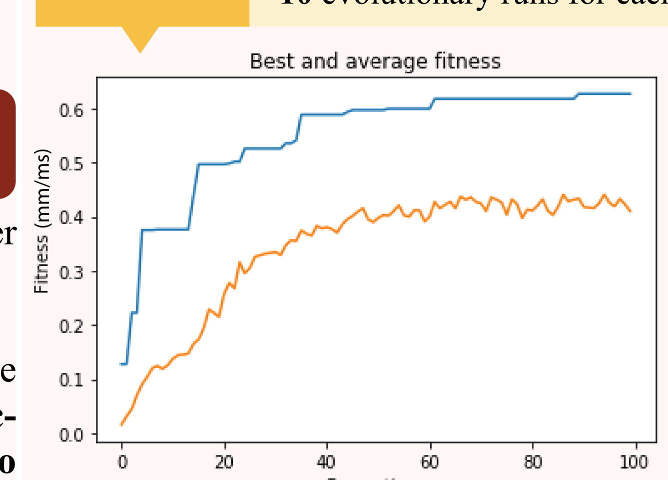

Fig 5: Visualization of the Evolution of the robot Controllers. Note how fitness of robot increases with generations (which is basically a measure of time)

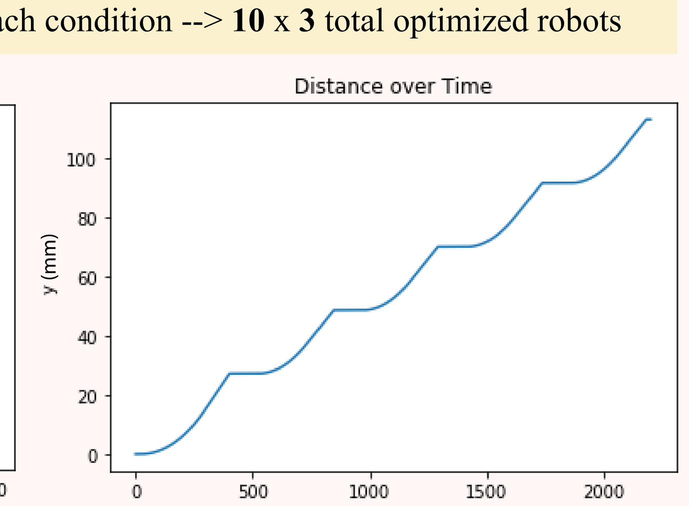

Fig 6: Visualization of the the position of the robot with time as it walks. Note the rhythmic walking dynamic that is observable

Equalizer

- Gets a better measure of the performance value (i.e. Fitness) of the Walkers (average velocity) with longer walking duration (2000 ms).
- Standardizes the environment where Fitness is meausured across conditions.

Isolator

- Qualitatively determined the threshold for 'good' performing solution (> 0.58 mm/ms)

Robustness test in Simulation

- Tested Walker ability to maintain performance when placed in a different environment to where they were trained (noise added to Leg Force).
- Still in simulation: serves as an indicator of which solutions might perform better in real life.

Performance test in Real Life

- Test the evolved control mechanism (i.e. Neural Networks) with the real-life Walker and compare Fitness value with simulated counter-part.

Preliminary Results

Results showed that the No-Noise conditon (labelled NN) was more robust (flatter curve in **Fig 7**) than the Omega-Noise condition (WNO) and the Neural-Network Noise condition (WNP).This was counter-intuitive and seemed to oppose my hypothesis, prompting a deeper dive into what happened during evolution (**Figs 8 and 9**).

