This is the accepted version of a paper presented at *International Conference on Intelligent Robots and Systems (IROS) 2022, Kyoto*.

Citation for the original published paper:

Iannotta, M., Dominguez, D C., Stork, J A., Schaffernicht, E., Stoyanov, T. (2022) Heterogeneous Full-body Control of a Mobile Manipulator with Behavior Trees In: *IROS 2022 Workshop on Mobile Manipulation and Embodied Intelligence (MOMA): Challenges and Opportunities*  
https://doi.org/10.48550/arXiv.2210.08600

N.B. When citing this work, cite the original published paper.

Permanent link to this version:  
http://urn.kb.se/resolve?urn=urn:nbn:se:oru:diva-102984
Heterogeneous Full-body Control of a Mobile Manipulator with Behavior Trees

Marco Iannotta*, David Cáceres Domínguez*, Johannes A. Stork, Erik Schaffernicht, and Todor Stoyanov

Abstract—Integrating the heterogeneous controllers of a complex mechanical system, such as a mobile manipulator, within the same structure and in a modular way is still challenging. In this work we extend our framework based on Behavior Trees for the control of a redundant mechanical system to the problem of commanding more complex systems that involve multiple low-level controllers. This allows the integrated systems to achieve non-trivial goals that require coordination among the sub-systems.

I. INTRODUCTION

Sophisticated mechanical systems, such as mobile manipulators, typically involve multiple sub-systems that have to be coordinated to create complex behaviors and achieve multiple goals at the same time. Integrating the control of these sub-systems in a modular way within the same structure is still challenging, because multiple and heterogeneous controllers that run at different control frequencies are required.

In [1], we propose a novel method to combine a Stack-of-Tasks (SoT) control strategy [2], [3] with Behavior Trees (BTs) [4]. The former is an approach that allows a redundant (i.e., DOF > 6) robot to fulfill a number of prioritized goals. The latter is a task switching structure functionally equivalent to Finite State Machines (FSMs) [5], [6], but that promises to address some of its limitations in terms of reactivity (i.e., ability to quickly and efficiently react to changes) and modularity (i.e., system’s components may be separated into building blocks, and recombined). In our approach, a BT and an SoT strategy run in parallel at different control frequencies. On the basis of the current robot and environment state, the BT periodically configures a hierarchical control problem, which is then solved by the SoT strategy at a much higher control frequency. In [1], we demonstrate how this framework can be successfully applied to a 7-DOF manipulator to achieve complex goals in a modular, transparent and reactive way.

In this paper, we extend the framework proposed in [1] to the problem of commanding additional controllers for the handling of more complex systems. Due to the nature of BTs, any behaviour can be encapsulated within a leaf node. In this way, the use of standard BT leaf nodes allows to handle the tasks for different and heterogeneous controllers within the same BT (Fig. 1). The proposed extension does not affect the advantages in terms of modularity, reactivity and transparency mentioned in the original work. Moreover, handling heterogeneous controllers under the same policy allows the integrated systems to achieve non-trivial goals that would be difficult to attain by controlling the sub-systems in isolation.

We test our approach in simulation on a mobile manipulator, whose goal is to approach a table, and perform a pick and place operation, while the mobile platform is moving (Fig. 3).

II. BACKGROUND

In this section we provide an overview of the used methodologies for the proposed approach. We start by describing the SoT control strategy in II-A and the BTs in II-B. In Sec. II-C we recall the main concepts of the framework we propose in [1] to combine the SoT approach with BTs.

A. Hierarchical Stack-of-Tasks Control Strategy

Each goal for a redundant mechanical system can be formulated in terms of minimizing a separate task (or error) function of the robot state, which can be regulated with an ordinary differential equation. In case of multiple tasks, the corresponding equations can be sorted by priority and solved each in the solutions set of higher priority tasks (task-priority or Stack-of-Tasks). Kanoun et al. [2] proposed a prioritized task-regulation framework based on a sequence of quadratic programs (QP) that generalizes the previous task-priority framework [3] to inequality tasks. In [7] a more numerically efficient solution of the same problem was proposed.

