Reactive Gimbal for Quadruped SLAM Visual Feature Enhancement

Pujith Kachana
Team Members: Paul Kim, Luke Barnes
Graduate Mentor: Vishwa Ramkumar
Faculty Advisor: Ye Zhao
Georgia Institute of Technology
pkachana3@gatech.edu

Abstract

As robots continue to perform more complex navigation tasks to interact with the physical world, it is ever important that robots can quickly and accurately learn about their surroundings and determine their relative configuration. This problem is referred to as simultaneous localization and mapping, or SLAM, and has seen many advancements and successes in recent years. One of the prevailing SLAM techniques is visual SLAM, in which a camera is used to capture images of the environment for the robot to build a map and localize upon. However, for dynamic robots with frequent accelerations, it can often be difficult to capture high-quality images to facilitate this process. This problem is exacerbated in the case of legged robot which generate an assortment of vibrations with every step. This study seeks to improve the image stream quality for SLAM systems on legged robots. We introduce a specialized gimbal to alleviate motion blur for legged robot imaging, along with an intensive analysis of different approaches for optimized performance.

1 Overview

Robot navigation has seen a rise in practical usage in recent years. From robot vacuums to autonomous cars, many modern day robotic advancements rely heavily on accurate navigation. For most applications of robot navigation, robotic agents must keep track of their environment and location in a process known as simultaneous localization and mapping, or SLAM. This study is part of a larger project from the LiDAR lab known as SLAMBox, which aims to develop a proprietary SLAM system that combines various sensor modalities and model representations to achieve robust navigation for legged robots. This system is currently being deployed and tested on a Mini-Cheetah quadruped robot. Fundamentally, all SLAM algorithms collect sensory measurements and form perceptions about their environment to generate a dynamic map, which they in turn use to determine their relative configuration. A common sensor modality for SLAM is cameras, giving rise to a variation known as visual SLAM. In most visual SLAM algorithms, the robot uses incoming image streams to compute feature and landmarks to append to its map, generating a spatial representation of areas of interest. This study seeks to improve the quality of image stream for the SLAMBox system to enable more accurate mapping and localization. Specifically, this project's objective is to make a cost effective gimbal that is particularly optimized for reduction of motion blur in legged robots, thereby enhancing the fidelity of the robot's SLAM system.

1.1 Landmarks for Visual SLAM

Visual SLAM, or vSLAM, primariliy uses camera readings to localize and map. There are many variations of vSLAM that each use different types of camera, different feature spaces, and different feature densities, but all vSLAM algorithms are limited by the quality of the images they receive. A poor image stream from the cameras will result in poor feature detection and localization, degrading the overall performance of the SLAM system. This problem can sometimes be solved by upgrading to higher resolution cameras or incorporating sensor fusion between multiple cameras, but there are certain cases where the fundamental problem lies beyond the camera itself. In the case of the quadruped SLAMBox, the vibrations caused by the quadruped's erratic gait introduce motion blur into the camera system regardless of the camera being used. The resulting images gathered from the camera while the quadruped is in motion are of poor quality, significantly affecting the quantity and accuracy of features detected. This makes it harder to find landmarks for the SLAM system to localize upon, hindering the correspondences between the visual and LiDAR frames. In this case, the system has to be resistant to external disturbances to the imaging process.

1.2 Gimbal

Gimbals are camera stabilizers that dampen sudden accelerations to the camera frame, thereby providing a smoother image stream. Gimbals have various forms and differ widely in their designs, some using various sensors and motors and other being entirely mechanical. For the purpose of offsetting motion blur in legged robotic imaging system, a gimbal with high reactivity and tolerance to sustained vibrations is required (Rajesh and Kavitha, 2015). The Mini-Cheetah requires a double-axis gimbal in oder to stay level to the ground plane due to its vibrations on the roll and pitch axes. The yaw axis is left as a degree of freedom as a yaw rotation signifies the robot is turning. In the particular case of legged robots, the gimbal must be quick and precise enough to offset vibrations caused by the robots gait. Most studies on gimbals for robots generally target aerial or wheeled robots, and are less common for legged robots, so the following sections include a thorough analysis of various approaches for legged robot gimbals.

