A Structural Approach to Dealing with High Dimensionality Parameter Search Spaces

Benjamin Hawker(✉) and Roger K. Moore

Department of Computer Science, University of Sheffield, Sheffield, UK
{ben.hawker,r.k.moore}@sheffield.ac.uk

Abstract. In the field of robotics, searching for effective control parameters is a challenge as controllers become more complex. As the number of parameters increases, the dimensionality of the search problem causes results to become varied because the search cannot effectively traverse the whole search space. In applications such as autonomous robotics, quick training that provides consistent and robust results is key. Hierarchical controllers are often employed to solve multi-input control problems, but multiple controllers increases the number of parameters and thus the dimensionality of the search problem. It is unknown whether hierarchies in controllers allows for effective staged parameter optimisation. Furthermore, it is unknown if a staged optimisation approach would avoid the issues high dimensional spaces cause to searches. Here we compare two hierarchical controllers, where one was trained in a staged manner based on the hierarchy and the other was trained with all parameters being optimised at once. This paper shows that the staged approach is strained less by the dimensionality of the problem. The solutions scoring in the bottom 25% of both approaches were compared, with the staged approach having significantly lower error. This demonstrates that the staged approach is capable of avoiding highly varied results by reducing the computational complexity of the search space. Computational complexity across AI has troubled engineers, resulting in increasingly intense algorithms to handle the high dimensionality. These results will hopefully prompt approaches that use of developmental or staged strategies to tackle high dimensionality spaces.

Keywords: Hierarchical · Control · Developmental · Learning

1 Introduction

Optimising parameters in functions is key to tailoring their competences to a problem. The more parameters to optimise puts strain on the chosen search algorithm. Eventually, a sufficiently challenging search will result in inconsistent results from the search algorithm, or no success at all [18]. In the field of robotics, this has limitations and challenges particularly in robots searching

A. Mohammad et al. (Eds.): TAROS 2020, LNAI 12228, pp. 159–170, 2020.
https://doi.org/10.1007/978-3-030-63486-5_19

for parameters autonomously. Many of the solutions rely heavily on kinematic and mechanical information that is implicitly or explicitly applied to the search algorithm to minimise the complexity of the search. Such information is not always available without an expert in the particular mechanical objective being learned. Furthermore, this information can vary greatly based on subtle properties of the robot or environment. While rewarding, the process of acquiring and implementing a lot of these priors is demanding. Furthermore, there are many domains where this approach isn't feasible due to the required information or expertise being unavailable. Being able to learn a problem without expressed detail of the problem is a valuable skill for autonomous agents to have.

1.1 Alternative Approaches

Model-based approaches solve the issue of complex search spaces through exhaustive search. Models can often be evaluated quicker than the robot can run in real time. This allows rudimentary algorithms to brute-force search with many trials in order to find a suitable solution [19]. However, exact models of a particular robot, environment and task are not always available. To build these have thorough knowledge of the robot and environment. Even then, it is easy to forget key details of the problem resulting in the parameters requiring manual tuning afterwards to optimise. Any time saved by making a simpler model places the engineer in a situation later where extra effort is required to manually tune the parameters to better fit the problem.

Machine Learning has a variety of approaches that generalise the kinematic properties in an environment. The extrapolation employed by the statistically based approaches allows inferences to be drawn about the search space, giving success in parameter selection where the parameters have generalisable or predictable behaviour [9–11]. However, approaches can require many trials in order to be successful. Modifications have been developed to improve the search to allow better generalisation in a limited number of trials. Approaches that succeed with consistent results in a handful of trials exist, but often require heavily informative priors or sensitive selection of key meta-parameters to guide the search algorithm [2]. This can be in the explicit model of the problem, or implicitly via a policy which guides the search to suit a particular demographic of problem. Again, these require an expert on the agent's environment who must select or build an informed policy or model. A "general policy" with which to solve robotic kinematic problems is not available due to the diversity among robots and environments.

