PythonRobotics: a Python code collection of robotics algorithms

Atsushi Sakai — University of California, Berkeley
https://atsushisakai.github.io/
Daniel Ingram — Appalachian State University
ingramds@appstate.edu
Joseph Dinius — inVia Robotics, Inc.
jdinius@inviarobotics.com
Karan Chawla — Skyryse Inc.
karan@skyryse.com
Antonin Raffin — ENSTA ParisTech
antonin.raffin@ensta-paristech.fr
Alexis Paques — Unmanned Life
Alexis@unmanned.life

Abstract

This paper describes an Open Source Software(OSS) project: PythonRobotics[21]. This is a Python code collection of robotics algorithms, especially focusing on autonomous navigation. It aims for beginners of robotics to understand basic ideas of each algorithm. The algorithms which are practical and widely used in both academia and industry are selected. Each sample code in this project only depends some standard modules in Python 3 in order to easy to use and easy to understand the algorithm. In this paper, related works of this project, some key ideas about this OSS project, and brief structure of this repository are introduced. Future works of this project are also discussed.

1 Introduction

In recent years, autonomous navigation technologies have received a huge attention in many fields. For examples, autonomous driving[22], drone flight navigation, and other transportation systems. An autonomous system is a system that can operate for long periods of time without any external control.

The autonomous navigation needs a wide range of technologies. To accomplish autonomous navigation, the system needs to know where am I (localization), where is safe (mapping), where to go (path planning), and how to go (path following). The autonomous system would not work correctly if any of of them is missing.

Educational materials are becoming more and more important for future developers to learn basic autonomous navigation technologies. Because, these autonomous technologies needs different technological backgrounds such as linear algebra, statics, probability theory,

View on GitHub

PythonRobotics

Python sample codes for robotics algorithms.

tar.gz .zip

PythonRobotics

build passing Say Thanks !

Python codes for robotics algorithm.

Table of Contents

- What is this?
- Requirements
- How to use
- Localization
  - Extended Kalman Filter localization

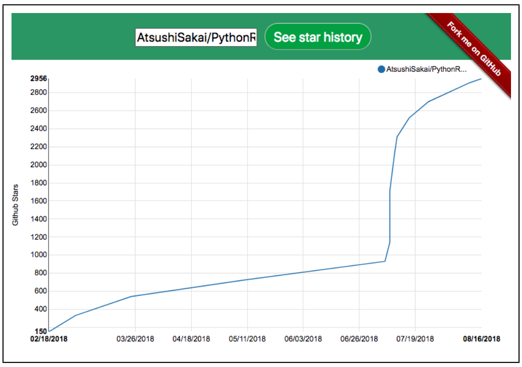


Figure 1: Left: GitHub star counts history graph using [16]

optimization theory, control theory, and sensing physics etc. It needs a lot of time to be familiar with these technological areas without good materials.

In this paper, an Open Source Software(OSS) project: PythonRobotics[21] is described. It aims for beginners of robotics to understand basic ideas of each algorithm. It is a code collection of robotics algorithms, especially focusing on autonomous navigation. It is written in Python[12] under MIT license[7].

This project was started from Mar. 21 2016 as a self-learning project. It already has over 3000 stars on GitHub, and the left figure in Fig.1 shows the star counts history graph using [16].

This paper is organized as follows: Section 2 reviews the related works. The philosophy of this OSS is presented in Section 3. It's repository structure description and some technological backgrounds and simulation results are provided in Section 4. Conclusions and some future works are drawn in Section 5. Section 6 shows acknowledgements for all contributors in this project. This is the project page https://atsushisakai.github.io/PythonRobotics/ and it's source code is provided at https://github.com/AtsushiSakai/PythonRobotics.

2 Related works

There are great references for learning autonomous navigation technologies.

S. Thrun et al. wrote a great text book "Probabilistic robotics" which is a bible of localization, mapping, simultaneous localization and mapping (SLAM) for mobile robotics[30]. E. Frazzoli et al. wrote a great survey paper about path planning and control techniques for autonomous driving [25]. G. Bautista et al. wrote a survey paper focusing on path planning for automated vehicles[22]. J. Levinson wrote an overview paper about systems and algorithms towards fully autonomous driving[24].

