Real-time localization, mapping and navigation for quadruped robots

Y Y Pan, X Q Ding, T Yang, Y Wang, R Xiong and L Jiang

Zhejiang University, Hangzhou, 310027, China

E-mail: panyiyuan@zju.edu.cn

Abstract. Autonomous quadruped robots require localization and mapping for navigation. Different from mobile robots, quadruped robots need local dense maps with more detailed information for motion planning. A key challenge, limited by the capacity of computers on robots, the perception systems have to balance the properties of the speed and accuracy to ensure that robots reach their destinations quickly and safely. In this paper, we propose a complete solution for the autonomous movement of quadruped robots. Localization is achieved by the Extended Kalman Filter which is updated by the results from laser based the ICP algorithm. Dense maps are created by raw ranger sensor data and accumulated based on accurate localization results. And then, dense maps are transformed to the occupancy grid maps which contain free and occupied space information for planning. For implementations, we demonstrate the effectiveness of our approach in a complex outdoor environment.

1. Introduction

With the development of technology, quadruped robots have been very stable in terms of their motion control. In response to future combat needs, intelligence is the most critical factor for robots. To realize the fast and safe autonomous mobile task of the quadruped robot, we have researched the positioning, mapping, and navigation.

LiDAR could provide precise geometric information of the environment, thus has been widely used for localization and mapping in many off-the-road applications such as autonomous delivery cars, inspection vehicles, etc [1,2]. When it turns to quadruped robots, laser-based localization methods need to be fast enough to support real-time ego-motion estimation and local dense mapping for motion planning. In our work, the whole laser map is organized by linked local maps, in which this topological design could boost the computational efficiency of both the localization and navigation processes. What’s more, an Extended Kalman Filter based multi-sensor fusion framework [3] is utilized to further merge the low-frequency localization results with high-frequency IMU information for real-time pose estimation feedback.

The second part is to search for the drivable area to plan over or around the obstacles. In this part, we need to understand the surroundings without the surroundings being drivable by obstacles. Mobile robots equipped with airborne ranging sensors (such as laser ranging, time of flight, and stereo camera sensors) can collect raw distance measurements at each instant and calculate the optimal height of each grid based on multiple repeated observations value according to the localization results.

Path planning is a traditional problem in the robot area, causing decades of researching. The
conventional path planning algorithm regards the robot as a particle, moving in a fully known environment. A result path is a sequence of continuously collision-free waypoints connecting the start point and the endpoint. In real-world implementation, every robot has a physical shape, which is beyond the consideration of the traditional path planning problem. If we directly use the conventional path planning algorithm, the robot may collide the obstacles. To relieve this problem, before applying the path planning algorithm, the shape of the obstacles are inflated, so called the configuration space, to create a safe region between the traversable point of the robot and the physical obstacles, then the usual path generated by the traditional path planning algorithm will be safe. Typically, the quaduped robot can move in any direction, which is independent of its orientation, which can be seen as an omnidirectional robot. Based on its kinematic property, whatever the specific type of motion, quaduped motion or wheeled motion, the configuration space can be created and the path can be generated through any conventional path planning algorithm. For an omnidirectional robot moving on the ground, the dimension of the configuration space is 2. As for the real-world usage, there should be a trajectory planner which runs based on the path result of the path planner and generates the velocity of the robot.

A typical method to generate the velocity of the robot is that the trajectory planner chooses a waypoint amidst all waypoints which can be seen as a temporary jail, and locally generates a velocity profile for execution. However, since the path planner does not consider the current (dynamic) state of the robot, if the goal of the robot changes suddenly, then the position of the waypoints also jump between the adjacent two frames, which leads to the sudden jump of the “local goal”, causing the discontinuity on the velocity profile. Typically, this problem is relieved through optimization on the velocity profile, however, due to the limit computational ability of the embedded device, the substantial optimization methods are not suitable for this problem.

In this paper, we apply the localization, mapping and path planning algorithm in the embedded device, and use a lightweight optimization process to let quaduped robots navigate autonomously.

