REAL-TIME STATE ESTIMATION FOR AGGRESSIVE DRIVING AUTONOMOUS GROUND VEHICLES

A Thesis
Presented to
The Academic Faculty

By

Dominic Pattison

In Partial Fulfillment
of the Requirements for the Degree
Bachelor of Science in the
School of Computer Science

Georgia Institute of Technology

May 2018

REAL-TIME STATE ESTIMATION FOR AGGRESSIVE DRIVING AUTONOMOUS GROUND VEHICLES

Approved by:

Dr. James M. Rehg
School of Interactive Computing
Georgia Institute of Technology

Dr. Evangelos Theodorou
School of Aerospace Engineering
Georgia Institute of Technology

Date Approved: May 3, 2018

ACKNOWLEDGEMENTS

I would like to thank Brian Goldfain and Paul Drews for their countless hours of help and mentorship throughout my time working on the Autorally project. Additionally, I'd like to thank Dr. James Rehg for his support, along with the other faculty members involved with the Autorally project: Dr. Evangelos Theodorou, Dr. Panagiotis Tsiotras, and Dr. Byron Boots for the use of the lab, the lab's hardware, and their graduate students' time. Finally, I'd like to thank all the additional help and teaching I received from other members of the project including Grady Williams, Kamil Saigol, and Keuntaek Lee.

TABLE OF CONTENTS

LIST OF FIGURES

SUMMARY

State estimation is an integral part of many robotic systems. It is particularly important in the realm of high performance ground vehicles where systems must be real-time, robust, and accurate. This project aims to tackle the problem of improved state estimation reliability, robust to GPS dropout, applied to the control of an aggressively driving autonomous vehicle in an unstructured environment. To accomplish this goal, the system combines components for visual SLAM, wheel odometry, IMU integration, and incremental inference on a factor graph. The result is a system that is tolerant to short periods of GPS dropout allowing a vehicle to safely handle the situation.

CHAPTER 1
INTRODUCTION

State estimation in robotics refers to estimating the position, velocity, and orientation along with other desired parameters of a robot. This is done by collecting information from sensors on the robot and performing some optimization to find what state is the most likely. More formally, given all previous observations $Z_{t:1}$, we are interested in finding X^*, the robot state X that maximizes the posterior probability, $P(X^*) = argmax_X P(X|Z_{t:1})$.

State estimation is ubiquitous in all of robotics and is especially important in the frontier of research in high performance robotic control. One such example is the Autorally project which operates a 1/5th scale autonomous ground vehicle to research high performance aggressive driving in unstructured rally-like environments [1]. In this application, it is critical for state estimation to be real-time, accurate, and robust. Introducing speed into the problem of control means all aspects, from the algorithms to the hardware, must adhere to a strict real-time constraint. Precise maneuvers necessitate a high degree of accuracy, and robustness affords safety and diverse operating conditions. There has been some success with control using limited or no state estimation [2, 3]. However, these approaches currently lack the generalizability, predictability, and reliability needed for safety critical systems like autonomous vehicles. This paper aims to meet the demands of high performance autonomous ground vehicles and will do so by incorporating state of the art components into one state estimation system. Specifically, this paper uses incremental inference on a factor graph for trajectory smoothing to integrate data from visual simultaneous localization and mapping (VSLAM), wheel odometry (WO), inertial measurement unit (IMU), and global position system (GPS). The contribution of this paper is the development of a robust, multimodal, real-time, factor graph-based state estimation method well suited for systems with highly nonlinear dynamics that might break motion models of more traditional methods. Experimental validation is shown on a 1/5th scale autorally platform in a high-speed, off-road, racing environment. Driving datasets are

publicly available through the Autorally project along with a public implementation of all software used [1].

[1] https://github.com/AutoRally/autorally.