2 Gimbal Design

2.1 Mechanical System

The prototype system we developed is a 2-axis, 3D printed gimbal. One motor is placed in the center of the gimbal behind the camera to offset roll and another is placed in line with the camera to offset pitch.

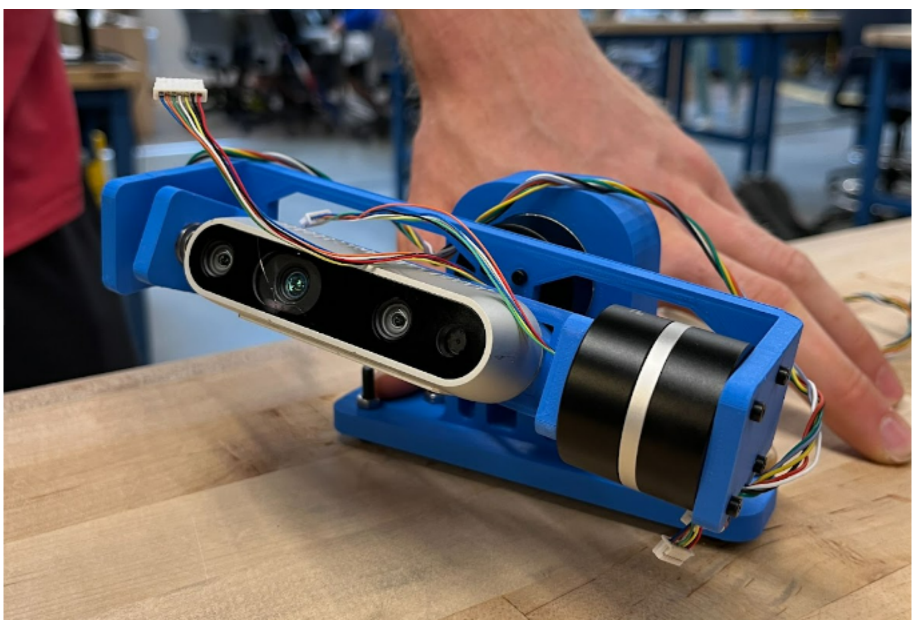

Figure 1: 2-Axis Gimbal

2.2 Electrical System

The gimbal is driven by a Teensy 4.1 and powered by the SLAMBox. An MPU6050, a 6-axis IMU, was used to collect accelation forces on the camera. The initial prototype used an ODrive as the motor driver and two 3-phase brushless DC motors for the gimbal actuation. We were later able to configure three L298N motor drivers to drive the two 3-phase brushless gimbal motors by using 3 half H-bridges for each motor across two L298 drivers. This significantly cut down the cost of the gimbal system since the $250 Odrive was replaced with a $12 contraption. This makes the gimbal much more accessible but also more difficult to tune, as it does not include convenient libraries for software support like the ODrive.

2.3 Software System

The software system consists of modules to collect and process IMU data (Dejan, 2019), compute motor velocity values for optimal gimbal actuation, and interface with the motor driver through SPI. The exact implementation details of the software to drive the gimbal required much deliberation and analysis since the algorithm must be efficient, robust, and accurate to compensate for the erratic movement of the robot. Initially, a reactive closed-loop control system was implemented, and predictive algorithms were later explored.

3 Reactive Gimbal

3.1 Sensor Fusion for Control Input

In order for the gimbal to properly utilize IMU data, the measurements must first be processed into anglular values relative to the ground plane, as this will serve as input to the control system. There are multiple ways to derive this offset angle, each with their own benefits. A simple solution is to use dead reckoning, or computing relative IMU angles by integrating gyroscope angular velocity values over a time period. This can lead to very precise angle measurements in the short term. Assuming constant angular velocity, an integral is not necessary, as the change in angle can be derived through multiplying the time step and angular velocity with this assumption. This makes it efficient to compute. However, this can lead to long-term bias and drift accumulation, as the starting 0 angle is the only tangible angle given to the algorithm. Another approach is to compute Euler's angles using component vectors of acceleration due to gravity. Since we know the orientation and magnitude of gravitational acceleration, it can be used as a reference for the orientation of the IMU. This approach is prone to noise, especially if the body is moving and experiences other forces besides gravity. This approach lacks short term accuracy but performance does not drift over time. Also, with this approach we cannot measure yaw since it is on the same axis as gravity.