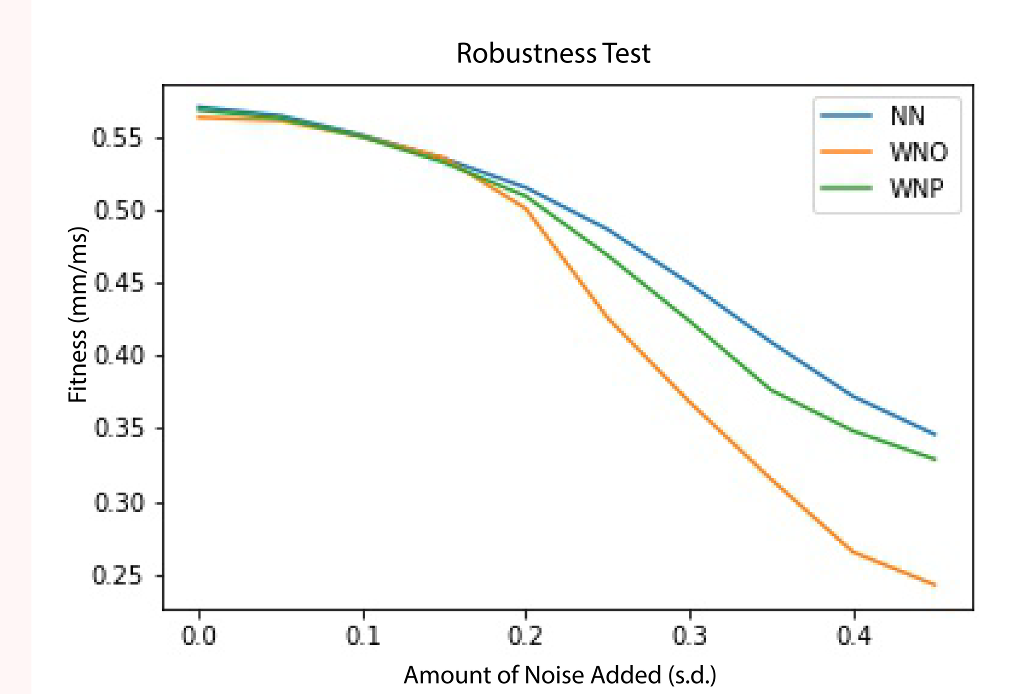

Fig 7: Robustness test results - plot of Fitness over Noise added

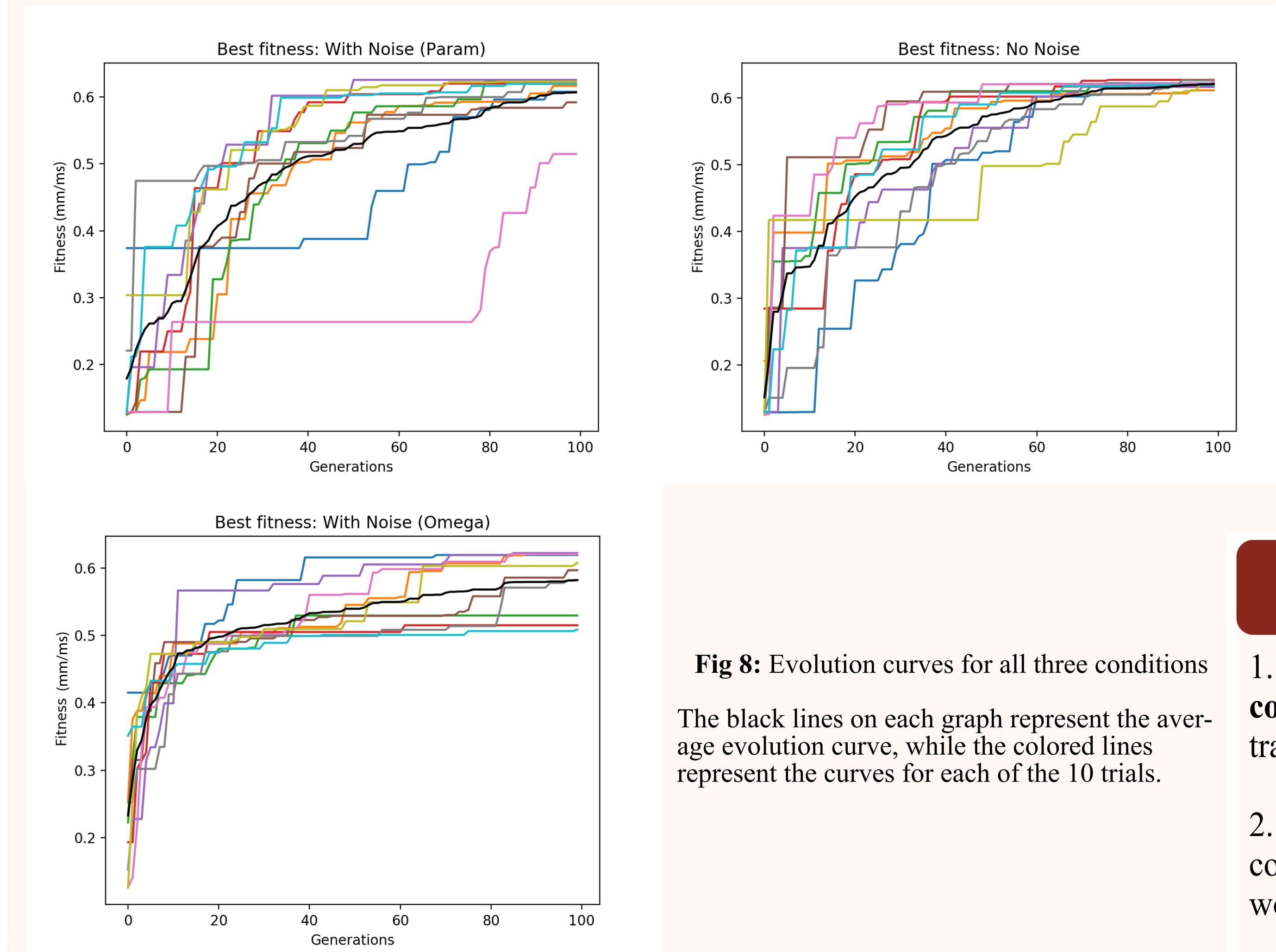

Fig 8: Evolution curves for all three conditions

The black lines on each graph represent the average evolution curve, while the colored lines represent the curves for each of the 10 trials.

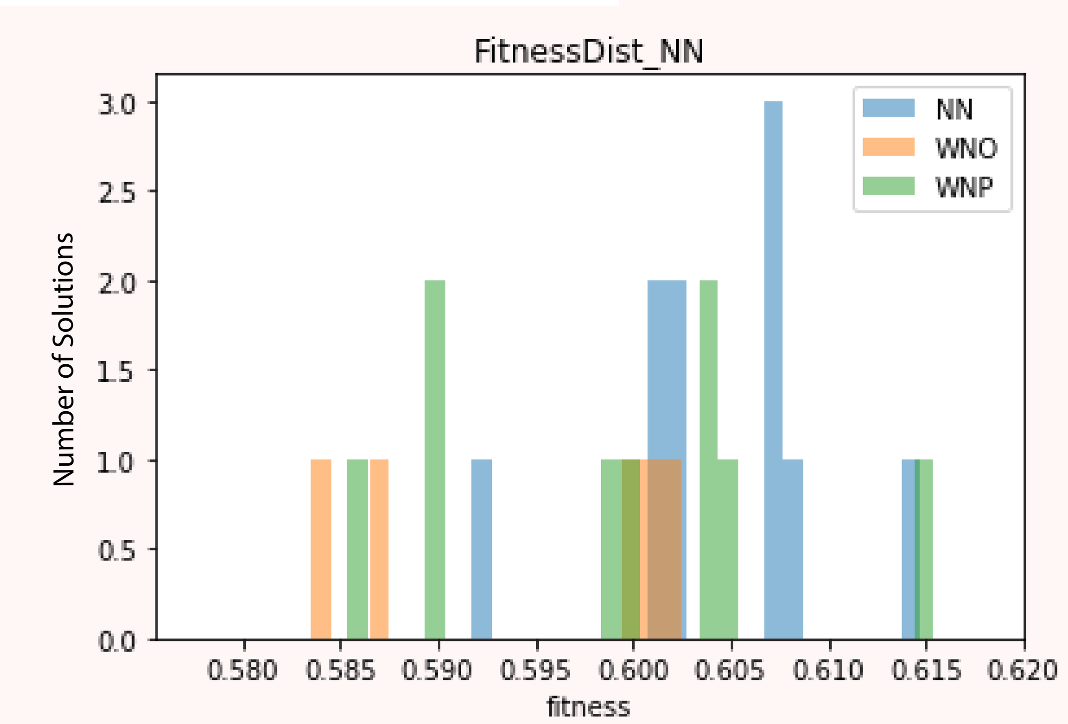

Fig 9: Fitness Distribution of the three conditions after isolation

What I found was that

1. The NN condition had all 10 trials evolve to a 'good' (>0.58) fitness level; the WNP condition had 9/10 trials do so and the WNO condition had 5/10 do so.
2. The NN condition (blue in **fig 9**) had a greater percentage of its Walkers at higher fitness values compared to the noisy conditions (orange and green in **fig 9**)

Some Immediate Takeaways

1. There seems to exist a tradeoff between spending optimization energy getting good at the behaviour (walking) and trying to generalize behaviour to the noisy environmental and internal conditions (i.e. developing ways to maintain behvaiour that isn't dependent on these conditions)

2. The tradeoff results in an uneven fitness distribution at the end of optimization.

3. A characteristic of this trade-off and uneven fitness distribution is that some Walkers evolve to be so good at walking that they retain their performance in the face of noisy conditions much better than the walkers specifically trained to be able to do so.