CHAPTER 2

BACKGROUND

2.1 Inference Methods

The canonical approach to state estimation is to fuse measurements from an inertial measurement unit (IMU) and GPS into an Extended Kalman Filter (EKF) [4, 5, 6, 7, 8]. In general, Kalman filters make two assumptions: the system is linear and the measurement error is Gaussian noise. EKF extends this to nonlinear systems by linearizing the prediction function around the current estimate. However, this extension only works well for systems whose prediction functions are approximately linear [4]. Another variant is the Unscented Kalman filter (UKF) which uses the Unscented Transform to select a minimal set of points to propagate through the nonlinear update function [4]. Monte Carlo Localization is yet another method which maintains a set of particles representing the distribution, transforms each through an update function, then performs importance sampling [9]. Both the UKF and Monte Carlo Method can model nonlinear systems; however, they both require an explicit update function. In aggressive driving scenarios, formulating an explicit update function analytically would require difficult to estimate information such as the coefficient of friction at every time step [1]. Further, in these models, there is no natural way to model complex constraints such as loop closures in VSLAM systems.

In aggressive driving scenarios, especially with large wheel slip in drifting maneuvers, it is particularly important to model the system's nonlinearity. In a recent notable example, Funk et al. developed an autonomous vehicle for on-road aggressive driving [10]. Their state estimation system incorporated similar ideas, but integrated them into an EKF. They do not operate the vehicle at the limits of handling due to the limitations of this model. This in not an appropriate limitation for the Autorally project which specifically attempts to works at the limits of performance of the hardware.

An alternative framework, used in this project, is trajectory optimization using factor graphs. A factor graph is a bipartite graph whose vertices are divided into factors and variables and whose edges join factors and variables [11]. In factor graphs, each factor represents a factor in the factorization of the joint probability density function where the neighborhood of the factor node is the set of variables involved in that particular factor of the function. Factor graphs are a natural representation of the SLAM or robot localization problem because the observations are treated as givens and the graph only encodes variables that we are interested in [11]. Factor graphs are a very powerful representation because it is possible to solve exactly or approximate the full joint probability distribution. There has been lots of work done on the implementation and theory of incremental inference on factor graphs which aids in real-time applications [12, 13, 14]. One example fuses measurements from GPS, IMU, and a VSLAM pipeline into a factor graph-based optimization. This is a similar project as the work presented here; however, the optimization was done offline rather than in real-time and the VSLAM pipeline was designed specifically for the application of crop monitoring [15].

2.2 Visual SLAM

A vision component has become an integral part of many modern state estimation algorithms [7, 8]. Visual odometry (VO) refers to estimating camera motion from consecutive images, and VSLAM is an extension of this problem that involves creating a map of the environment while localizing within that map using images from a camera system. VO can be divided into two types: direct and feature based methods [16, 17]. Direct methods rely on directly comparing pixel values to estimate the motion of features. Direct methods tend to be very fast, skipping the costly feature extraction step, but they also tend to be less accurate. Feature-based methods rely on extracting features and tracking those features over consecutive frames to estimate motion. Feature-based methods tend to be slower due to feature extraction but also tend to more accurate. The Semi-Direct Visual Odometry algorithm attempts to combine the two methods [16, 18]. This approach has been shown to be very effective, but it lacks a loop closure component meaning systems relying

on it are susceptible to long-term drift.

ORB SLAM is a feature based visual SLAM system that that utilizes oriented Brief (ORB) features [19, 20]. This methods relies on efficient feature extraction and matching (due to the feature descriptor selected), robust mapping including aggressive culling of map points and keyframes, and global pose optimization. The mapping algorithms attempts to only keeps relevant features and key frames. This is meant to keep the growth of the map in check which in turn keeps the run-time in check. The global pose optimization provides long-term drift free navigation when loop closures are encountered. This project uses a modified version of ORB SLAM because of its attempts to trade off between accuracy and speed, ideal in a real-time system.

