Robots Producing Their Own Hierarchies with DOSA; The Dependency-Oriented Structure Architect

Benjamin Hawker
The Department Of Computer Science
The University of Sheffield
Sheffield, United Kingdom
ben.hawker@sheffield.ac.uk

Roger K Moore
The Department Of Computer Science
The University of Sheffield
Sheffield, United Kingdom
r.k.moore@sheffield.ac.uk

***Abstract*—Hierarchical Control Structures are employed in a variety of robots. However, the choice of hierarchy to use affects the ability to control the environment. If a robot could help define the required control hierarchy, this would help avoid the engineer picking an invalid hierarchy and secure effective control. This paper proposes a methodology named DOSA (Dependency Oriented Structuring Architect) which allows a robot to define their own control hierarchy from experience. DOSA brings consistency to hierarchical control, using a clear understanding of hierarchies to encapsulate the heuristic approach an engineer uses and automate a process that requires specialised expertise.**

***Index Terms*—Hierarchical, Control, Automation, Learning**

I. Introduction

Hierarchical Structures are used across robotic solutions with variable success [1]–[3]. Hierarchies when built well have competence that outdoes non-hierarchical approaches [4]. It is a challenge to find a definition of a hierarchy that gives the required insight for an engineer to be able to produce a valid control hierarchy. Prescott identifies the difference between a layered system and a hierarchical system [5], but no clearer definition exists. Solutions to robotic problems that employ hierarchical controllers show that a hierarchy is formed from dependencies, where the control of one signal requires the control of another [6]. Dependencies therefore are key to understanding a hierarchical structure, which are something an engineer and perhaps an autonomous agent can deduce. The question then becomes whether these dependencies can be identified. This paper proposes the use of dependencies between input signals as a means of understanding the constraints on possible hierarchies. Then, these constraints can be used to identify a suitable plan of experimentation in the environment that would allow a suitable hierarchy to be deduced. This framework for deriving hierarchies can be used to identify valid hierarchies, improving a control engineer's derivation of the hierarchy and in turn improving control.

II. DOSA; The Dependency-Oriented Structuring Architect

A. How Dependencies Could Identify the Hierarchy

To briefly consider closed loop control, a controller should output to all actuators that it needs control over. It should not be outputting to actuators that do not affect the input signal being controlled. This would be at best inconsequential, or more likely counter-productive. This principle can reduce the possible hierarchies before attempting any combinations. Identifying which actuators affect which inputs is possible either with simple observation or automatically with Input-Output Analysis [7]. If a controller for a specific input is to hierarchically depend on another controller, it should be to control actuators that it is affected by and no other actuators. After all, a controller should not be directly or indirectly outputting to actuators that are inconsequential to the control task of that controller.

B. Stage One: Identifying Possible Hierarchical Constraints

For each input, the set of actuators that must be controlled can be identified. A controller for that input should only hierarchically delegate to a controller whose set of actuators is a subset of its own actuator set. This way, it will not end up outputting to an actuator that it does not require for control. If a robotic arm with motors and sensors at each joint are considered, there would be some clear findings.

- The position of the hand is affected by the wrist joint, the elbow joint and the shoulder joint.
- The position of the wrist is affected by the elbow and shoulder joints, but not the wrist joint
- The position of the elbow is only affected by the shoulder joint.
- Nothing changes the position of the shoulder, so the position of the shoulder need not be considered.

By examining these subsets, invalid hierarchies can be identified, as detailed in Fig. 1. The possible combinations of hierarchically arranged controllers is reduced by understanding the inputs they affect.

	Shoulder Joint	Elbow Joint	Wrist Joint
Elbow Position	Yes	No	No
Wrist Position	Yes	Yes	No
Hand Position	Yes	Yes	Yes

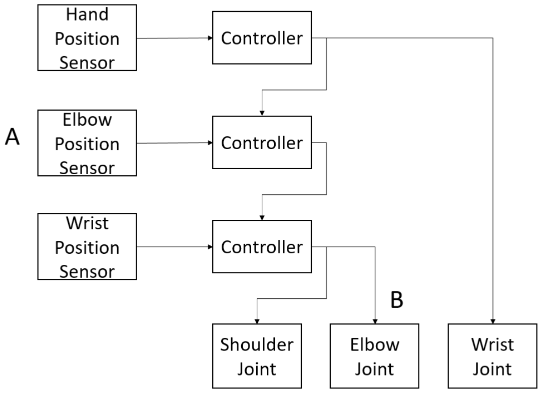

Fig. 1. A table showing which inputs are affected by which actuators above and an example controller for those inputs and actuators below. This controller is invalid when considering the constraints from the table above. The elbow position control signal (A) outputs to the elbow joint actuator (B) indirectly. The table indicates the elbow position is not dependent on the elbow joint, making the hierarchy invalid.

C. Stage Two: Identify Resolution Order

The agent can now developmentally experiment to work out which hierarchy is best, but a suitable order is required to allow this developmental progression. This becomes a simple logic problem, since no controller should be processed before any possible subordinate controller is processed. Therefore, controllers need to be added to the queue such that no possible subordinate is added after it.

- Make a list of every controller and the set of actuators it must output to.
- For each controller's actuator set, identify how many other controller's actuator set is a subset.
- Every controller that has no subsets among the other controller's actuator sets is safe to process, as it has no possible subordinates in the remaining controllers. Add all those controllers to the resolution queue, and remove them from the list.
- If the list is empty, the task is complete. Otherwise, return to the second step.