These papers helps readers to learn state of the arts autonomous navigation technologies. However, it might be difficult to understand the basic ideas of the technologies and algorithms for beginners of robotics, because these papers doesn't include implementation examples such as sample codes.

Many universities provide great courses to learn mobile robotics. For example, University of Freiburg provides a introduction course of mobile robotics[2]. Swiss Federal Institute of Technology in Zurich (ETH in Zurich) also provides a course of autonomous mobile robot[3] and robotics programming with Robot Operating System(ROS)[4].

These classes of universities also helps readers to learn the autonomous navigation technologies. However, it might be difficult to understand the basic ideas of the technologies and algorithms for beginners of robotics if you are not a student of the class, because these lecture notes doesn't include implementation examples of robotics algorithms.

Robot Operating System (ROS) is a middle-ware for robotics software development[13][28]. ROS initially released from 2007, it became a defacto standard platform in many industries fields not only academia. ROS includes basic navigation software such as Adaptive Monte Carlo Localization (AMCL package) and local path planner based on dynamic window approach[14]. Many autonomous navigation ROS packages are also opened as an OSS.

The OSS is also useful to learn basic ideas of autonomous navigation algorithms. However, it might be difficult sometimes, because of lack of documentation about it's basic algorithm. Also, it might be written by complicated programming languages such as C++ for calculation performance and platform reasons. This is not good for focusing to learn robotics algorithms.

Udacity Inc, which is a online learning company founded by S. Thrun, et al, is providing great online courses of autonomous navigation and autonomous driving [17]. This course provide not only technical description, programming homework using Python. The learning strategy is great for understanding it's technical background.

PythonRobotics project has a similar concept to the Udacity's course. It aims for beginners of robotics to understand basic technical ideas of each autonomous navigation algorithm. The details of concepts of this OSS project will be described in the next section.

3 Philosophy

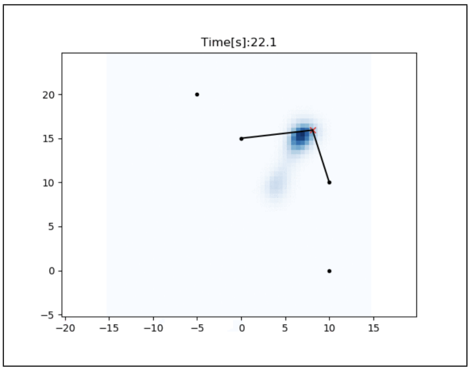
In this section, the philosophy of this project is described. This project based on three philosophies.

The first one is that the codes have to be easy to read for understanding each algorithm's basic idea. This project aims for beginners of robotics to understand basic ideas of each algorithm. Therefore, the code have to be easy to read and understand the algorithm. Programming language, Python[12] is adopted in this project because it has good code readability and it allows us to focus on algorithm itself. Python has great libraries for matrix operation, mathematical and scientific operation, and visualization. These libraries also allows us to focus on algorithm itself.

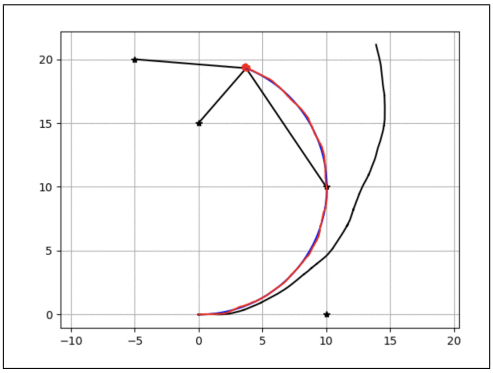
The second one is the algorithms which is widely used in academia and industry and practical are selected. For example, Kalman filters and particle filter for localization, grid mapping for mapping, dynamic programming based approaches and sampling based approaches for path planning, and optimal control based approach for path tracking.

The last philosophy is minimum dependency. It allows us to use the codes easily and to convert the codes to other programming languages such as C++, Java for practical usage. Each sample code only depends some modules on Python3 as bellows.