CHAPTER 3

METHODS

The state estimation system presented in this paper utilizes both state of the art work and standard components. The system integrates IMU and GPS (when available) measurements along with integrate wheel odometry estimates and transforms computed from VSLAM or VO. As seen in figure 3.1, this make the system complete, integrating every sensor and piece of information available to the vehicle. Additionally, the system is designed in a modular fashion utilizing the Robot Operating System for intermodule communication [21].

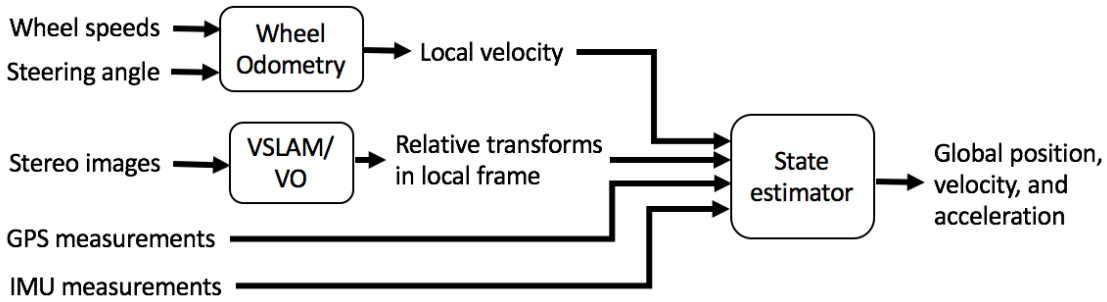

Figure 3.1: The inputs and outputs of the state estimation system and its subsystems.

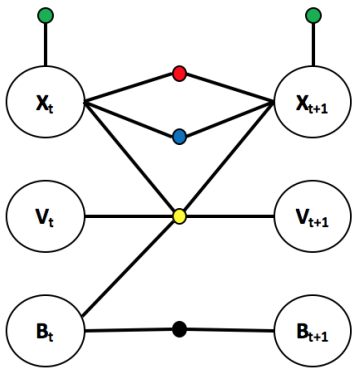

Figure 3.2: A typical time slice of the structure of the factor graph used. The graph vertices are separated into factors and variables. GPS factors are shown in green, visual odometry is shown in red, wheel odometry is shown in blue, an IMU factor is shown in yellow, and a constant IMU bias factor is shown in black.

IMU integration uses a state of the art implementation from GTSAM which is detailed in [22] and [23]. The integrated IMU measurements create a factor in the graph that constrains the previous position, velocity, and bias variables along with the next position and velocity variables. GPS measurements are incorporated in a very simple manner where the measurement is transformed into a local coordinate frame using an initial measurement. This then creates a unary constraint on a position variable.

The wheel odometry model is a very simple Ackermann steering model. It takes the current steering angle and four wheel speeds, measured from Hall effect sensors, and computes an estimated forward velocity and yaw rate at approximately 70 hz (where yaw is the vehicle's orientation around the axis perpendicular to flat ground) [1]. This estimated velocity is then integrated over a given time step to calculate a relative transform between the pose at the beginning of the time step and the end of the time step. This relative transform then acts as a constraint on two consecutive position variables. It is important to note that the model makes the assumption that the vehicle has no lateral component (no sliding) to its velocity and that there is no wheel slip. Although these assumptions are clearly broken in this particular use case, the model creates an estimate that is accurate enough over the time scale to be useful. This suggests with a more descriptive model, like the one described in [24], the results would be even better.

The VSLAM component of the system relies on the open source implementation of ORB SLAM released by the authors [19, 20]. The library was then modified to output the relative transform between each image frame. These transforms are then composed into one relative transform that describes the change of the vehicles position between two given times. As with the wheel odometry, this transform is interpreted as a constraint between two consecutive position variables. The time at which GPS messages are received is not perfectly in sync with the frame rate of the cameras. Therefor, the closest frames are taken at the boundary, and it is assumed that the error associated with this assumption is mostly negligible and accounted for in the covariance of the constraint. A more generalized approach would be to estimate the trajectory in continuous time [15].

