A Generic Approach to Self-localization and Mapping of Mobile Robots Without Using a Kinematic Model

Patrick Kesper[1], Lars Berscheid[1], Florentin Wörgötter[1], and Poramate Manoonpong[2]

[1] Third Institute of Physics - Biophysics, Georg-August-University Göttingen, Friedrich-Hund-Platz 1, 37077 Göttingen, Germany
pkesper@gwdg.de, lars.berscheid@online.de, worgott@gwdg.de

[2] CBR Embodied AI & Neurorobotics Lab, The Maersk Mc-Kinney Moller Institute, University of Southern Denmark, Campusvej 55, 5230 Odense M, Denmark
poma@mmmi.sdu.dk

Abstract. In this paper a generic approach to the SLAM (Simultaneous Localization and Mapping) problem is proposed. The approach is based on a probabilistic SLAM algorithm and employs only two portable sensors, an inertial measurement unit (IMU) and a laser range finder (LRF) to estimate the state and environment of a robot. Scan-matching is applied to compensate for noisy IMU measurements. This approach does not require any robot-specific characteristics, e.g. wheel encoders or kinematic models. In principle, this minimal sensory setup can be mounted on different robot systems without major modifications to the underlying algorithms. The sensory setup with the probabilistic algorithm is tested in real-world experiments on two different kinds of robots: a simple two-wheeled robot and the six-legged hexapod AMOSII. The obtained results indicate a successful implementation of the approach and confirm its generic nature. On both robots, the SLAM problem can be solved with reasonable accuracy.

Keywords: SLAM, Mobile Robots, Hexapod Robot, Probabilistic Robotics, Laser Range Finder, Inertial Measurement Unit

1 Introduction

Solving the Simultaneous Localization and Mapping (SLAM) problem is important for a vast variety of different robotic tasks, e.g. performing autonomous navigation [1, 2] or completing domestic tasks [3, 4]. Probabilistic and other SLAM techniques have been applied to nearly all kinds of robots, e.g. wheeled [5], flying [6, 7], walking [8, 9] or even underwater robots [10]. In contrast to wheeled and flying robots, the application of SLAM to walking robots is scarce. While all these approaches show impressive results, they typically rely either on visual devices [9, 11, 12], on leg/body kinematics [13, 14] or a multitude of sensors including robot proprioceptive sensing, e.g wheel encoders. Therefore, they are difficult to transfer to different robotic systems. In this paper we present our Generic SLAM approach. The approach is based on a probabilistic SLAM algorithm and relies only on two portable sensors, an inertial measurement unit (IMU) and a laser range finder (LRF). It can be applied to different robotic systems. We have evaluated the performance of this approach on a wheeled robot and a six-legged walking robot in real-world experiments.

2 Materials and Methods

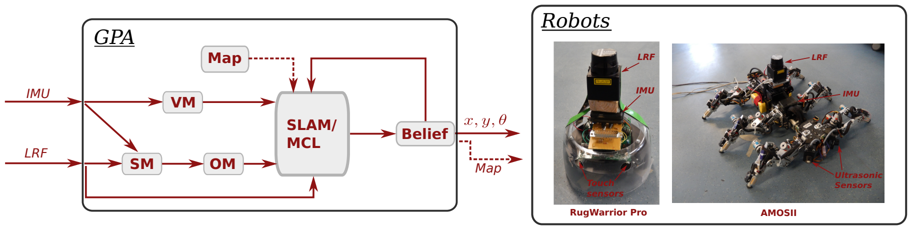

Fig. 1. This figure summarizes the Generic SLAM approach. It receives measurements from an inertial measurement unit (IMU) and a laser range finder (LRF) as an input to the SLAM/Monte-Carlo localization (MCL) algorithm, which updates the belief of the robot. Based on this belief, the most likely state and map can be estimated. The approach is tested on two different robots, the RugWarrior Pro and AMOSII.