- numpy[8] for matrix operation
- scipy[15] for mathematics, science, and engineering computing
- matplotlib[6] for visualization
- pandas[10] for data analysis



Histogram filter



Particle filter

Figure 2: Localization simulation results

- cvxpy[5] for convex optimization

These modules are OSS and could be used for free. This repository doesn't include any commercial software.

4 Repository structure

In this section, the brief structure of this project is described.

This repository has five directories which means five technical categories in autonomous navigation, Localization, Mapping, SLAM, Path planning, and Path tracking. Each directory includes several directories which has each sample code with different algorithms.

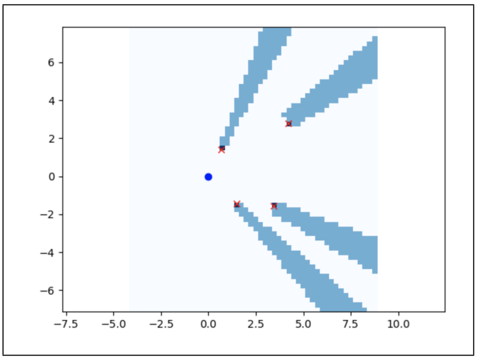
In the following subsections, some algorithms and some simulation examples described.

4.1 Localization

Localization is an ability of a robot to know it's position and orientation with sensors such as GNSS and IMU. In localization, Bayesian Filters such as Kalman filters, histogram filter, and particle filter are widely used[30]. This repository includes some sample codes using these algorithms. Fig.2 shows localization simulation results using histogram filter and particle filter.

4.2 Mapping

Mapping is an ability of a robot to understand surroundings with sensors such as LIDAR and imaging sensor. Robots have to recognize obstacle positions and it' shape for obstacle avoidance. In mapping, Grid map, machine learning algorithms are widely used[30][18]. This repository includes some sample codes using these algorithms. Fig.3 shows mapping simulation results using Grid mapping with 2D ray casting and 2D object clustering with k-means algorithm.


Grid mapping with 2D ray casting

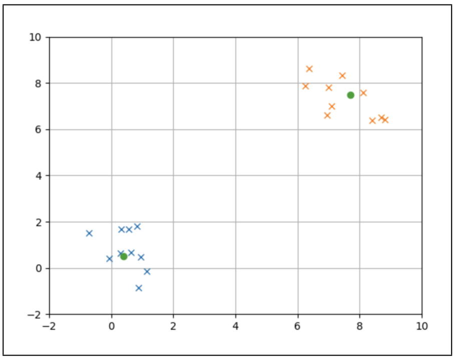
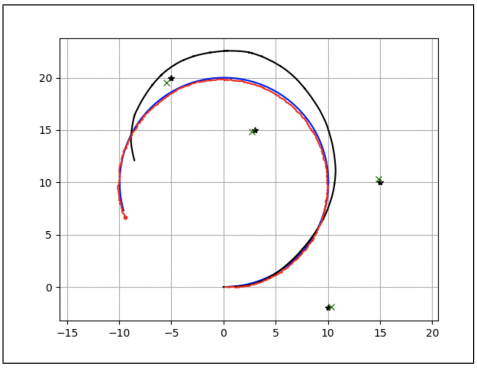