1.2 Hierarchical Control

Hierarchical Control as a field has considered developmental approaches to optimisation [1,4,13]. Fields such as Perceptual Control Theory have noted that optimisation of higher levels of a hierarchy requires the lower levels to function [14]. What remains untested is whether the hierarchy is an indicator of which parameters can be optimised independent of the others. Can each level of the hierarchy, starting at the bottom, be optimised independent of what comes above it in the hierarchy? Whether this has been done has not been tested. Furthermore, if this is possible, it is not clear if this approach avoids the downsides that increased dimensionality causes.

1.3 Summary

This requirement of expert knowledge to minimise complexity presents limitations in autonomous robotics. Furthermore, autonomous robots have a restricted number of trials with which to find a new parameter set. New methodological approaches that aid reducing the complexity of searches would benefit autonomous robots.

This paper describes an approach to the problem based on hierarchical control and staged optimisation of parameters. An experiment was conducted in order to show whether the staged approach suffers less from inconsistent results which is a common effect of dimensionality issues.

2 Experimental Setup

2.1 Baxter Robot

The experiment was conducted with the Baxter Robot, a six foot 14-DOF industrial robot. The task concerned the left arm, specifically the joint shown on the left of Fig. 1, named s0. This joint rotates the arm along the X-Z plane. The rest of the arm was held in the position shown in the picture on the left in Fig. 1, so the controller could consistently achieve control.

The task was to control the angular position of the elbow (e1) with respect to the shoulder joint (s0) in the X-Z plane. Applying force in either direction of the s0 joint moves the arm around Baxter, changing the angle between s0 and e1 as indicated in the right panel of Fig. 1.

2.2 PID Cascade Control

A Proportional-Integral-Derivative Controller (also referred to as a PID Controller) is a negative-feedback controller widely used within control systems engineering due to the simplicity and effectiveness of control provided [5].

A negative feedback controller controls a particular external variable by continuously minimising error, where error is defined as the difference between the actual value and the desired value for the controlled variable [20]. If e is the error, then the control process can be defined as:

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \tag{1}$$

where $u(t)$ is the control output at time t, $e(t)$ is the error at time t and k_p, k_i and k_d are parameters. The original inspiration was from manual control of steering

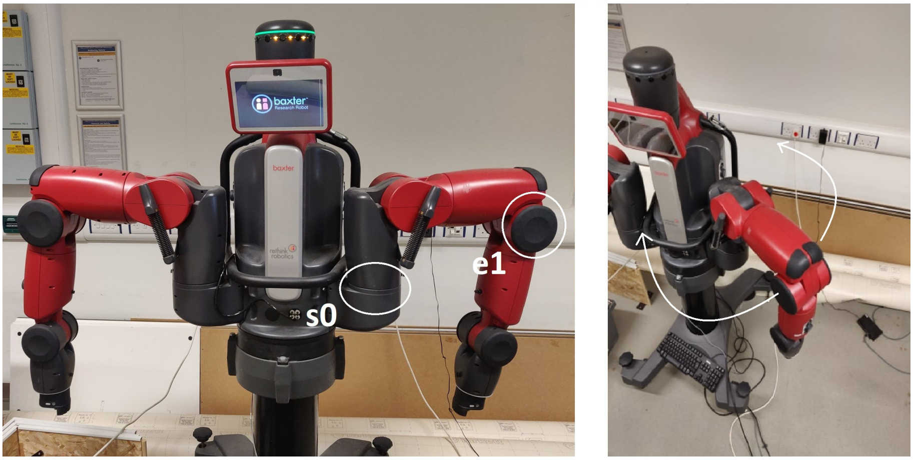


Fig. 1. A pair of images of the Baxter Robot. The left image shows the whole robot, the controlled joint (s0) and the location that was being controlled (e1) through moving s0. The right image shows the effect of applying force to the s0 joint, either positive or negative.

ships, where it was realised that a sailor would not just aim to minimise error proportionally but also aim to account for lingering error and avoid large rates of change [12]. The elegant and simple design affords utility while being Bounded-Input Bounded-Output Stable, making the general responses predictable.

Cascading PID Control (also known as Cascade Control) refers to two (or more) PID controllers where the reference signal for one PID controller is the control output (u) from the higher controller. Cascade control is used for many control applications in recent literature both as is [17] and with modifications [3,16].