Next Steps

1. Evolve for longer (250 generations instead of 100) to ensure all conditions **converge** to a good and similar fitness level. (Hence, mitigating the issues of the trade-off metioned in the previous section)

2. Finish construction of the real-life Walker and test real-life performance. I've completed the design and construction of the overall outer structure. Next step would be to wire up the parts and integrate them to evolved Neural Networks.

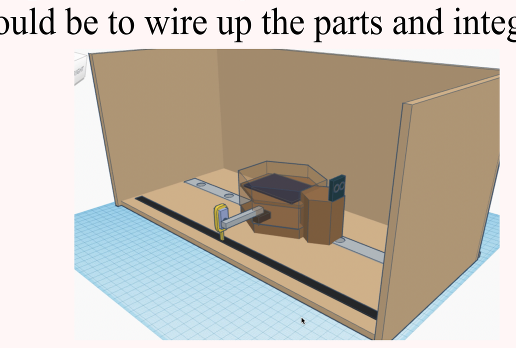

Fig 10: 3D design of real-life walker

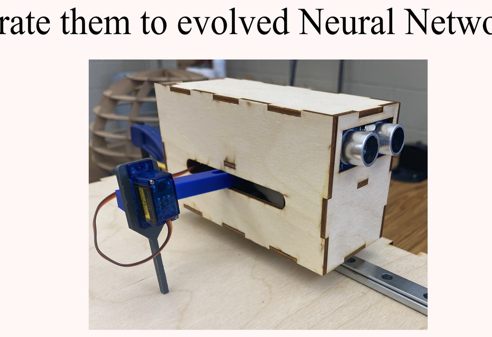

Fig 11: Outer-Construct of the real-life walker

3. Test Hypothesis 3 by comparing robustness test results and real-life test results: is the robustness test I have created indeed a good indication of reality gap crossing ability?

4. Use results to further perfect methodology upon which I can **permute conditions** and carry out the **systematic study** that I started out with the purpose of doing.

Conclusion and Significance

Preliminary results I have collected suggest that No-Noise conditions are more robust to changes to environmental states than the Noise conditions.

If after tweaking the methodology and conducting the real-life test (as described in the 'Next Steps' section), the No-Noise condition continues to be more robust and transferable - given that several[4,5] other papers show training with noise increasing robustness and transferability - it suggests that the location, type and amount of noise added really does matter: i.e. there is a **sweet-spot collection** of these parameters where transferability and robustness is boosted by noisy training.

This makes my Research Question all the more pertinent and useful in the conscientious design of simulations that ensure better behaviour tranferance to reality.

Acknowledgement and References

I would like to thank: Eduardo J. Izquierdo for his mentorship; and Derek Whiteley and Mathew Francisco for their guidance and resources for constructing the real robot.

[1] McDonald, Glenn. "We Are Not Living In A Simulation. Probably." Fast Company, Fast Company, 14 Mar. 2018, www.fastcompany.com/40537955/we-are-not-living-in-a-simulation-probably [2] "Math Insight." The Idea of a Dynamical System - Math Insight, mathinsight.org/dynamical_system_idea. [3] Beer, Randall D. "The Dynamics of Adaptive Behavior: A Research Program." Robotics and Autonomous Systems, vol. 20, no. 2-4, 1997, pp. 257–289., doi:10.1016/s0921-8890(96)00063-2. [4] Beer, Randall D., and John C. Gallagher. "Evolving Dynamical Neural Networks for Adaptive Behavior." Adaptive Behavior, vol. 1, no. 1, 1992, pp. 91–122., doi:10.1177/105971239200100105. [5] Lyttle, David N., et al. "Robustness, Flexibility, and Sensitivity in a Multifunctional Motor Control Model." Biological Cybernetics, vol. 111, no. 1, 2016, pp. 25–47.,doi:10.1007/s00422-016-0704-8.