The Journal of Open Source Software

# sbp-env: A Python Package for Sampling-based Motion Planner and Samplers

**Tin Lai**[1]

**1** School of Computer Science, The University of Sydney, Australia

**DOI:** 10.21105/joss.03782

**Software**

- Review
- Repository
- Archive

**Editor:** Daniel S. Katz

**Reviewers:**

- @KanishAnand
- @OlgerSiebinga

**Submitted:** 26 September 2021
**Published:** 15 October 2021



## Background

Sampling-based motion planning is one of the fundamental methods by which robots navigate and integrate with the real world (Elbanhawi & Simic, 2014). Motion planning involves planning the trajectories of the actuated part of the robot, under various constraints, while avoiding collisions with surrounding obstacles. Sampling-based motion planners (SBPs) are robust methods that avoid explicitly constructing the often intractable high-dimensional configuration space (C-Space). Instead, SBPs randomly sample the C-Space for valid connections and iteratively build a roadmap of connectivity. Most SBPs are guaranteed to find a solution if one exists (Kavraki et al., 1996), and such a planner is said to be *probabilistic complete*. A further development for SBPs is *asymptotic optimality*(Elbanhawi & Simic, 2014): a guarantee that the method will converge, in the limit, to the optimal solution.

SBPs are applicable to a wide range of applications. Example include planning with arbitrary cost maps (Iehl et al., 2012), cooperative multi-agent planning (Jiang & Wu, 2020), and planning in dynamic environments (Yershova et al., 2005). On the one hand, researchers have focused on the algorithmic side of improving the graph or tree building (Elbanhawi & Simic, 2014; Klemm et al., 2015; Lai et al., 2019; Lai, 2021; Lai & Ramos, 2021b; Zhong & Su, 2012). On the other hand, the advancement of neural networks allows an abundance of learning approaches to be applied in SBPs (Bagnell, 2014; Strub & Gammell, 2020) and on improving the sampling distribution (Alcin et al., 2016; Lai et al., 2020, 2021; Lai & Ramos, 2020, 2021a).

## Statement of need

The focus of motion planning research has been mainly on (i) the algorithmic aspect of the planner using different routines to build a connected graph and (ii) improving the sampling efficiency (with methods such as heuristic or learned distribution). Traditionally, robotic research focuses on algorithmic development, which has inspired several motion planning libraries written in C++, such as Move3D (Simeon et al., 2001) and OMPL (Sucan et al., 2012). In particular, OMPL has been one of the most well-known motion planning libraries due to its versatility, and it has been a core part of the planning algorithm used in the MoveIt framework (Chitta et al., 2012). However, swapping the sampler within each planner is very restrictive, as planners are typically hard-coded to use a specific sampler. In addition, it is cumbersome to integrate any learning-based approach into a framework as there is only a limited number of choices of deep-learning libraries in C++.

Python has been a popular language to use in Machine Learning due to its rapid scripting nature. For example, PyTorch (Paszke et al., 2019) and Tensorflow (Abadi et al., 2016) are two popular choices for neural network frameworks in Python. A large number of learning

approaches are available as Python packages. It shall be noted that the aforementioned OMPL has Python bindings available; however, OMPL uses an outdated Py++ code generator, and every modification to the source code will require hours to updates bindings plus recompilation. Some Python repositories are available that are dedicated to robotics motion planning (Sakai et al., 2018); however, most only showcase various planning algorithms, without an integrated environment and simulators.

