Hazard Detection & Positioning System for Asteroid Navigation

By Jack Freeman, ECE '22

Introduction

Consider a robot navigating a room. If the room is mapped out the robot might be able to navigate around obstacles and avoid getting stuck, but what if the room wasn't known? A potential solution to this problem could be Simultaneous Localization and Mapping (SLAM). SLAM is the computational problem of navigating and creating a map of an unknown environment while navigating the room. The unique advantage that SLAM provides is location data within the environment that the robot is navigating. This article explains the history and mathematics of SLAM as well as highlighting a case study and the application of SLAM to Team Outer Space

Background

History

SLAM was first pioneered in the 1980's in by R.C. Smith and P. Cheeseman (Smith, Cheeseman, et. al. 1986). At this time SLAM was barely a concept, the idea of representing and estimating uncertainty and location probabilities in a room. Over the last 40 years, with the improvement of sensors, faster processing, and advancing research, SLAM has become one of the lead solutions towards air, space, and local navigation of unknown environments.

Mathematical Explanation

SLAM relies on a sensor, typically a camera or LiDAR (Light Detection and Ranging hardware) for environmental data. The robot begins by taking input data of the environment and estimating the distance to its surroundings. Input data from a camera would be a series of pictures; for a LiDAR it would be a series of multiple distance measurements. These distances measure how far the LiDAR is from a wall, like an invisible tape measurer. Based on these distances the bot can create a map-like rendering of its current location. Depending on the needs of the project and the sensors available this process can create either a 2D or 3D map of the environment. The next step is for the robot to move to a new location close to the original location, where the process of taking input data and measuring distances is repeated.

Now with two different data points, Bayes' rule is applied to update the most probable location of the robot in the new position. Bayes rule is a simple equation that is used to measure conditional probability. Bayes' rule, which is shown below helps determine conditional probability. A sample example might be when drawing cards from a deck. If event A is drawing a Spade card and event B is drawing an Ace the following would be true: When one card is drawn, the probability of drawing a spade given that the card is an Ace is ¼. This is equal to the probability of drawing an Ace given that the card is a spade (1/13) times the probability of drawing a Spade (¼) divided by the probability of drawing an Ace (1/13).

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

Figure 1. Bayes rule for conditional probability

While the basic premise of Bayes rule is simple, we can expand this equation to measure the previously mentioned location estimation as shown in Figure 1. In this equation ut represents the sensors, ot represents the sensor observations, t is the discrete time steps, xt is the location of the robot, and mt is the environment map.

$$P(m_t|x_t, o_{1:t}, u_{1:t}) = \sum_{x_t}\sum_{m_t} P(m_t|x_t, m_{t-1}, o_t, u_{1:t})P(m_{t-1}, x_t|o_{1:t-1}, m_{t-1}, u_{1:t})$$

Figure 2. SLAM location estimation equation using Bayes' rule

While the general concept of SLAM is straightforward, there are many issues that begin to arise. One critical part of this approximation is the accuracy of the measured step. Inaccuracies or drift of the robot quickly propagate so a reliable and accurate IMU (Inertial Measurement Unit) is necessary to measure the robot movement. An IMU measures changes in acceleration which can be used to determine changes in velocity/position; however, this sensor can be very inaccurate especially when the bot does not move in a straight line. In addition to error with the IMU, if the odometry of the robot is inaccurate, the robot position data will introduce significant error to the location estimation and mapping. Since each new location estimation is based on the last one this can become significant quickly. An example of this would be if your following directions but mess up one of the earlier directions. Consider a series of directions where you take the Third exit, then take 2 lefts, go straight through a stoplight, and then take a right. If instead of taking the third exit you take the second exit. You can imagine that the final destination might not be anywhere near the intended destination. The solution to minimize this error is to have the robot complete a circular loop, this is known as loop closure. Loop closure helps the robot know where it is since it already has identical - or near identical sensor readings from the location. During this process the map is updated along with the position. Depending on the accuracy of the software packages a loop could be as big as several city blocks or as small as a loop around a lamp (Angeli et. al., 2009).

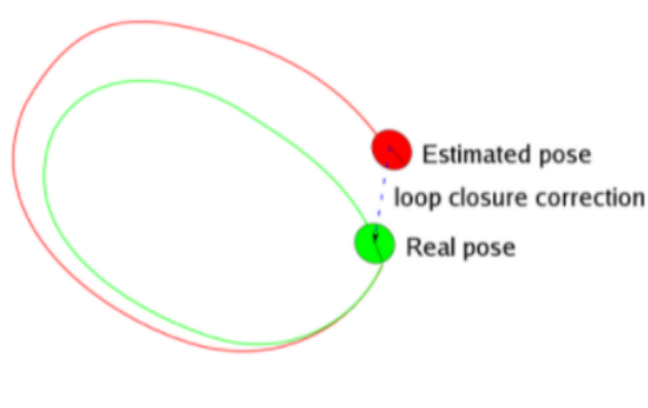

Figure 3. Loop closure example. Green represents the actual path that the robot takes. Red represents the robot thinks it takes (Source: ENSTA)

Case Study

One specific application of SLAM was a project known as ORB-SLAM recently published in 2016 by Raúl Mur-Artal, Juan D. Tardós, J. M. M. Montiel and Dorian Gálvez-López (DBoW2). ORB Slam utilizes a singular monocular stereo or RGB-D Camera to compute camera trajectory and reconstruct a 3D environment in real-time. ORB-SLAM implements SLAM in the standard approach as mentioned earlier with the addition of feature tracking. Feature tracking within SLAM is the process of finding and tracking specific areas of an image such as a specific color, pattern, or shape. The addition of feature tracking helps identify unique areas of images which improves the accuracy of matching these areas between time steps. An example of some of these features is shown below in Figure 4.