The Generic SLAM approach is summarized in fig. 1. It receives measurements from IMU and LRF as inputs. Here, two cases are possible. On the one hand the IMU output can be used to compute the translational and rotational velocity of a robot, which is then given to the Velocity Model (VM). This is appropriate, if the IMU data has a low noise level, e.g. when the movement of the robot is mostly linear. On the other hand, a new state estimate can be computed based on the acceleration values in combination with scan matching (SM). The result is then used as an input to the Odometry Model (OM). This approach is especially useful, if the IMU data exhibits a strong background noise. Scan matching is able to compensate for this noise at the cost of increased computational complexity. One of these approaches must be chosen manually. This SLAM/MCL algorithm utilizes the generated control action and the output of the LRF to recursively update the state distribution (belief) of the robot. Based on this belief, the most likely state can be estimated. The individual modules of the approach are explained as follows:
Odometry Model (OM): The Odometry Model uses two consecutive states x_{t-1} and x_t to estimate the movement of a robot. The control action is given through $u_t = (\overline{x}_{t-1}, \overline{x}_t)^T$. Based on the difference between both states the rotations δ_{rot1} and δ_{rot2}, as well as the translation δ_{trans} can be computed by applying geometry. To account for model and measurement errors Gaussian noise is

added to these variables. The variance of this noise is chosen to reflect the noise characteristics of the utilized IMU. See [15] for more details.
Velocity Model (VM): The Velocity Model follows a different approach compared to the OM. While the OM relies on relative motion information, the VM directly utilizes the translational and rotational velocity of a robot. Thus, the control action is given by $u_t = (v_t, \omega_t)^T$, where v_t denotes the translational velocity of the robot and ω_t the rotational velocity. Again, Gaussian noise is added to these velocities to account for measurement errors. The velocity model assumes these values to be constant over a given time span Δt. In this case the robot moves on the arc of a circle. See [15] for more details.
Scan Matching (SM): It is possible to obtain odometry information with a laser range finder by utilizing a technique called *Scan Matching*. The basic idea of scan matching is to evaluate the relative change between two consecutive scans to obtain an estimate of the corresponding movement of the robot. Here, a technique called *Polar Scan Matching (PSM)* is chosen ([16]). The PSM procedure does not yield any direct information about the velocity of the robot. It only estimates a position. Thus, it has to be used in combination with the OM.
SLAM/MCL: The SLAM/MCL algorithm updates the belief of the robot based on the inputs from OM/VM and the previous belief. Both procedures are implemented using a Particle Filter leading to the so called *FASTSlam* algorithm [15]. To run FASTSlam, a measurement probability $p(z_t|x_t)$ and a state transition probability $p(x_t|u_t, x_{t-1})$ must be known. Possible choices for $p(x_t|u_t, x_{t-1})$ can be derived from the described motion models, *Likelihood Fields* are used to model $p(z_t|x_t)$ [15]. Furthermore, an method to update maps according to the measurement and state of a particle is required. Here, we use *Occupancy Grid Maps* in combination with *Bresenham's Line Algorithm* [17].

3 Experiments and Results

Here, we use two different robot platforms to demonstrate the general use of our approach and to evaluate its performance, the RugWarrior Pro [18] and the six-legged hexapod AMOSII [19]. To be able to solve the SLAM problem the Hokuyo URG-04LX-UG01 Laser Range Finder and the x-IMU are used as a portable sensor modules. For the experiments on the real robots three different courses with varying difficulty and complexity are set up (fig. 2). In all experiments the robots start at the depicted location and traverses the corresponding course until reaching the end. For all experiments the size of one grid cell is 0.05 m x 0.05 m.
RugWarrior Pro: For the RugWarrior Pro the VM was used. The results of all three courses are shown in fig. 2. During the experiment for course a), the robot was steered to the left on purpose to test the functionality of the algorithm. Indeed, the curve can be seen in the computed paths, both for SLAM and MCL. Furthermore, the map created by the SLAM algorithm matches the measured dimensions. However, the computed map appears to be quite noisy. This was due to the unstable mounting of the LRF and IMU on the robot. The MCL algorithm provides a more accurate path than the SLAM procedure due