Fig. 1. Integration between the unified SoT-BT framework proposed in [1] and additional controllers. The BT, the SoT approach and the additional controller run in parallel at different control frequencies.
The assumption behind this method is that it is always possible to reach a globally optimal solution by minimizing a quadratic error function. In reality, this is not always the case and it often happens to deal with problems that require non-quadratic objectives. In these situations, a single SoT might not be enough, because the control strategy might bring the robot to a disadvantageous configuration with respect to the global goal. The only way to address these scenarios, without considering a more complex error function or a motion planner, is to compose a sequence of local approximations (i.e., Stack-of-Tasks) that prevent the control strategy from falling into a local minimum. In [1] we propose to exploit these nodes for task removal when needed, without removing the possibility to use the standard ones just for controlling the flow of the BT.

We use the new Control Nodes to remove the previously set tasks. Task removal takes place immediately before the Control Node returns its final outcome (i.e., Success or Failure), without affecting the node behavior described in Section II-B. The creation of new Control Nodes allows us to exploit these nodes for task removal when needed, without removing the possibility to use the standard ones just for controlling the flow of the BT.

In this paper we extend the framework presented in Sec. II-C to the problem of commanding complex mechanical systems that are tracked by heterogeneous low-level controllers. The key point of having the BT and the SoT approach running in parallel at different control frequencies is extended to the new controller. The use of a BT as control policy allows us to easily perform the integration of the new controller, by exploiting the intrinsic modular nature of BTs when performing task composition. As previously mentioned, in [1] we define for convenience new Action and Control Nodes to respectively set and remove single manipulation tasks (e.g., follow a line or go to a position) in the SoT. On the contrary, current works on BTs typically encapsulate entire operations that involve multiple goals in standard Action Nodes. To show the ease of integration of our framework with every kind of controller and approach in the literature, we use standard Action Nodes to handle the
tasks related to additional controllers. However, the use of a control strategy similar to the SoT one, that decomposes complex goals in a set of tasks to be handled separately in the Action Nodes, would improve the overall modularity and transparency. The resulting framework is shown in Fig. 1. At each tick, new tasks for the additional controllers are commanded in addition to the dynamic update of the hierarchical control problem for the SoT strategy. In this way, we can coordinate the sequential or simultaneous execution of different controllers to achieve one or multiple goals at the same time.

We test our approach in simulation with a Franka Emika Panda 7-DOF manipulator installed on a 4-wheel SUMMIT-XL STEEL mobile platform. We build upon the implementation of the combined SoT and BT framework used in [1]. We design a task that consists of approaching a table, picking up and placing a 30mm cube from one side of the table to the other, while the mobile platform moves between the extremes of the table. Fig. 3 provides an overview of the task execution and its relative BT. For brevity, we do not show the entire subtree for the pick and place operation, which we designed according to the methodology described in [1]. The robot starts in position A. The Action Node Move to C is executed to make the robot approach the table, while the Condition Node Robot Close to the Table prevents the Pick and Place subtree from being executed (Fig. 3(a)). Once the robot reaches the position B, the condition is met and the Pick and Place subtree is ticked while the Action Move to C is still running (Fig. 3(b)). When the robot reaches the end of the table (position C), the Action Move to B is executed without affecting the right part of the tree (Fig. 3(c)). The Decorator Node Repeat makes the platform keep on moving between the positions B and C until the pick and place operation is completed.

For the control of the mobile platform we use a standard differentiation kinematic controller and we keep the velocity constant in module, switching only its direction to move the platform between the positions B and C. The modular nature of BTs has allowed us to design the BT and tune the controllers parameters in isolation for the two operations performed by the robotic arm (i.e., picking and placing) and the one performed by the mobile platform (i.e., movement between the two positions). However, a further tuning of the SoT parameters was needed to make the pick and place operation work when combined with the platform movement. In Fig. 4 we show the execution in the Gazebo simulator [9] of the pick and place operation described in Fig. 3(b) and 3(c).