2.3 Control System for This Experiment

For this experiment, a cascading PID controller was employed to control Baxter's inner shoulder joint (known as s0) to position the elbow at a particular angular position. The higher order controller controlled the angular position of the elbow, sending signals to the lower controller which controlled the velocity of the s0 joint. The lower controller sent a control signal applying torque to the joint. The controller is shown in Fig. 2.

2.4 Bat Algorithm

Evolutionary Algorithms, inspired by the Genetic Algorithm, benefit from good convergence in a small amount of trials. Evolutionary Algorithms are inspired by patterns noticed in nature, where Bat Algorithm is inspired by the echo-location used by bats to search an area for possible prey [21]. These properties have made

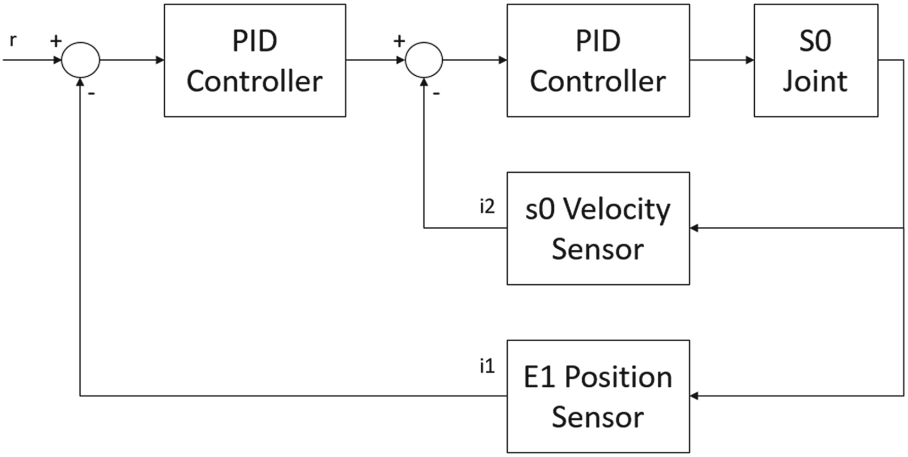


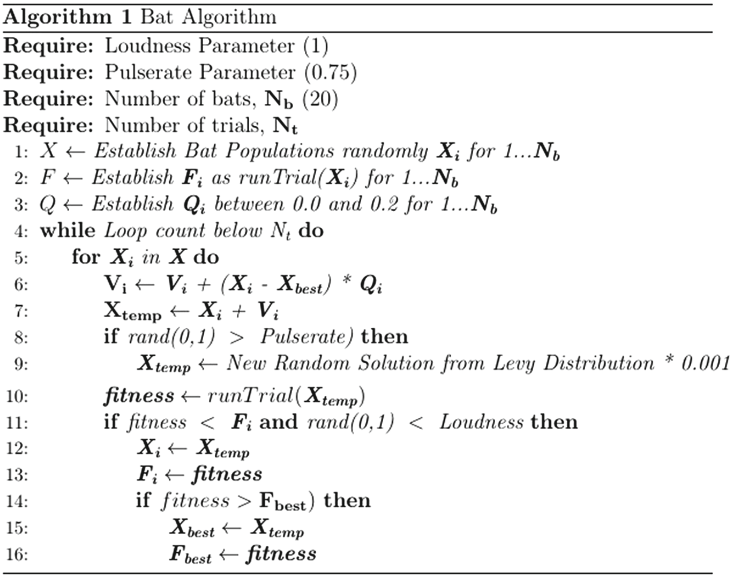
Fig. 2. A diagram showing the cascading PID controller used in this experiment.

the Bat Algorithm useful in control of robots [15] and more generalised AI tasks such as path planning [8].

The variant of the algorithm used in this experiment extends Yang's work. A velocity based approach to updating the candidates [6,7] and a levy-flights based random walk are utilised. The algorithm optimised candidates to minimise error on the staged and all-in-one curriculums, with 30 iterations in total (which were divided equally between the two training stages in the staged approach). See Fig. 3.