To reconcile the benefits and drawbacks of these two approaches, a filter can be applied on the measurements. An initial consideration was a Kalman Filter, as it combines noisy or indirect measurements of hidden variables with predicted model values to compute a state estimate (Becker, 2016). This can effectively fuse the dead reckoned and trigonometrically computed angular values together to yield a more precise measurement. Another consideration was the complementary filters which, like a Kalman filter, can be used to fuse sensory data from two noisy sources. The complementary filter applies a low and high pass filter on both measurements before combining them to offset low and high frequency variance, respectively, creating an estimate with the most reliable aspects of both measurements combined. The complementary filter was ultimately used for the prototype due to its simplicity and relatively efficient performance (Dejan, 2019).

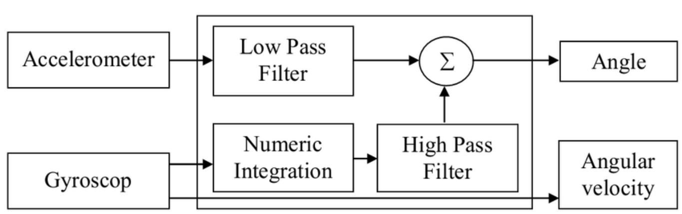

Figure 2: IMU Complementary Filter

3.2 PID System

A traditional approach to real time control is a PID system. PID is a closed-loop control system that takes a sensor input and target value to produce an effector output based on the offset using the inputs proportional, integral, and derivative values. More clearly, three values, the P, I, and D components, are calculated and combined to create an effector output to reach the target sensory value. The P, or proportional component, is used to approach the desired sensor value with respect to the magnitude of the offset. This is the simplest component of PID and is enough to create a performative system in most cases, although it can sometimes result in steady state error. The I, or integral component, keeps track of past sensor values and integrates over previous offsets. This can help combat steady state error by detecting persistent offset values over time, although it can sometimes result in overshoot. Lastly, the D, or derivative component, predicts rate of change in error to allow smoother actuator commands to desired sensor value. This combats the overshoot problem, and the P, I, and D coefficients can be tuned to create a complete and robust control system. PID systems use some or all of these three elements in combination to create a control system to trigger actuators to reach a target value. In the case of this project, the PID sensor will the processed IMU measurements and the target value will be an angle of 0 from the ground plane (Rajesh and Kavitha, 2015). For the purpose of the 2-axis gimbal, the I component may be unnecessary, as we do not anticipate any steady state error. To derive motor velocity values from IMU reading, a PD loop will be used for this gimbal. However, it can be seen that one loop is not sufficient to complete the system. Although velocity values can be derived with one loop, motor inputs require current. Another layer of computation is required to translate desired motor velocity to current.

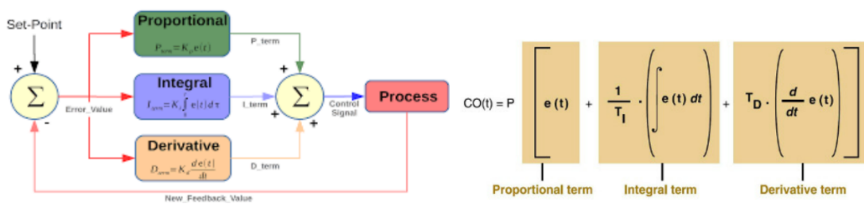

Figure 3: PID Diagram

3.3 Cascade Control System

The system that was ultimately implemented for this study was a cascade control loop, which is essentially nested PID loops where the output of one loop feeds into another. The original system input is the IMU angle data, which is then used to calculate a desired velocity for the motor. This velocity reading is then used as an input for another PID loop internal to the motor that increases current until that velocity is reached. This system allows for flexible and reactive control of motor current given the IMU offset angle. To elaborate, the ODrive first measures the motor's position error and feeds it into the PID, using only the P coefficient. Then, there is also a velocity controller, which can be used separately to set a target velocity. In our case of setting a target position, the position command is converted to a velocity command and is given as an input to the velocity PID, which uses the PI coefficients, Finally, this value is converted into a current command, which it given as an input to the current PID using the PI coefficients. After one iteration of this PID control loop is completed, the encoder sends position feedback, which is then reused as an input to the position command and derived to an input for the velocity command.