2D object clustering with k-means algorithm

Figure 3: Mapping simulation results


Extended Kalman Filter based SLAM

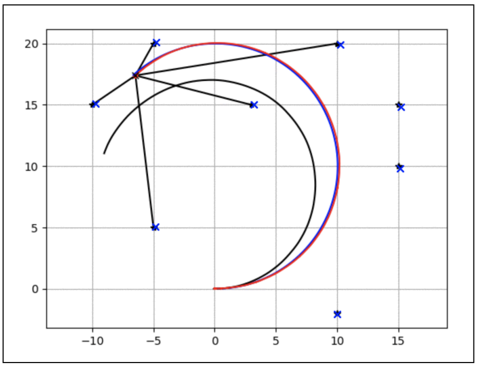

FastSLAM 2.0 based SLAM

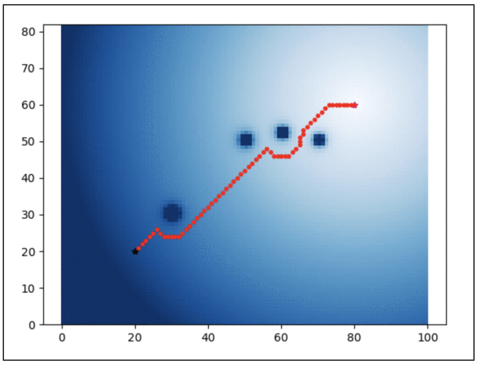
Figure 4: SLAM simulation results

4.3 SLAM

Simultaneous Localization and Mapping: SLAM is an ability of a robot to estimate the pose of a robot and the map of the environment at the same time. SLAM problem is hard to solve, because a map is needed for localization and a good localization is needed for mapping, which is a kind of chicken and egg problem. Popular SLAM solution methods include the extended Kalman filter, particle filter, and Fast SLAM algorithm[30]. This repository includes some sample codes using these algorithms. Fig.4 shows SLAM simulation results using Extended Kalman Filter and results using FastSLAM2.0[30].

4.4 Path planning

Path planning is an ability of a robot to search feasible and efficient path to the goal. The path have to satisfy some constraints based on movement model and obstacle positions and optimize some objective function such as time to goal and distance to obstacle etc. In path planning, dynamic programming based approaches and sampling based approaches are widely used[22]. This project includes some sample codes using these algorithms. Fig.6 shows potential field path planning and LQR-RRT* path planning[27].


Potential Field path planning

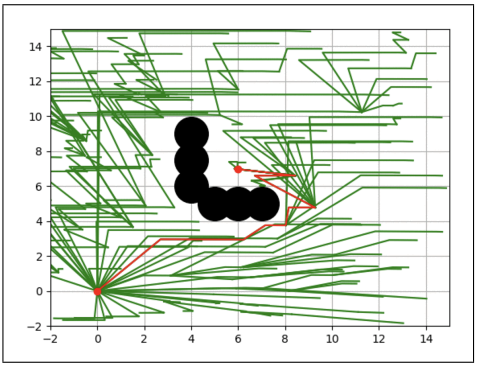
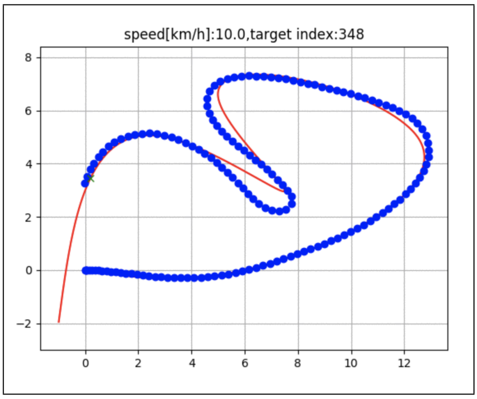
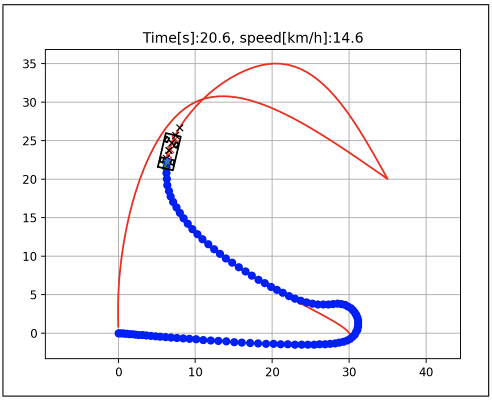

LQR-RRT* path planning[27]

Figure 5: Path planning simulation results



Rear wheel feedback steering control and PID speed control [25]



Iterative linear model predictive control

Figure 6: Path tracking simulation results

4.5 Path tracking

Path tracking is an ability of a robot to follow the reference path from path planning algorithms. The role of the path tracking controller is to stabilize to the reference path or trajectory which has modeling error and other forms of uncertainty. In path tracking, feedback control techniques and optimization based control techniques are widely used[22]. This project includes some sample codes using these algorithms. Fig.6 shows Rear wheel feedback steering control and PID speed control and iterative linear model predictive path tracking control[27].