2.5 Designing Curricula for Developmental Learning

Two curricula were developed for learning the problem. One expressed the higher level problem of controlling the angular position, which both approaches used. The staged curriculum also trained the lower controller on how to control the velocity of the s0 joint. For each curriculum, the average error over each task is the score. A curriculum could be built based on a particular task where the candidate simply passes or fails. This is realistic to the environment, as often a difference between average error is not important as long as the candidate passes the task. However, pass or fail tasks are usually domain specific. Average error, while not necessarily indicative of passing or failing, implicitly tests important properties of a controller. The rise time, settling time, overshoot and steady state error all impact the average error and are four important properties which one would test in a domain specific environment. Therefore, average error suffices as a good indicator of improving performance. Modifying the curriculum to account for particular properties would be simple to do, if knowledge of the domain is provided to indicate which of the four properties is most important to control.



```
Algorithm 1 Bat Algorithm
Require: Loudness Parameter (1)
Require: Pulserate Parameter (0.75)
Require: Number of bats, $\mathbf{N_b}$ (20)
Require: Number of trials, $\mathbf{N_t}$
$X \leftarrow$ Establish Bat Populations randomly $\boldsymbol{X_i}$ for 1...$\boldsymbol{N_b}$
$F \leftarrow$ Establish $\boldsymbol{F_i}$ as runTrial($\boldsymbol{X_i}$) for 1...$\boldsymbol{N_b}$
$Q \leftarrow$ Establish $\boldsymbol{Q_i}$ between 0.0 and 0.2 for 1...$\boldsymbol{N_b}$
while Loop count below $N_t$ do
    for $\boldsymbol{X_i}$ in $\boldsymbol{X}$ do
        $\mathbf{V_i} \leftarrow \boldsymbol{V_i} + (\boldsymbol{X_i} - \boldsymbol{X_{best}}) * \boldsymbol{Q_i}$
        $\mathbf{X_{temp}} \leftarrow \boldsymbol{X_i} + \boldsymbol{V_i}$
        if rand(0,1) $>$ Pulserate) then
            $\boldsymbol{X_{temp}} \leftarrow$ New Random Solution from Levy Distribution * 0.001
        $\boldsymbol{fitness} \leftarrow runTrial(\boldsymbol{X_{temp}})$
        if fitness $<$ $\boldsymbol{F_i}$ and rand(0,1) $<$ Loudness then
            $\boldsymbol{X_i} \leftarrow \boldsymbol{X_{temp}}$
            $\boldsymbol{F_i} \leftarrow \boldsymbol{fitness}$
            if $fitness > \mathbf{F_{best}}$) then
                $\boldsymbol{X_{best}} \leftarrow \boldsymbol{X_{temp}}$
                $\boldsymbol{F_{best}} \leftarrow \boldsymbol{fitness}$
```

Fig. 3. The algorithm employed in this experiment, inspired by Fister's velocity adaptations of the Bat Algorithm [6].

Top Level: Position Control. The position curriculum had three trials that the candidates were tested on. Between each of these trials, the controller and position of the robot were reset. The reset point was the middle point of the range of movement, which is approximately 40°. The error over time for all three trials was recorded and averaged.

- Move to 5°, 8 s time limit
- Move to 55°, 8 s time limit
- Move to 95°, 8 s time limit

Bottom Level: Velocity Control. The Velocity Control curriculum was designed as one continuous trial, so changes in behaviours are accounted for in the curriculum. The agent began at the middle point as before, but then each of these tests immediately moved onto the next. Again, the average error over the whole period was the score for those parameters.

- Maintain a velocity of -0.3 m/s until past $-10°$.
- Maintain a velocity of 0 for 3 s.
- Maintain a velocity of 0.6 m/s until past 110°.
- Maintain a velocity of -0.6 m/s until past $-10°$.
- Maintain a velocity of 0.3 m/s until past 110°.
- Maintain a velocity of 0 for 4 s.