3.4 Results

Below are graphs from tests performed on the assembled gimbal, where 0 is the camera orientation and 1 is the gimbal tracking. It can be seen that at slow speeds, the gimbal almost perfectly compensated the angular displacement of the camera measured by the IMU. However, this performance degraded at higher speeds that are more representative of the Mini Cheetah's vibrations. This can be solved by better parameter tuning and weight distribution to dampen vibrations.

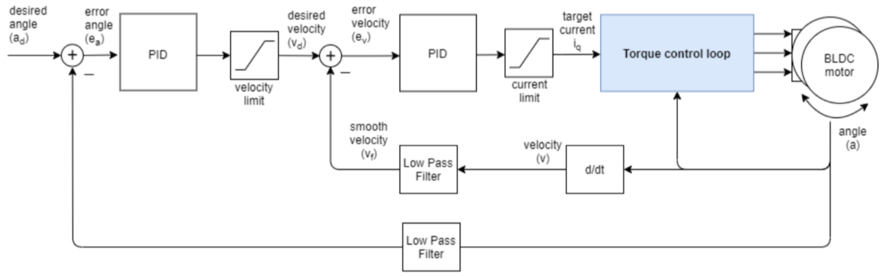

Figure 4: Cascade-Control Diagram

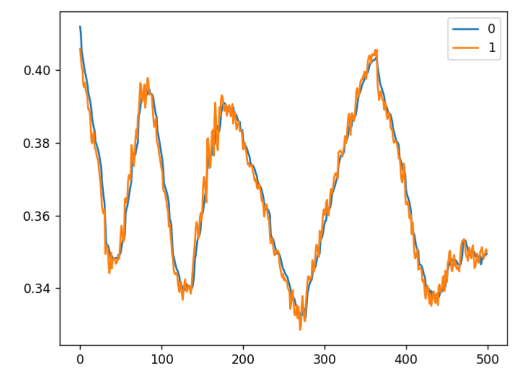

Figure 5: Slow Movement

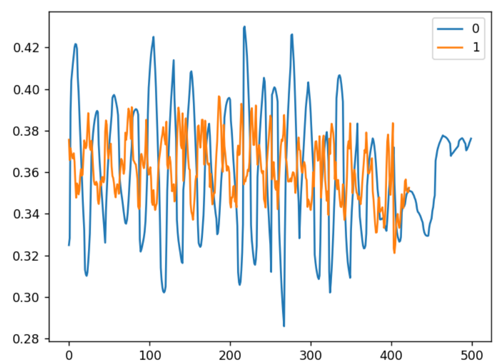

Figure 6: Fast Movement

4 Predictive Gimbal

As an extension to our current gimbal, we explored the possibility of predicting disturbances in the systems to preemptively offset vibrations. All legged robots, including the Mini-Cheetah used for the SLAMBox, have a generally cyclic gait pattern which generates the largest and most frequent vibrations. If the gait of a robot can be utilized to drive more informed gimbal actuations, it may be possible to increase the reactivity of a legged robot gimbal.

4.1 Quadruped Gait Analysis

An initial approach that was considered was to analyze the robot's gait and the corresponding forces generated by the ground contacts. This would allow for the modelling of the vibration around the camera system, therefore allowing (Gujarathi and Bhole, 2019). However, this is a complicated approach with various parameters, as the vibrations are a function of the robot's speed, terrain, and other confounding variables. It is an over-analysis of the system for a localized task, so this approach was not pursued.