2. Related Works

During the navigation tasks, quaduped robots need to know where it is. The localization module is to generate the poses of sensors together with the positions of sparse features textures. Employed to minimize the difference between two clouds of points, the algorithm of the Iterative closest point (ICP) can be used to localize robots. Using the approach of structure from motion, robots can estimate global positions and orientations of the camera by matching 3D-3D matches between the compressed model and the captured images. It also can generate accurate localization results quickly.

The dense map can represent the real environment outside the field of view of the sensor. There are three popular dense map representations including occupancy grids [6], OctoMaps [7] and elevation map [8]. Occupancy Grid Mapping divides the environment into square grids of the same size. Each grid has three states: occupied, free, and unknown. So it is straightforward for robots to know which grid can pass. Based on octrees and uses probabilistic occupancy estimation, OctoMaps combine coherent map volumes to reduce the memory requirement. However, it constructs a 3-D representation of the environment, so it is redundant for ground robots to navigate. Therefore, 2.5-dimensional elevation maps are very suitable for motion planning, such as wheeled robots and quaduped robots. Elevation maps can represent the height of each grid in the environment, and also provides sufficient information for the quaduped robots landing point.

The mainstream path planning algorithms discretize the environment and then apply the classical graph search algorithm, such as the A* [9], D* [10] and the focused D* [10] algorithms. A* algorithm is the most classical path planning algorithm which can generate the shortest path. The D* and the focused D* algorithm can create the optimal path amidst the dynamic obstacles. However, for a varying goal point, none of the above-mentioned algorithms consider the continuity of the result path between two time frames, since the shortest paths for different goals are not necessary to be continuous. Although those algorithms are general algorithms that can be used in many complicated environments. However, if the goal point changes, they have to fully re-plan the whole path or re-generate the whole graph.
3. Methods
We propose an efficient approach for quadruped robots to reach artificially set goal quickly and safely without GPU. First, the entire set of sensor equipment needs to scan the global environment to build a globally consistent feature map for localization, and then use the localization results and height information of each grid to build a global map for navigation. In the navigation task, the robot can perform global planning based on the constructed global map, and build a dense elevation map based on laser data for real-time local localization, the overview of the whole frame can be seen in figure 1.

3.1. Offline laser mapping
The offline laser mapping method consists of three modules: pose estimation module, local map maintenance module and loop closing module. It processes laser scans sequentially and merges the information into topological laser map.

When a new laser scan arrives, the pose estimation module will compute its relative transformation to the current local map using Iterative Closest Point (ICP) method [1]. The initial guess for ICP method can be estimated from IMU preintegration [4] between the interval of two sequential laser scans. If the system has not been initialized, i.e., there is no local map for pose estimation, the pose of the new laser scan is set as identity.

The local map maintenance module transforms the processed laser scan to the frame of the current local map, then merges its information into the local map considering the spatial occupancy. The dynamic property of each map point will further be updated according to the heuristic rules proposed in [5]. Then the distance between the origins of the new scan and local map will be checked. If the distance is longer than threshold $\alpha$, which in our case is set as $20m$, the current local map is considered as complete for saving, and a new local map is created with the origin the same as the new scan.

When a local map is completely created, the loop closing module will check its distance with the other existing local maps. If the origins are close enough, ICP method will be applied to their local maps for relative pose estimation. The estimated result will further be utilized to compute the overlap of two local maps, which is considered as the validation of loop closing. If passed, a graph-based optimization is applied on the poses of all the local maps to decrease the drift.

3.2. Topological laser map based localization
During online localization, a sliding map based odometry is performed to accumulate laser map points of the surrounding area. In this situation, only one local map will be maintained, which only contains laser points that are within a radius of $r$ from the robot. This sliding map is sent for localization with near pre-built local maps using ICP method. If success, the relative transformation between the origin of the odometry and the global map will be published as localization result.

The localization results computed with the method of last subsection is time consuming, thus
cannot be directly used as the feedback of motion planning or local terrain perception. We utilize an
Extended Kalman Filter based framework to loosely fuse the localization results as well as the high-
frequency IMU measurements. In this way the output pose estimation results are at the same frequency
of the IMU measurement, and are in the global coordination of the pre-built map, thus could satisfy
the requirement of the other modules.