The task taken into account allows to test the framework in a context that requires a basic form of coordination between the two involved controllers. On the one hand, thanks to the BT, we can easily manipulate the coordination under the same control policy. On the other hand, the use of the framework proposed in [1] allows to inherit its advantages in terms of modularity and transparency. Moreover, the platform movement during the pick and place simulates a possible navigation task that is typically performed by a mobile manipulator and makes the pick and place operation more challenging for the robot.

IV. CHALLENGES

The task described in Sec. III for evaluating the proposed approach highlights two open challenges which are essentially related to the parameters tuning and the policy design. As previously mentioned, we needed to additionally refine the parameters of the SoT when performed in parallel with the platform movement. This aspect underlines that, while the BT nature improves the policy modularity from the
design point of view, the combination of different controllers brings inevitably some dependence in the parameters tuning. Thus, the re-usability of subtrees in the BT is limited by the disadvantage of having to tune anyway the parameters involved, which becomes even harder when combining multiple controllers, given to the combinatorial explosion.

Moreover, in general the policy design is not as trivial as for the simple task considered in this paper. Designing the BT to coordinate multiple controllers to achieve multiple goals at the same time is extremely complex, especially when dealing with dynamic environments, as it becomes difficult to predict all possible scenarios in advance. Recent works have showed promising results that apply different methodologies to learn both the parameters and the structure of a BT [10], [11], [12]. Our framework opens new possibilities for applying learning methodologies to address the above-mentioned challenges.

REFERENCES

[1] D. Cáceres Domínguez, M. Iannotta, J. A. Stork, E. Schaffernicht, and T. Stoyanov, “A stack-of-tasks approach combined with behavior trees: A new framework for robot control,” *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 12110–12117, 2022. [Online]. Available: https://doi.org/10.1109/LRA.2022.3211481

[2] O. Kanoun, F. Lamiraux, and P.-B. Weber, “Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task,” *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 785–792, 2011.

[3] B. Siciliano and J.-J. Slotine, “A general framework for managing multiple tasks in highly redundant robotic systems,” in *Fifth International Conference on Advanced Robotics 'Robots in Unstructured Environments*, 1991, pp. 1211–1216 vol.2.

[4] M. Iovino, E. Sculins, J. Styrud, P. Ögren, and C. Smith, “A survey of behavior trees in robotics and AI,” *CoRR*, vol. abs/2005.05842, 2020. [Online]. Available: https://arxiv.org/abs/2005.05842

[5] P. Ögren, “Increasing modularity of uav control systems using computer game behavior trees,” 08 2012.

[6] A. Marzinotto, M. Colledanchise, C. Smith, and P. Ögren, “Towards a unified behavior trees framework for robot control,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 5420–5427.

[7] A. Escande, N. Mansard, and P.-B. Wieber, “Hierarchical quadratic programming: Fast online humanoid-robot motion generation,” *The International Journal of Robotics Research*, vol. 33, no. 7, pp. 1006–1028, 2014.

[8] M. Colledanchise and P. Ögren, “Behavior trees in robotics and AI: an introduction,” *CoRR*, vol. abs/1709.00084, 2017. [Online]. Available: http://arxiv.org/abs/1709.00084

[9] N. Koenig and A. Howard, “Design and use paradigms for gazebo, an open-source multi-robot simulator,” in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE Cat. No. 04CH37566), vol. 3. IEEE, pp. 2149–2154.

[10] M. Mayr, K. Chatzilygeroudis, F. Ahmad, L. Nardi, and V. Krueger, “Learning of parameters in behavior trees for movement skills,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 7572–7579.

[11] M. Colledanchise, R. Parasuraman, and P. Ögren, “Learning of behavior trees for autonomous agents,” *IEEE Transactions on Games*, vol. 11, no. 2, p. 183–189, Jun 2019. [Online]. Available: http://dx.doi.org/10.1109/TG.2018.2816806

[12] K. French, S. Wu, T. Pan, Z. Zhou, and O. C. Jenkins, “Learning behavior trees from demonstration,” in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7791–7797.