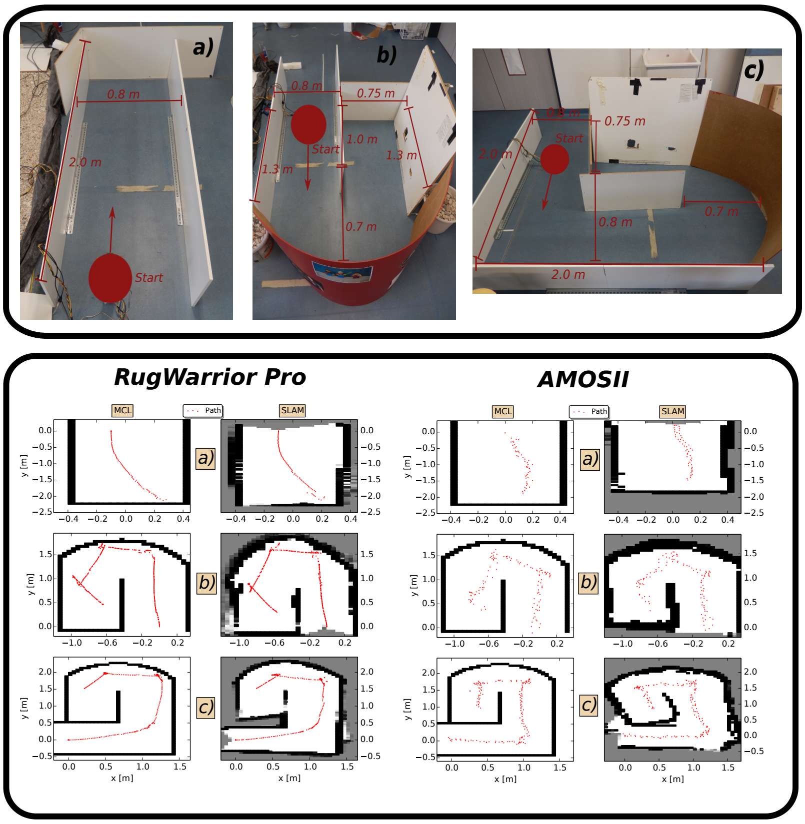

Fig. 2. This figure shows results of the MCL and GridSLAM algorithm for RugWarrior Pro and AMOSII traversing the three different courses a), b), and c).. The size of one grid cell is 0.05m x 0.05m. A video illustrating examples of the real robot experiments can be seen at http://manoonpong.com/TAROS2015/supple.mp4

to the higher noise sensitivity of SLAM. Overall, the accuracy of the SLAM algorithm can be estimated with ±0.05 m. The results of course b) are similar to course a). Again, the MCL algorithm provides a slightly less noisy path. But the difference is surprisingly small considering the additional turns of the robot. The achieved accuracy is ±0.1 m. Course c) is the longest and most complex setup. However, the accuracy of the results is similar to course a) and b). The accuracy of the computed map is about ±0.1 m. In summary, regardless of the complexity and length of the course, the SLAM procedure is able to track the path of the robot with reasonable accuracy. The obtained maps match the dimensions of the real environments. Furthermore the MCL and SLAM algorithm are able to successfully deal with erroneous acceleration, velocity and range measurements.
AMOSII: Now, the Generic SLAM approach was tested on the AMOSII robot. Scan matching was applied to compensate the increased movement noise. The

same testing procedure and environments as utilized for the RugWarrior Pro were used. The results are very similar to the ones obtained from the RugWarrior Pro (fig. 2). However, the accuracy is slightly decreased. This is due to the stronger noise present in the IMU data. The MCL algorithm is able to track the correct path of the robot on all courses. In particular, it reproduces the erratic movement of AMOSII, which can also be seen in the video provided as supplementary material. The calculated positions do not lie on a straight line, but are oscillating left and right, exactly like the real motion of AMOSII. The results of the SLAM algorithm are consistent with the MCL results. Deviations are about 0.1 m at maximum. The maps created by the SLAM algorithm provide a rough, but usable representation of the real environment. The differences in wall positions are between 0.1 m and 0.2 m. When looking at course c), it becomes apparent that uncertainties accumulate during the SLAM algorithm. This seems reasonable, because the SLAM algorithm has no 'ground truth' available to completely compensate the uncertainties. In MCL this is possible due to the provided map of the environment.