4.2 Vibration Analysis

The next approach considered was to limit the analysis to only the local vibrations around the camera. This reduces a level of indirection from, as the vibrations are being directly observed rather than treated as a product of other variables. The first method used for this approach was autocorrelation, which is an efficient algorithm for wave periodicity estimation. It measures self-similarity of waves with their shifted counterparts and samples the superimposed wave forms in increments of wave shifts. This sample peaks once every period, allowing the dominant frequency to be computed (SpeechZone, 2016). Specifically, this draws out the major vibrations caused by the robot's ground contacts. This method was not effective for the legged robot gimbal, however, as there were various sources of vibrations. Reconciling the single dominant frequency was insufficient for a robust gimbal.

An alternate method for vibration analysis is the Fast Fourier Tranform, or FFT. FFT is a common signal transformation algorithm that shifts the time domain of acceleration values into frequency domain, thereby extracting the frequency spectrum of a systems vibrations. This enables more flexibility and robustness than autocorrelation since any particular frequency can be targeted and isolated. The drawback of this method is its computational expense, introducing timing and hardware constraints for high-frequency gimbals.

4.3 LSTM Model

The last approach was to use machine learning to predict the next set of acceleration values given a history of recent accelerations. The appropriate model for this problem is the long short-term memory model, or LSTM, which have feedback connections to detect temporal relationships and discover long term dependencies within the data. A generic LSTM model was used to test acceleration prediction capabilities against sample vibration data, but the accuracy was low and sporadic. Satisfactory accuracy never reached even after hyper-parameter tuning. This is most likely due to the noisy and random vibrations in the legged robot system. The recorded vibrations from the Mini-Cheetah performing various actions are shown below.

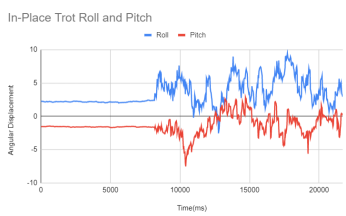

Figure 7: Trot

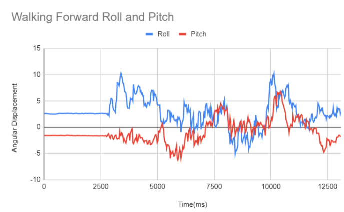

Figure 8: Walk

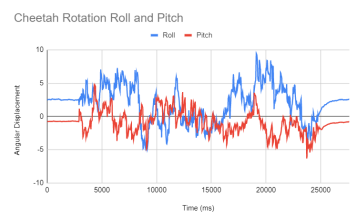

Figure 9: Turn

5 Conclusion

The difficulty of the SLAMBox gimbal system comes from the strict performance constraints established by the legged robot and the SLAM system. Traditional gimbal technology is well established, but reconciling the high-frequency, low-amplitude vibrations of the robot with the high resolution requirements of the SLAM perception system requires a highly reactive and precise gimbal. The budget constraints introduce an additional challenge. Overall, it can be seen through the performance benchmarks that the quadruped gimbal prototype has demonstrated sufficient reactivity to improve the performance of the SLAMBox. Although the precision and tracking of the gimbal degrades at high speeds over large distances, the relatively small vibrations caused by the quadruped fall within the gimbals range of reactivity. It is still possible that the cyclic nature of a legged robots gait can be exploited for better performance , although the attempted methods of preemptive gimbaling have not displayed any improvements. Future plans for this project include further pursuing predictive gimbal algorithms and refining the mechanical structure of the gimbal prototype.

Note: The gimbal has been further tuned since from the creation of this report. The report will be updated with new performance metrics.

References

Becker, A. (2016). Kalman filter tutorial.

Dejan (2019). Arduino and mpu6050 accelerometer and gyroscope tutorial.

Gujarathi, T. and Bhole, K. (2019). Gait analysis using imu sensors. *IEEE*.

Rajesh, R. J. and Kavitha, P. (2015). Camera gimbal stabilization using conventional PID controller and evolutionary algorithms. *IEEE*.

SpeechZone (2016). Autocorrelation for estimating f0.

A Appendix

A.1 Notes of Study

Notes:
https://docs.google.com/document/d/17qbc3Eje4kQNM2wM97HafHMr2RLSzSD775S0SuOXCxk/edit?usp=sharing