2.6 The Full Architecture

An overarching control program assigns which optimisation approach the Bat Algorithm will use, staged or all in one, as well as the number of trials to be run. The Bat Algorithm produces possible parameter combinations (hereafter called candidates) which need to be tested. When one needs testing, it is sent to the curriculum trial controller, which tests the candidate on the curriculum through a series of control tasks. On receiving a candidate to test, the curricula trial controller will set the parameters of the Cascading PID Controller to those of the candidate. Then, it passes reference signals to the Cascading PID Controller for each control task. It will keep doing this until all control tasks that are part of this curriculum have been sent. Once the Cascading PID Controller receives reference signals for a control task, the Cascading PID Controller sends control signals to the robot which returns sensory feedback. From this feedback, the Cascading PID Controller calculates the average error over the period of the control task. This average error is fed back to the curricula trial controller, which then averages the average error across all the control tasks. This is fed back to the Bat Algorithm, which feeds into whether this candidate should be kept or discarded. Eventually, when all the trials are complete, the Bat Algorithm feeds back to the overarching control program the best candidate at minimising average error.

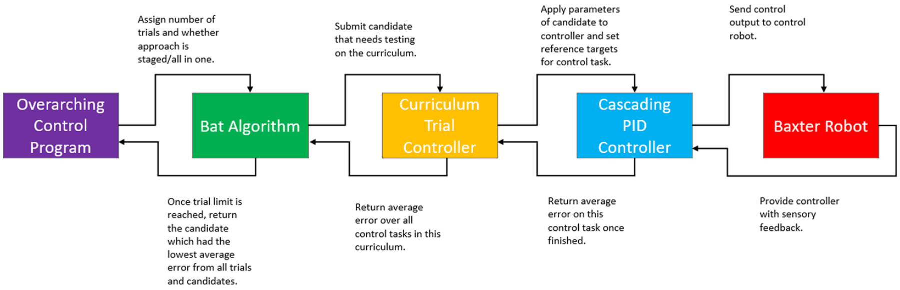

Assign number of trials and whether approach is staged/all in one.
Submit candidate that needs testing on the curriculum.
Apply parameters of candidate to controller and set reference targets for control task.
Send control output to control robot.
Overarching Control Program
Bat Algorithm
Curriculum Trial Controller
Cascading PID Controller
Baxter Robot
Once trial limit is reached, return the candidate which had the lowest average error from all trials and candidates.
Return average error over all control tasks in this curriculum.
Return average error on this control task once finished.
Provide controller with sensory feedback.

Fig. 4. A Flow Chart showing the program flow of the combined architecture. Each arrow indicates some information or a command being sent from one part of the architecture to another. (Color figure online)

3 Experimental Results and Discussion

3.1 Execution Time

Due to the size of Baxter and the heavy weight of the limbs, each test on the curriculum required 20 to 30 s. With 20 trials and 20 candidates, this results in a running period of several hours, which is not suitable across all robotics solutions. However, in each run of the algorithm, effective candidates were found in the first two to four trials. Each staged approach took only two to four trials to acquire a candidate that was below or equal to 110% of the average error of the eventually found best candidate. For the all-in-one approach, this was between four and eight trials. This presents a quicker time frame than the maximum number of trials used, but is important to test the effectiveness in situations where greater time is allowed. Furthermore, many autonomous robots will be able to act faster than Baxter, whose joints are not built to be quick or responsive. With a robot which enacts trials quicker combined with the low number of trials required, this reduces the time to be effective from hours to minutes.

