Generalizing Behavior Trees and Motion-Generator (BTMG) Policy Representation for Robotic Tasks Over Scenario Parameters

Faseeh Ahmad[1], **Matthias Mayr**[1], **Elin Anna Topp**[1], **Jacek Malec**[1], **Volker Krueger**[1]

[1]Lund University

{faseeh.ahmad, matthias.mayr, elin_anna.topp, jacek.malec, volker.krueger}@cs.lth.se

Abstract

We propose a generalisation of a behaviour tree and motion-generator based robot arm policy representation for learning and solving tasks such as contact-rich tasks like peg insertion or pushing an object. We use planning to generate skill sequences needed to execute these tasks and rely on reinforcement learning to obtain parameters of the policy. We assume gaussian processes as a suitable method for this generalisation and present preliminary, promising results from initial experiments.

Introduction

In previous papers (Rovida et al. 2018; Mayr et al. 2021, 2022) we have developed a representation based on behavior trees (BT) (Colledanchise and Ögren 2014) and motion-generator (MG), (BTMG). (Rovida et al. 2018) They are easy to interpret, can be **robust** to faults and errors that can occur during execution and they can be **reactive**, allowing the robot to act and deal with uncertain conditions and recover from failures.

BTMG is a parameteric policy representation that allow us to solve contact-rich tasks like “peg-in-hole” or pushing an object. Parameters of a BTMG can vary from deciding the structure of behavior trees to specifying the actual controller stiffness values of the MG. We can either specify these parameters manually (Rovida et al. 2018) or learn them using reinforcement learning (RL) (Mayr et al. 2021, 2022). We generate skill sequences for these tasks in a skill-based system, SkiROS (Rovida et al. 2017) that uses Planning Domain Definition Langauage (PDDL) for task planning.

Although BTMGs are shown to be quite promising, one key shortcoming of this representation is that they can be scenario specific. For instance, pushing an object to different goal locations using a BTMG requires learning the parameters. This is problematic and is also common for the original formalization of dynamic motion primitives (DMPs) (Ijspeert, Nakanishi, and Schaal 2002). This problem was later resolved by generalizing DMPs (Ijspeert et al. 2013). In this work, we aim to generalize the BTMG policy representation in a similar way. Our proposed solution is to generate a novel BTMG for a new task instance as a weighted sum of the ”basis”-BTMG (parametric BTMG for different instances of a task). This poses some interesting questions: 1) What kind of Basis-BTMGs should we use? 2) How many Basis-BTMGs do we need? 3) How can we use Basis-BTMGs to interpolate?

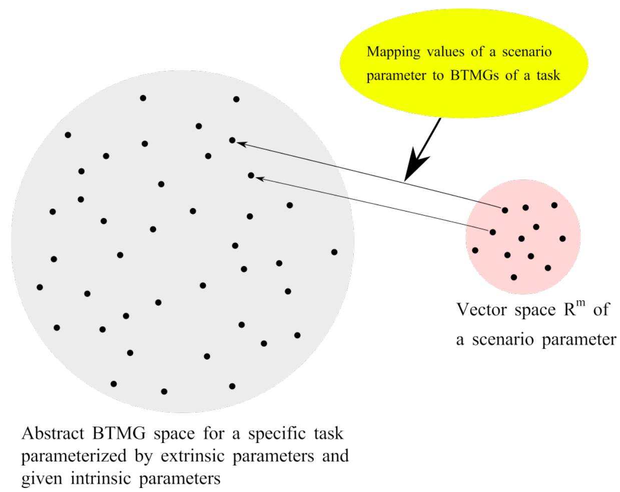

Figure 1: *The mapping (yellow) of the vector space of a scenario parameter (pink) to the abstract BTMG space of a task (gray).*