D. Section Three: Identifying a Suitable Hierarchy Through Progressive Experimentation

Each controller can now be progressively added, attempting the possible different arrangements that would meet the constraints of that controller. Fig. 2 demonstrates the addition of a second controller and the valid arrangements including this controller. Given an optimisation algorithm to tune the controller's parameters, the best arrangement can be selected that minimises error over all controllers. This could be either hierarchically placed on top of one or more existing controllers or entirely aside from them. The process is complete once all controllers have been added in the resolution order. This process can streamline the complex process of trying all hierarchical configurations and make obtaining hierarchies practical and led by unifying methodology.

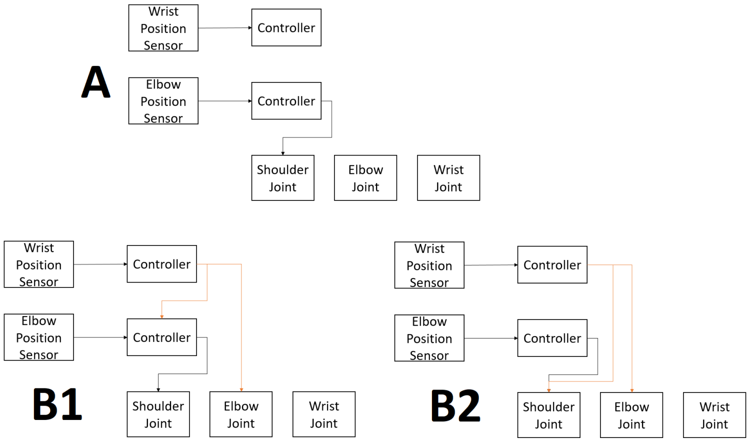

Fig. 2. A controller that is in the process of being progressively derived (A) and two possible options for where the current controller could connect to (B1 and B2). Given the constraints in Fig. 1, reaching the shoulder joint and elbow joint actuator can be achieved in two ways. This is either by controlling an existing controller (B1) or connecting directly to the actuators (B2) and can be decided by which produces the least error.

III. Limitations, Discussion and Future Work

This paper details a methodology named DOSA that can derive hierarchies from experience in the environment. Also, this paper clarifies the nature of a hierarchy and how dependencies are critical to the definition and deployment of a hierarchy. This paper only considers simple scenarios, so more complex scenarios need to be considered to allow a wide application for DOSA. Such examples would be inputs that have complex relationships with actuators (such as multiple actuators being required to change an input value). Input-Output analysis covers understanding many complex input-output relationship [8], [9] but testing that these work in control specific environments is required. Furthermore, this paper doesn't cover the required action if inputs have an identical actuator set. A variety of solutions have been derived by hand, including a hierarchy for the arm used in Living Control Systems 3. The next steps are to automate DOSA and apply it on a simple robotic control system that would typically be solved by a cascade controller. Input-Output analysis shall be used for stage one and evolutionary algorithms to optimise in step three. The long-term objective of this work is to have agents be able to fully derive and optimise their own control hierarchies independent of a human engineer. This would save required time, expertise and guarantee a consistent and effective control hierarchy.

References

[1] A. Sahrin and P. M. H, "Cascade Control With PID-PSO Method on The Stabilizer Unit," in *The 2nd International Conference on Applied Electromagnetic Technology (AEMT) 2018*, 2018.

[2] R. Brooks, “A robust layered control system for a mobile robot,” *IEEE Journal on Robotics and Automation*, vol. 2, no. 1, pp. 14–23, 1986. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1087032

[3] B. Digney and M. Gupta, “A distributed adaptive control system for a quadruped mobile robot,” in *IEEE International Conference on Neural Networks*. IEEE, 1993, pp. 144–149. [Online]. Available: http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=298516

[4] Y. Zhong and Y. Luo, “Comparative study of single-loop control and cascade control of third-order object,” in *Procedia Engineering*, vol. 15. Elsevier, jan 2011, pp. 783–787.

[5] T. J. Prescott, P. Redgrave, and K. Gurney, “Layered Control Architectures in Robots and Vertebrates,” *Adaptive Behavior*, vol. 7, no. 1, pp. 99–127, jan 1999. [Online]. Available: http://adb.sagepub.com/content/7/1/99.short

[6] R. Kennaway and L. R. Date, “Control of a multi-legged robot based on hierarchical PCT http://www2.cmp.uea.ac.uk/˜jrk/PCT/jp9913p.pdf,” 1999.

[7] P. J. Schroeder and B. Korel, “Black-box test reduction using input-output analysis,” in *Proceedings of the ACM SIGSOFT 2000 International Symposium on Software Testing and Analysis*, vol. 25, no. 5, sep 2000, pp. 173–177. [Online]. Available: http://portal.acm.org/citation.cfm?doid=347636.349042

[8] H. Ying, “Deriving analytical input-output relationship for fuzzy controllers using arbitrary input fuzzy sets and Zadeh fuzzy AND operator,” *IEEE Transactions on Fuzzy Systems*, vol. 14, no. 5, pp. 654–662, oct 2006.

[9] P. Roy, R. Ray, C. Wang, and W. F. Wong, “ASAC: Automatic sensitivity analysis for approximate computing,” in *ACM SIGPLAN Notices*, vol. 49, no. 5. New York, New York, USA: Association for Computing Machinery, may 2014, pp. 95–104. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2597809.2597812