4 Conclusion

RugWarrior Pro: The real-world experiments with the RugWarrior Pro are successful. However, in recent research more impressive results are presented. In [20] a wheeled robot traversed an approximately 7 km long path with many dynamic obstacles. In [5] a map of a 4 m x 6 m environment is created with an error of less than 0.07 m. But all of these setups rely on visual devices or wheel encoders and are generally equipped with more and better hardware. Thus, it is reasonable, that using only two sensory inputs on the inferior RugWarrior Pro sacrifices accuracy.
AMOSII: The accuracy of the obtained results is similar to other approaches. In [8] a humanoid robot maps a 4 m x 7 m environment with a maximum error of 0.1 m. In [11] a humanoid robot walks in a circle with a radius of 0.75 m. The resulting map and trajectory again have a maximum error of 0.1 m. Lastly, in [12] a 0.5 m x 0.5 m environment with rough terrain is mapped with similar accuracy. However, in all of these works the robots used visual information and/or kinematic models. In this paper, we showed that we are able to achieve the same results with our minimal and generic approach.
In large, open environments without any objects the LRF does not return any usable information. Consequently, the accuracy of the MCL and SLAM algorithm decreases drastically. Both algorithms work well in indoor environments due to the abundance of objects and walls. In this paper, a minimal and generic SLAM implementation relying only on a LRF and an IMU is proposed and successfully tested on wheeled and legged robots. Consequently, a possible next step could be experiments with other kinds of robots. In particular, flying robots for indoor navigation are an interesting choice due to their wide availability and versatility.
Acknowledgment: This research was supported by BCCNII Göttingen with grant number 01GQ1005A (project D1).

References

1. D. Wooden, M. Malchano, K. Blankespoor, A. Howardy, A.A. Rizzi, and M. Raibert. Autonomous navigation for bigdog. In *2010 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4736–4741, May 2010.
2. W. Burgard, A. B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. Experiences with an interactive museum tour-guide robot. *Artificial Intelligence*, 114(1 - 2):3 – 55, 1999.
3. J. L. Jones, N. E. Mack, D. M. Nugent, and P. E. Sandin. Autonomous floor-cleaning robot, August 27 2013. US Patent 8,516,651.
4. P. E. Sandin, J. L. Jones, et al. Lawn care robot, January 21 2014. US Patent 8,634,960.
5. I. Wieser, A. V. Ruiz, M. Frassl, M. Angermann, J. Mueller, and M. Lichtenstern. Autonomous robotic slam-based indoor navigation for high resolution sampling with complete coverage. In *Position, Location and Navigation Symposium-PLANS 2014, 2014 IEEE/ION*, pages 945–951.
6. J. Müller. *Autonomous navigation for miniature indoor airships.* PhD thesis, Universitätsbibliothek Freiburg, 2013.
7. S. Grzonka, G. Grisetti, and W. Burgard. A fully autonomous indoor quadrotor. *Robotics, IEEE Transactions on*, 28(1):90–100, 2012.
8. N. Kwak, O. Stasse, T. Foissotte, and K. Yokoi. 3d grid and particle based slam for a humanoid robot. In *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, pages 62–67.
9. A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(6):1052–1067, 2007.
10. A. Kim and R. M. Eustice. Real-time visual slam for autonomous underwater hull inspection using visual saliency. *IEEE Transactions on Robotics 2013.*
11. O. Stasse, A. J. Davison, R. Sellaouti, and K. Yokoi. Real-time 3d slam for humanoid robot considering pattern generator information. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 348–355.
12. D. Belter and P. Skrzypczynski. Precise self-localization of a walking robot on rough terrain using parallel tracking and mapping. *Industrial Robot: An International Journal*, 40(3):229–237, 2013.
13. M. Bloesch, M. Hutter, M. A. Hoepflinger, S. Leutenegger, C. Gehring, C. D. Remy, and R. Siegwart. State estimation for legged robots-consistent fusion of leg kinematics and imu. *Robotics*, page 17, 2013.
14. P. Wawrzyński, J. Możaryn, and J. Klimaszewski. Robust estimation of walking robots velocity and tilt using proprioceptive sensors data fusion. *Robotics and Autonomous Systems*, 2014.
15. S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics.* MIT press, 2005.
16. A. Diosi and L. Kleeman. Fast laser scan matching using polar coordinates. *The International Journal of Robotics Research*, 26(10):1125–1153, 2007.
17. J. E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems journal*, 4(1):25–30, 1965.
18. J. L. Jones. RugWarriorPro: Assembly guide. *AK Peters Ltd*, 1999.
19. P. Manoonpong, U. Parlitz, and F. Wörgötter. Neural control and adaptive neural forward models for insect-like, energy-efficient, and adaptable locomotion of walking machines. *Frontiers in neural circuits*, 2013.
20. R. Kümmerle, M. Ruhnke, B. Steder, C. Stachniss, and W. Burgard. Autonomous robot navigation in populated pedestrian zones. *Journal of Field Robotics*, 2014.