Formalization

A BTMG is parameterized by two types of parameters; **intrinsic parameters** and **extrinsic parameters**. Intrinsic parameters decide the structure of the BT, number of control flow and execution nodes, etc. These parameters also decide how much velocity can be allowed, how fast the arm should move, etc. Implicit parameters could be implied by the specific task at hand, e.g., push task and peg-in-a-hole task. In this work, we do not want to change intrinsic parameters. Extrinsic parameters on the other hand represent, e.g., how much force can be applied, offsets, path velocity of the end-effector, etc. In a nutshell, extrinsic parameters are optimized while intrinsic ones are assumed to be known apriori. Note that object goal pose is not necessarily a parameter here. Consider a pushing task: while the object goal pose represents the centre of mass of the object at the goal location, the point on the object where the peg touches the object is expected to be different. The centre of mass of the object should be at the goal location. We specify pushing the object through a push vector(see Figure 2 defined by start and goal offsets from object start and goal locations.

In previous work (Mayr et al. 2022), we have used RL to learn extrinsic parameters of the BTMG policy representation for a specific instance of a push task and a peg insertion task. Apart from BTMGs, we also consider scenario parameters like object goal pose, object start pose, object weight, etc. These parameters represent variations or dimensions over which we want to generalize the BTMG representation of a task. Figure 1 shows the vector space $\mathbb{R}^m$ of a particular scenario parameter (shown in pink). Every point in this space shows a set of unique values of scenario parameters. For instance, any point or object goal pose would be a 6D-vector $x, y, z, \alpha, \beta, \gamma$ representing the goal pose of the object.

Mapping

In order to generate various policy representations of a task that generalizes over a scenario parameter, we are interested in a mapping that for a given task and scenario parameter to the corresponding extrinsic BTMG parameters. Figure 1 shows the mapping (yellow) that maps the vector space of a scenario parameter to BTMGs of a specific task.

We propose to use gaussian processes (GP) (Rasmussen 2003; Forte, Ude, and Gams 2011; Zhou and Asfour 2017) as a mapping function. We start by collecting data samples using RL by learning extrinsic parameters for BTMGs of a task over a particular scenario parameter. We use these samples to train the GP by using a scenario parameter as input and extrinsic parameters as output. The idea is to then use this trained GP to interpolate and return the values of extrinsic parameters for different values of the scenario parameter.

We also want to clarify that using GP to interpolate is not a new idea as it has already been used in literature (Forte et al. 2012; Rasmussen 2003). The novelty of our approach lies in using GPs as a mapping function in the context of BTMG policy representation that allows it to generalize over scenario parameters.

Using GP has two major benefits: 1) It provides mean and variance bounds over the extrinsic parameters of BTMG of a skill. 2) They are known for generalizing over domains and have been used in this context in Dynamic Motion Primitives (DMPs) as discussed before.

Experiments

We tested our approach on a push task where the robot had to push an object from a start location to a specified object goal poses, see Figure 2. In our setting, the BTMG of the push task has four learnable extrinsic parameters: 1) Offsets in start locations s_x, s_y, 2) Offsets in goal locations g_x, g_y. Together, these parameters decide the push vector. We start by collecting training and testing samples of goal locations of the object. Instead of randomly choosing samples over the space we use Latin hypercube space(Ye 1998) to achieve evenly distributed samples across the entire region. We choose defined number of samples within the bounds. Choosing the number of samples is not a trivial task and dependent on work space and the type of the scenario parameter.

We use RL to get the best s_x, s_y, g_x, g_y for every training sample. These are used to train our GP. The GP takes object goal pose as input and produces offsets s_x, s_y, g_x, g_y as output. For simplicity, we only change the x and y coordinates of the object goal pose. The trained GP is then used to generate offsets for the test points.

We trained the GP on samples distributed across a restrictive space and tested it on unseen samples. The initial results look promising as the GP was able to find offsets that managed to solve the task for all the test points. The offsets managed to push the object throughout without slipping off. We analyzed the performance of offsets by calculating the error between actual and specified goal pose of the object. Initial results suggest that the error for the offsets obtained thorough GP is in the same range for the offsets learned through RL.

Future Work

For future work, we are planning to generalize over a larger space to obtain BTMG parameters for multiple scenario parameters together. We would also like to extend this generalization to other tasks.