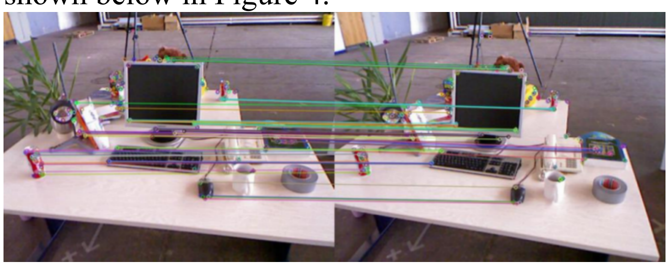

Figure 4. Matching features between two images (Source: Qiang Li et. al. 2020)

In a step by step process, an image is captured and stored in grayscale. Then the image is parsed and key points are identified. This same process is repeated for another image. Then the two key points between the images are compared and the features are matched. The example above shows an example where these significant points are shown between two photos. Notice that the similar objects aren't necessarily distinct colors since the images are grayscale. Next the images are used to reconstruct a 3-D structure which again relies on the matching significant points. The previous steps are repeated as more images are taken and the 3-D structure is updated with the new points. After a set amount of time, distance, or some other benchmark the results can be visualized using the images taken, constructed graph, or feature points.

Applications towards our specific project

The broad goal of the Team Outer Space project is to navigate between two points on an asteroid while being able to accurately determine the current location of the robot during this path. In our original approach to solving this problem, we planned to rely on visual SLAM to create a local map of our environment to determine the current position of the robot The specific package we were previously looking at to accomplish this goal is the Matlab vSLAM package. This package provides functions to assist with processes' such as extracting features, matching features, reconstructing a 3-D structure, estimating motion from the camera, visualizing results, and managing data. Our specific case application would have also relied on the IMU to determine motion, not just the camera. In addition to using SLAM to explore and create a map, we previously were going to rely on the map output from SLAM to create a path planning algorithm. One challenge that SLAM created for our specific application is that SLAM is most often used to avoid vertical protruding obstacles. Since our project goal is modeled towards exploring an asteroid we will be faced with craters, potholes, and other obstacles that might not be picked up with the camera. In order to detect and avoid these craters our project has added an additional LiDAR sensor to detect craters. Since these features will need to be included in the SLAM map we will need to incorporate these distances into our SLAM algorithm. This process would have looked similar to that of loop closure, where the local mapping is updated before the general iterative process of SLAM continues. In our final product, our team chose not to implement SLAM because of time constraints and the nature of our problem. We didn't need to create a map of our environment as long as the location is known so SLAM was not key to solving the issue. Instead of SLAM we are solely using the Camera, LiDAR, and IMU to track current position and distance traveled. The front3 Jack Freeman Senior Handbook Project - Tech Note Final Draft 3/31/22 LiDAR is still being included to map negative distances thus solving all of our potential issues with craters.

Future of SLAM

One of the issues that arises utilizing SLAM for asteroid navigation is the limited computational power. This causes current active SLAM projects to often move extremely slowly. The mars rover for example has a maximum speed of less than 0.1 mph, which the time of slow speed of SLAM attributes to (Zhang et. al. 2018). With computational power increasing yearly, out-of-core parallel SLAM could be possible in the next coming years. Out-of-core parallel computation is the process of splitting up a workload into multiple chunks that can separately be solved. The idea behind this would be that the SLAM generated map could be split among cores or potentially by solving multiple stages of SLAM in parallel entirely (Cadena et. al. 2016).

References

1. Adrien Angeli, Stéphane Doncieux, Jean-Arcady Meyer, David Filliat. Visual topological SLAM and global localization. International Conference on Robotics and Automation (ICRA), 2009, Kobe, Japan. pp.4300 - 4305, ff10.1109/ROBOT.2009.5152501ff. Ffhal-00652601f

2. Bo Zheng, Zexu Zhang, "An Improved EKF-SLAM for Mars Surface Exploration", International Journal of Aerospace Engineering, vol. 2019, Article ID 7637469, 9 pages, 2019. https://doi.org/10.1155/2019/7637469

3. C. Cadena et al., "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age," in IEEE Transactions on Robotics, vol. 32, no. 6, pp. 1309-1332, Dec. 2016, doi: 10.1109/TRO.2016.2624754.

4. M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte and M. Csorba, "A solution to the simultaneous localization and map building (SLAM) problem," in IEEE Transactions on Robotics and Automation, vol. 17, no. 3, pp. 229-241, June 2001, doi: 10.1109/

5. M. Z. Zia et al., "Comparative design space exploration of dense and semi-dense SLAM," 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 1292-1299, doi: 10.1109/ICRA.2016.7487261.

6. Smith, Randall & Self, Matthew & Cheeseman, Peter. (1986). Estimating Uncertain Spatial Relationships in Robotics. Autonomous Robot Vehicles. 1. 435-461. 10.1109/ROBOT.1987.1087846.

7. T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): part II," in IEEE Robotics & Automation Magazine, vol. 13, no. 3, pp. 108-117, Sept. 2006, doi: 10.1109/MRA.2006.1678144.

8. Qiang Li et al 2020 J. Phys.: Conf. Ser. 1693 012068