3.2 Comparison of the Chosen Parameters

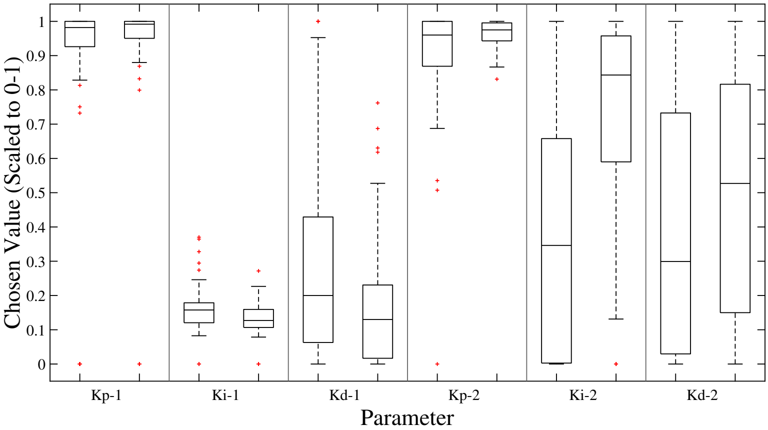


Fig. 5. A graph showing the spread of choices for the six parameters chosen by each approach. For each parameter labelled on the x-axis, there are two boxplots representing the spread of parameter values chosen. The middle line represents the mean, the box's upper and lower bounds represent the 75th and 25th percentile respectively, and the upper and lower whiskers are the upper and lower adjacent values respectively. The left box in each section indicates the chosen values by the all-in-one approach and the right box represents the values chosen by the staged approach. The most notable difference is the choice of K_i in the velocity controller, Ki-2, where the staged approach went for an integral-heavy parameter set.

The Staged Approach had a separate training procedure for the three parameters in the lower controller. However, the values chosen for the lower controller influenced the choices of the second stage of training. Given this, it is notable that both approaches found similar parameters for the higher controller. This can be seen in the first three pairs of boxes and means (labelled kp-1, ki-1 and kd-1) in Fig. 5. For each pair in Fig. 5, the all-in-one approach has chosen parameters similar to the staged approach.

The most notable difference between the two schemes is in the K$_i$ value for the lower controller indicated by the third and fourth columns from the right in Fig. 5. The staged approach on average has a much higher K$_i$ value, whereas the all-in-one approach favours a lower value. The integral typically causes the controller to overcome steady state error which would be expected in a velocity controller. The amount of force required to counter a small error (or apply a small amount of velocity) is more than the proportional term would allow. As such, an integral is expected here to allow error to build and apply more torque to the joints. The slightly higher K$_d$ value is also expected as a result, as the K$_d$ value offsets the overshooting a high K$_i$ value can often cause (Fig. 6).

3.3 Comparison of Error

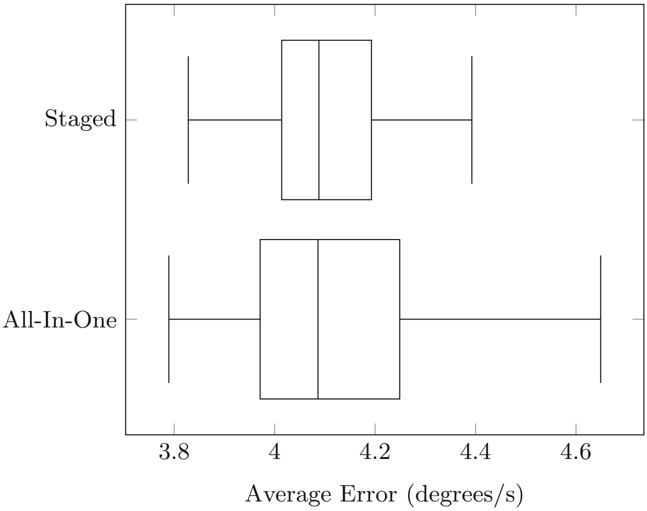


Fig. 6. Box Plots of the average error of the best solutions found by the all-in-one and staged approaches. The middle line represents the mean, the box's upper and lower bounds represent the 75th and 25th percentile respectively, and the upper and lower whiskers are the upper and lower adjacent values respectively.

Both medians are similar with no significant difference, but the spread of results differs. The all-in-one approach has a greater degree of both excellent and poor results further from the median. This is as hypothesised, as the higher dimensionality of the search space allows for all possible combinations to be considered. However, the dimensionality also increases the complexity of the search space. Given the initial candidates are randomly selected, these can be a poor selection from the state space and not allow the algorithm to appropriately minimise error (Fig. 6).