We would also like to evaluate the performance of trained GP models for a skill by comparing it with baseline regression models. Since, we aim to use a generic model, we expect GP to perform well to generalize scenario parameters of BTMG of difference skills.

We would also like to investigate sensitivity of the model to different scenario parameters. Basically how well GP performs for different types of scenario parameters? Furthermore, we would also like to see if GP can be used to interpolate intrinsic parameters as well i.e. generating new BTs for a skill.

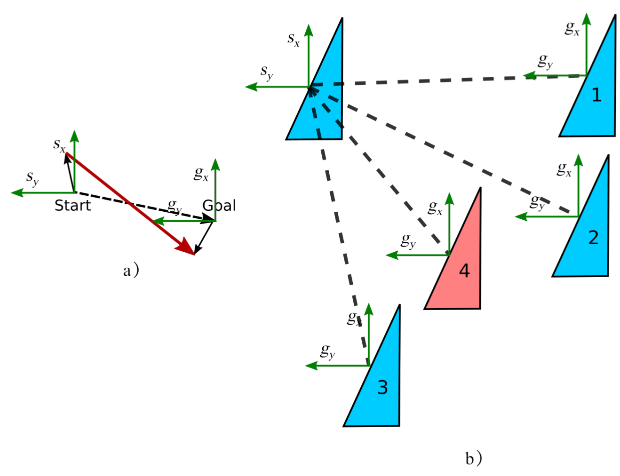

Figure 2: *a) Shows the push vector (red) defined by offsets in start and goal location. The image b) shows the push task setup with different goal locations where (1-3) are training examples and (4) is a test example.*

References

Colledanchise, M.; and Ögren, P. 2014. How Behavior Trees Modularize Robustness and Safety in Hybrid Systems. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1482–1488.

Forte, D.; Gams, A.; Morimoto, J.; and Ude, A. 2012. On-line motion synthesis and adaptation using a trajectory database. *Robotics and Autonomous Systems*, 60(10): 1327–1339.

Forte, D.; Ude, A.; and Gams, A. 2011. Real-time generalization and integration of different movement primitives. *2011 11th IEEE-RAS International Conference on Humanoid Robots*.

Ijspeert, A. J.; Nakanishi, J.; Hoffmann, H.; Pastor, P.; and Schaal, S. 2013. Dynamical movement primitives: learning attractor models for motor behaviors. *Neural computation*, 25(2): 328–373.

Ijspeert, A. J.; Nakanishi, J.; and Schaal, S. 2002. Learning rhythmic movements by demonstration using nonlinear oscillators. In *Proceedings of the ieee/rsj int. conference on intelligent robots and systems (iros2002)*, CONF, 958–963.

Mayr, M.; Ahmad, F.; Chatzilygeroudis, K.; Nardi, L.; and Krueger, V. 2022. Skill-based Multi-objective Reinforcement Learning of Industrial Robot Tasks with Planning and Knowledge Integration. *arXiv preprint arXiv:2203.10033*.

Mayr, M.; Chatzilygeroudis, K.; Ahmad, F.; Nardi, L.; and Krueger, V. 2021. Learning of Parameters in Behavior Trees for Movement Skills. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Rasmussen, C. E. 2003. *Gaussian processes in machine learning*. Springer Science & Business Media.

Rovida, F.; Crosby, M.; Holz, D.; Polydoros, A. S.; Großmann, B.; Petrick, R. P.; and Krüger, V. 2017. SkiROS-A skill-based robot control platform on top of ROS. In *Studies in Computational Intelligence*, volume 707, 121–160.

Rovida, F.; Wuthier, D.; Grossmann, B.; Fumagalli, M.; and Krüger, V. 2018. Motion Generators Combined with Behavior Trees: A Novel Approach to Skill Modelling. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 5964–5971.

Ye, K. 1998. Orthogonal column Latin hypercubes and their application in computer experiments. *Journal of the American Statistical Association*.

Zhou, Y.; and Asfour, T. 2017. Task-oriented generalization of dynamic movement primitive. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.