3.3. Construct the dense elevation map with traversability information
With the localization results and calibration, the relative transformation matrix $T_i^L$ from the inertia
frame to the sensor frame is determined. So all of the points $P_i = (x_i, y_i, z_i)$ collected by the sensor can
be transformed into the inertia frame:

$$P_i^L = T_i^L \circ P_i^L$$ (1)

In the robot-centric grid map, each movement of the robot is accompanied by the movement of the
grid map, and the elevation of each grid can be updated by the latest measurement and data saved in
the corresponding grid according to the Kalman Filter.

The drivable area is to be calculated after obtaining the 2.5-dimension elevation map. Neighbor
height deviation can be utilized to evaluate the traversability of each cell. The deviation can be derived
using the formula below:

$$H_d = |h_{P(x,y)} - h|$$ (2)

Where $h_{P(x,y)}$ is the height of the cell $P(x,y)$ and $h$ is the mean height of all the grids in a N*N window
centered on the cell $P(x,y)$.

3.4. Planning an optimal path
Since the robot only has the updated information of its local environment, although there is a global
path, only the local part of the path is useful to guide the direction of motion for the robot. Hence,
without loss of generality, the tracking process of the global path is equivalent to the choice of the local
destination and the dynamic path planning from the robot’s current position to the local target. In each
iteration, the robot stays at a new position which leads to a new position of local area, then a new goal
is generated. Through such a rolling process we can finish the task.

![Figure 2. The sensing device which can be equipped in the quadruped robots.](image)
4. Experiments

We test our approach on a quadruped robot equipped with sensors. The sensing device is shown in figure 2, which contains 2 Velodyne VLP-16 lasers and an IMU. Moreover, all experiments except training and testing segmentation network were implemented with C++ and run on Robot operating system (ROS) with Intel i7-8700(CPU) and Nvidia 1060(GPU).

Considering the running time, we construct the elevation map which is shown in figure 3 with size of 12m *12m and the resolution is 0.1m. In other words, 120 * 120 = 14400 grids of map will be generated and updated. The result can be saw in figure 3.

In the figure 4, we show that the path is continuous with respect to the time. At the beginning of the movement of the robot, there is no existed path connecting the robot’s current position and the destination, so the algorithm purely executes the conventional shortest path finding algorithm. At the following time, the robot keeps moving, so a path segment is required which starts at the robot’s current position and ends at a waypoint of the already-existed path. Through this process, the whole path is always easy to keep, without huge computational cost of re-planning process.

5. Conclusion

In this paper, we propose an autonomous navigation scheme for quadruped robot, including constructing global maps in advance, real-time position, building a dense elevation map with traversability information, and finally performing path planning in the global map and local map. After several tests in complex outdoor environments, the robot was able to reach the artificially set target successfully. In the future work, we will add complex environmental structures such as trenches and steps to navigation planning tasks to take advantage of traversability of the quadruped robot in multiplex terrains, so that the quadruped robot can be applied to military tasks as soon as possible.

Figure 3. Drivable area results of dense map in real environment. The green area represents the drivable area which has high traversability score, and the red represents the impassable area which has low traversability score.

Figure 4. The planning trajectory. The red arrow is the target point we set. During the planning process, collision detection is also continuously performed.
References
[1] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat 2013 *Autonomous Robots* 34(3) pp 133148
[2] D. Withers and P. Newman 2017 *IEEE International Conference on. IEEE* pp 6233-9
[3] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwar 2013 *IEEE/RSJ international conference on intelligent robots and systems* pp 3923-9
[4] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza 2017 *IEEE Transactions on Robotics* 33(1) p 121
[5] F. Pomerleau, P. Krsi, F. Colas, P. Furgale, and R. Siegwart 2014 *IEEE International Conference on. IEEE* pp 3712-9
[6] Elfes A. 1989 *Computer* 22(6) 46-57
[7] Hornung A, Wurm K M, Bennwitz M, et al. 2013 *Autonomous robots* 34(3) 189-206
[8] Pfaff P, Triebel R, Burgard W. 2007 *International Journal of Robotics Research* 26(2) 217-30
[9] Hart, P. E. Nilsson and N. J. Raphael 1968 *IEEE Transactions on Systems Science and Cybernetics SS- C4* 4(2) 100-7
[10] Stentz, Anthony. 1995 The focussed D* algorithm for real-time replanning. IJCAI. 95