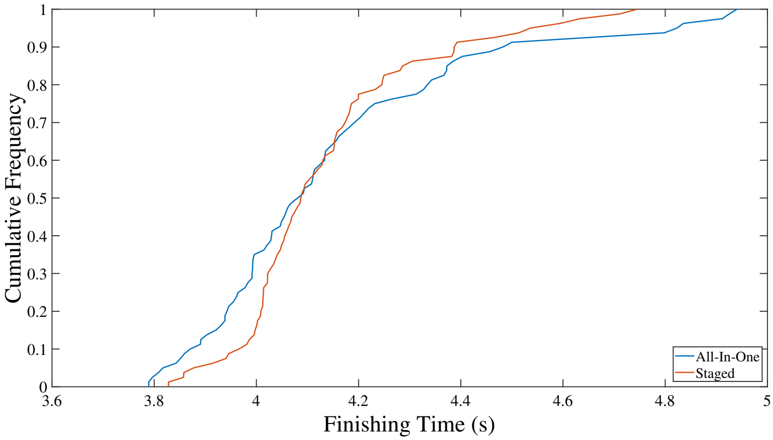


Fig. 7. A graph showing a Cumulative Distribution Function of the average error of the best solutions found by the all-in-one and staged approaches.

When comparing all the solutions and their scores from both approaches, neither has significantly lower error than the other. However, The poorest 25% of solutions from the staged approach perform significantly better than the poorest 25% of solutions from the all-in-one approach. The best 25% of the all-in-one solutions significantly outperform the best 25% of the staged solutions (Fig. 7).

This result is applicable in fields where consistent reoptimisation of parameters is preferred, as poor results can result in catastrophic failure and are not worth the occasionally better performances such as autonomous robotics. Furthermore, it is notable that the solutions do not have distinctly different medians given the staged approach is computationally simpler. Two three-dimensional search spaces are less complex to traverse than one six-dimensional search space, meaning the same results on average are being achieved on a simpler version of the problem. The staged approach is computationally simpler as it does not consider every possible combination of all six parameters, but rather optimising three independent of what values may be selected for the other three. This could theoretically limit the controller by not allowing it to find suitable parameter combinations between the higher and lower controller. However, given the medians are similar, this indicates that the poor results from the higher dimensionality offset the benefits of having access to more parameter combinations.

4 Conclusion and Future Work

In this paper, results have been presented comparing a staged parameter selection approach with the standard all-in-one approach for control of a joint in a robotic arm. It has been shown that the staged approach has more consistent results, particularly that the worst solutions of the staged approach are better than the worst solutions of the all-in-one approach. The staged approach is computationally simpler yet retains a similar median performance. The value of consistency in autonomous robotics has been discussed. However, what remains to be determined is the extent to which this effect would scale and how effective these candidates are in a general setting. This paper shows the effect of different parameter optimisations and how they're affected by dimensionality, but does not express how effective the controllers are per se. A set of trials aimed at testing generalised performance would need to be used for this.

It is not concluded whether the resistance to the effects of dimensionality in the staged approach scales to higher dimensions. When the algorithm struggles to search the space due to high dimensionality, inconsistency will occur. However, as the dimensionality continues to increase, the effectiveness of the solutions should worsen rapidly. Further work needs to be done to demonstrate how resistant the staged approach is in higher dimensional searches.

Finally, more formal work could be performed to detail exactly what a good curriculum is. It is evident that the curriculum designed here met the purpose of maintaining good results by achieving consistent staged parameter optimisation. However, further discussion and methodological analysis is necessary to identify what comprises an effective learning curriculum.

Acknowledgement. With thanks to Adam Hartwell and Jonathan Aitken, for their technical support and advice on matters of Control Engineering.