5 Conclusion and future work

In this paper, I introduced an OSS project which is a Python code collection of robotics algorithms, especially for autonomous navigation. Related works of this project, some key ideas about this OSS project, and brief structure of this repository were described.

The future works of this project are as followed:

- Adding technical and mathematical documentation using Jupyter notebook[1].
- Adding simple image processing sample codes for autonomous navigation only using OpenCV[9].
- Adding simple multi-robots simulations for autonomous navigation.

If readers were interested in these future projects, contributions are welcome. Readers can also support this project financially via Patreon[11].

6 Acknowledgments

We appreciate all contributors: Atsushi Sakai[26], Daniel Ingram[23], Joe Dinius[20], Karan Chala[19], Antonin RAFFIN[29], and Alexis Paques[26].

We also appreciate over 3000 robotics lovers who give a star to this repository in GitHub.

References

[1] Project jupyter. http://jupyter.org/.

[2] Introduction to mobile robotics - ss 2018 - arbeitsgruppe: Autonome intelligente systeme, . http://ais.informatik.uni-freiburg.de/teaching/ss18/robotics/.

[3] Autonomous mobile robots - spring 2018 âĂŞ autonomous systems lab | eth zurich, . http://www.asl.ethz.ch/education/lectures/autonomous_mobile_robots/spring-2018.html.

[4] Programming for robotics - ros âĂŞ robotic systems lab eth zurich, . http://www.rsl.ethz.ch/education-students/lectures/ros.html.

[5] Cvxpy. http://www.cvxpy.org/index.html.

[6] Matplotlib. https://matplotlib.org/.

[7] Mit license. https://opensource.org/licenses/MIT.

[8] Numpy. http://www.numpy.org/.

[9] Opencv. https://opencv.org/.

[10] pandas. https://pandas.pydata.org/.

[11] Atsushi sakai is creating open source software | patreon. https://www.patreon.com/myenigma.

[12] Python. https://www.python.org.

[13] Ros (robot operating system), . http://wiki.ros.org.

[14] Ros navigation stack, . http://wiki.ros.org/navigation.

[15] Scipy. https://www.scipy.org/.

[16] Star history. http://www.timqian.com/star-history/#AtsushiSakai/PythonRobotics.

[17] Artificial intelligence for robotics | udacity. `https://www.udacity.com/course/artificial-intelligence-for-robotics--cs373`.

[18] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

[19] Karan Chawla. Github account. `https://github.com/karanchawla`.

[20] Joe Dinius. Github account. `https://github.com/jwdinius`.

[21] Atsushi Sakai et al. Atsushisakai/pythonrobotics: Python sample codes for robotics algorithms. `https://github.com/AtsushiSakai/PythonRobotics`.

[22] David Gonzalez Bautista, JoshuÃľ PÃľrez, Vicente Milanes, and Fawzi Nashashibi. A review of motion planning techniques for automated vehicles. pages 1–11, 11 2015.

[23] Daniel Ingram. Github account. `https://github.com/daniel-s-ingram`.

[24] Jesse Levinson, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Soeren Kammel, J. Zico Kolter, Dirk Langer, Oliver Pink, Vaughan Pratt, Michael Sokolsky, Ganymed Stanek, David Stavens, Alex Teichman, Moritz Werling, and Sebastian Thrun. Towards fully autonomous driving: systems and algorithms. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, 2011.

[25] Brian Paden, Michal Cap, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. 2016.

[26] Alexis Paques. Github account. `https://github.com/AlexisTM`.

[27] Alejandro Perez, Robert Platt Jr, George Konidaris, Leslie P. Kaelbling, and Tomas Lozano-Perez. Lqr-rrt*: Optimal sampling-based motion planning with automatically derived extension heuristics. pages 2537–2542, 05 2012.

[28] Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

[29] Antonin RAFFIN. Github account. `https://github.com/araffin`.

[30] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005. ISBN 0262201623.