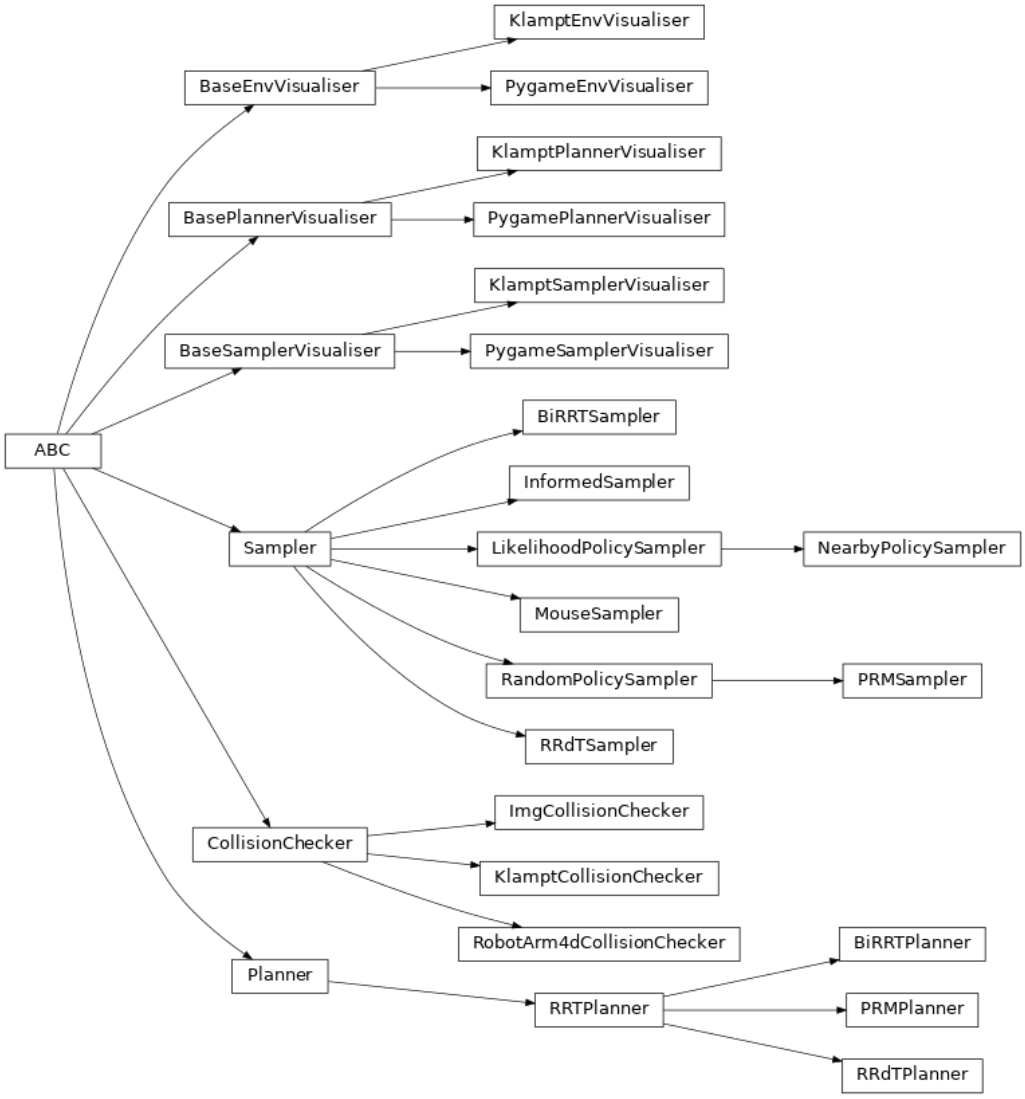


**Figure 1:** Implementation details on the class hierarchy structure of `sbp-env`.

## Overview

We introduce `sbp-env`, a *sampling-based motion planners' testing environment*, as a complete feature framework to allow rapid testing of different sampling-based algorithms for motion planning. `sbp-env` focuses on the flexibility of tinkering with different aspects of the framework, and it divides the main planning components into two main categories: (i) samplers and (ii) planners. The division of the two components allows users to decouple them and focus only on the component that serves as the main focus of the research. `sbp-env` has implemented the entire robot planning framework with multiple degrees-of-freedom, which

allows benchmarking motion planning algorithms with the same planner under different back-end simulators. Separating the two components allows users to quickly swap out different components in order to test novel ideas.

Building the framework enables researchers to rapidly implement their novel ideas and validate their hypotheses. In particular, users can define the environment using something as simple as an *image*, or as complicated as an *xml file*. All samplers and planners can be added as a plugin system, and `sbp-env` will auto-discover newly implemented planners or samplers that have been added to the dedicated folders.

Figure 1 illustrates the hierarical structure of our package. Our implementation of `sbp-env` define abstract interfaces for **sampler** and **planners**, from which all corresponding concrete classes must inherit. In addition, there are classes that represent the full-body simulations of the environments and the corresponding visualisation methods. Note that all visualisation can be turned off on-demand, which is beneficial when users benchmark their algorithms. The docunmentation of `sbp-env` is available at https://cs.tinyiu.com/sbp-env.

## References


Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., & others. (2016). Tensorflow: A system for large-scale machine learning. *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 265–283.