References

1. Brooks, R.: A robust layered control system for a mobile robot. IEEE J. Robot. Autom. **2**(1), 14–23 (1986). https://doi.org/10.1109/JRA.1986.1087032. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1087032
2. Chatzilygeroudis, K., Vassiliades, V., Stulp, F., Calinon, S., Mouret, J.B.: A survey on policy search algorithms for learning robot controllers in a handful of trials, July 2018. http://arxiv.org/abs/1807.02303
3. Deng, H., Li, Q., Cui, Y., Zhu, Y., Chen, W.: Nonlinear controller design based on cascade adaptive sliding mode control for PEM fuel cell air supply systems. Int. J. Hydrogen Energy **44**(35), 19357–19369 (2019). https://doi.org/10.1016/j.ijhydene.2018.10.180
4. Digney, B.L.: Learning hierarchical control structures for multiple tasks and changing environments. In: From Animals to Animats, vol. 5, pp. 321–330. MIT Press, September 1998. http://dl.acm.org/citation.cfm?id=299955.299998
5. DiStefano, J.J., Stubberud, A.R., Williams, I.J.: Theory and Problems of Feedback and Control Systems. McGraw-Hill, New York (1967)
6. Fister, I., Fister, D., Yang, X.S.: A hybrid bat algorithm, March 2013. http://arxiv.org/abs/1303.6310
7. Fister, I., Fong, S., Brest, J.: A novel hybrid self-adaptive bat algorithm (2014). https://doi.org/10.1155/2014/709738
8. Guo, J., Gao, Y., Cui, G.: The path planning for mobile robot based on bat algorithm. Int. J. Autom. Control **9**(1), 50–60 (2015). https://doi.org/10.1504/IJAAC.2015.068041

9. Lizotte, D., Wang, T., Bowling, M., Schuurmans, D.: Automatic gait optimization with Gaussian process regression - Proceedings of the 20th International Joint Conference on Artifical Intelligence. Technical report (2007)
10. Martinez-Cantin, R., de Freitas, N., Doucet, A., Castellanos, J.: Active policy learning for robot planning and exploration under uncertainty. In: Robotics: Science and Systems III. Robotics: Science and Systems Foundation, June 2007. https://doi.org/10.15607/RSS.2007.III.041. http://www.roboticsproceedings.org/rss03/p41.pdf
11. Matsubara, T., Hyon, S.H., Morimoto, J.: Learning parametric dynamic movement primitives from multiple demonstrations. Neural Netw. **24**(5), 493–500 (2011). https://doi.org/10.1016/j.neunet.2011.02.004
12. Minorsky, N.: Directional stability of automatically steered bodies. J. Am. Soc. Naval Eng. **34**(2), 280–309 (1922). https://doi.org/10.1111/j.1559-3584.1922.tb04958.x. http://doi.wiley.com/10.1111/j.1559-3584.1922.tb04958.x
13. Morimoto, J., Doya, K.: Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning. Robot. Auton. Syst. **36**(1), 37–51 (2001). https://doi.org/10.1016/S0921-8890(01)00113-0
14. Powers, W.T.: Behavior: The Control of Perception. Wildwood House Ltd. (1974). ISBN: 9780704500921
15. Rahmani, M., Ghanbari, A., Ettefagh, M.M.: Robust adaptive control of a bio-inspired robot manipulator using bat algorithm. Expert Syst. Appl. **56**, 164–176 2016). https://doi.org/10.1016/j.eswa.2016.03.006
16. Reyes-Ortiz, O.J., Useche-Castelblanco, J.S., Vargas-Fonseca, G.L.: Implementation of fuzzy PID controller in cascade with anti-windup to real-scale test equipment for pavements. Eng. Trans. (2020). https://doi.org/10.24423/ENGTRANS.1066.20200102
17. Utami, E., Sahrin, A.,Utomo, G.R.: Cascade control with PID-PSO method on the stabilizer unit. In: The 2nd International Conference on Applied Electromagnetic Technology (AEMT) 2018 (2018)
18. Trunk, G.V.: A problem of dimensionality: a simple example. IEEE Trans. Pattern Anal. Mach. Intell. PAMI **1**(3), 306–307 (1979). https://doi.org/10.1109/TPAMI.1979.4766926
19. Wang, Q.J.: Using genetic algorithms to optimise model parameters. Env. Model. Softw. **12**(1), 27–34 (1997). https://doi.org/10.1016/S1364-8152(96)00030-8
20. Wiener, N.: Cybernetics or Control and Communication in the Animal and the Machine. Technology Press (1948)
21. Yang, X.S.: A new metaheuristic bat-inspired algorithm. Stud. Comput. Intel. **284**, 65–74 (2010). https://doi.org/10.1007/978-3-642-12538-6