Alcin, O. F., Ucar, F., & Korkmaz, D. (2016). Extreme learning machine based robotic arm modeling. *2016 21st International Conference on Methods and Models in Automation and Robotics, MMAR 2016*, *1*, 1160–1163. https://doi.org/10.1109/mmar.2016.7575302

Bagnell, J. A. (2014). Reinforcement Learning in Robotics: A Survey. *Springer Tracts in Advanced Robotics*, *97*, 9–67. https://doi.org/10.1007/978-3-319-03194-1_2

Chitta, S., Sucan, I., & Cousins, S. (2012). Moveit! [ros topics]. *IEEE Robotics & Automation Magazine*, *19*(1), 18–19. https://doi.org/10.1109/mra.2011.2181749

Elbanhawi, M., & Simic, M. (2014). Sampling-based robot motion planning: A review. *IEEE Access*, *2*, 56–77. https://doi.org/10.1109/access.2014.2302442

Iehl, R., Cortés, J., & Siméon, T. (2012). Costmap planning in high dimensional configuration spaces. *2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, 166–172. https://doi.org/10.1109/aim.2012.6265953

Jiang, J., & Wu, K. (2020). Cooperative pathfinding based on memory-efficient multi-agent RRT. *IEEE Access*, *8*, 168743–168750. https://doi.org/10.1109/access.2020.3023200

Kavraki, L. E., Kolountzakis, M. N., & Latombe, J.-C. (1996). Analysis of probabilistic roadmaps for path planning. *Proceedings of IEEE International Conference on Robotics and Automation*, *4*, 3020–3025. https://doi.org/10.1109/robot.1996.509171

Klemm, S., Oberländer, J., Hermann, A., Roennau, A., Schamm, T., Zollner, J. M., & Dillmann, R. (2015). RRT*-Connect: Faster, asymptotically optimal motion planning. *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 1670–1677. https://doi.org/10.1109/robio.2015.7419012

Lai, T. (2021). Rapidly-exploring Random Forest: Adaptively Exploits Local Structure with Generalised Multi-Trees Motion Planning. *arXiv:2103.04487 [cs.RO]*. http://arxiv.org/abs/2103.04487

Lai, T., Morere, P., Ramos, F., & Francis, G. (2020). Bayesian Local Sampling-Based Planning. *IEEE Robotics and Automation Letters (RA-L)*, *5*(2), 1954–1961. https://doi.org/10.1109/lra.2020.2969145

Lai, T., & Ramos, F. (2020). Learning to Plan Optimally with Flow-based Motion Planner. *arXiv:2010.11323 [cs.RO]*. http://arxiv.org/abs/2010.11323

Lai, T., & Ramos, F. (2021a). PlannerFlows: Learning Motion Samplers with Normalising Flows. *IEEE/RSJ Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*.

Lai, T., & Ramos, F. (2021b). Rapid Replanning in Consecutive Pick-and-Place Tasks with Lazy Experience Graph. *arXiv:2109.10209 [cs.RO]*. http://arxiv.org/abs/2109.10209

Lai, T., Ramos, F., & Francis, G. (2019). Balancing Global Exploration and Local-connectivity Exploitation with Rapidly-exploring Random disjointed-Trees. *Proceedings of The International Conference on Robotics and Automation*. https://doi.org/10.1109/icra.2019.8793618

Lai, T., Zhi, W., Hermans, T., & Ramos, F. (2021). Parallelised Diffeomorphic Sampling-based Motion Planning. *Conference on Robot Learning (CoRL)*.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., & others. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, *32*, 8026–8037.

Sakai, A., Ingram, D., Dinius, J., Chawla, K., Raffin, A., & Paques, A. (2018). *PythonRobotics: A Python code collection of robotics algorithms*. http://arxiv.org/abs/1808.10703

Simeon, T., Laumond, J.-P., & Lamiraux, F. (2001). Move3D: A generic platform for path planning. *Proceedings of the 2001 IEEE International Symposium on Assembly and Task Planning (ISATP2001). Assembly and Disassembly in the Twenty-First Century.(cat. No. 01th8560)*, 25–30. https://doi.org/10.1109/isatp.2001.928961

Strub, M. P., & Gammell, J. D. (2020). Adaptively Informed Trees (AIT*): Fast Asymptotically Optimal Path Planning through Adaptive Heuristics. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 3191–3198. https://doi.org/10.1109/icra40945.2020.9197338

Sucan, I. A., Moll, M., & Kavraki, L. E. (2012). The open motion planning library. *IEEE Robotics & Automation Magazine*, *19*(4), 72–82.

Yershova, A., Jaillet, L., Siméon, T., & LaValle, S. M. (2005). Dynamic-domain RRTs: Efficient exploration by controlling the sampling domain. *Proceedings of IEEE International Conference on Robotics and Automation*, 3856–3861. https://doi.org/10.1109/robot.2005.1570709

Zhong, J., & Su, J. (2012). Triple-Rrts for robot path planning based on narrow passage identification. *2012 International Conference on Computer Science and Information Processing (CSIP)*, 188–192. https://doi.org/10.1109/csip.2012.6308826