Figure 3.2 shows a typical time slice of the factor graph. The graph vertices are separated into factors and variables. X_t, X_{t+1} represent pose, including position and orientation, V_t and V_{t+1} represent velocity, and B_t and B_{t+1} represent IMU bias. The state that we want to estimate is the current pose, velocity, and IMU bias at time $t+1$. The product of all factors associated with previous variables ($X_{t:1}$, $V_{t:1}$, and $B_{t:1}$) is proportional to the posterior probability distribution

$$P(X_{t:1}, V_{t:1}, B_{t:1}|Z_{t:1}) \propto \prod_{i=1}^{t} \ell(X_i, V_i, B_i; Z_i) = \ell(X_{t:1}, V_{t:1}, B_{t:1}; Z_{t:1}) \tag{3.1}$$

where Z_i is the set of measurements associated with the variables at time ii, and $\ell(.)$ is a likelihood function, proportional to the probability function. Thus, at time $t+1$, we have a new set of measurements to incorporate into an updated estimate. Specifically, we have a GPS factor, $l(X_{t+1}; G_{t+1})$, an IMU factor, $l(X_{t+1}, V_{t+1}; X_t, V_t, B_t, I_t)$, a VO factor, $l(X_{t+1}; X_t, VO_t)$, a WO factor, $l(X_{t+1}; X_t, WO_t)$, and an IMU bias factor, $l(B_{t+1}; B_t)$. The product of these new factors is the incremental update to be applied,

$$\begin{aligned} \ell(X_{t+1}, V_{t+1}, B_{t+1}; X_{t:1}, V_{t:1}, B_{t:1}, Z_{t+1}) = \ell(X_{t+1}; G_{t+1}) * \ell(X_{t+1}, V_{t+1}; X_t, V_t, B_t, I_t) \\ * \ell(X_{t+1}; X_t, VO_t) * \ell(X_{t+1}; X_t, WO_t) \\ * \ell(B_{t+1}; B_t) \end{aligned} \tag{3.2}$$

The updated distribution is then the product of these new factors and the previous factors

$$\begin{aligned} P(X_{t+1:1}, V_{t+1:1}, B_{t+1:1}|Z_{t+1:1}) &\propto \ell(X_{t:1}, V_{t:1}, B_{t:1}; Z_{t:1}) \\ &* \ell(X_{t+1}, V_{t+1}, B_{t+1}; X_{t:1}, V_{t:1}, B_{t:1}, Z_{t+1}) \\ &= \ell(X_{t+1:1}, V_{t+1:1}, B_{t+1:1}; Z_{t+1:1}) \end{aligned} \tag{3.3}$$

From the previous state, IMU measurements are integrated to get an initial estimate of the next state which is used as a starting point for the optimization to find X^*_{t+1}, V^*_{t+1}, and B^*_{t+1} that maximize equation 3.3, in other words, find the most probable state.

For evaluating the system presented in this paper, the AutoRally platform was used [1]. The

Autorally platform is a fully autonomous vehicle developed at the Georgia Institute of Technology for performing research on the cutting edge of high performance robotic control and perception. It is based on a 1/5th scale remote-control car that is fitted with an electric motor and onboard computing and sensing which includes stereo cameras, a high precision IMU, and a GPS setup to receive RTK corrections.

Figure 3.3: (Left) The AutoRally platform with onboard sensing and computing. (Right) The AutoRally platform performing a high speed maneuver.

The platform is driven autonomously and manually on both of the Georgia Tech Autonomous Racing Facilities (GTARF) on the main campus and the Georgia Tech Research Institute (GTRI) campus. GTARF at the main campus is an oval dirt track that is eight feet wide and 225 feet long. GTARF at the GTRI campus is a more complex dirt track that is 15 feet wide and 550 feet long. These facilities offer a controlled environment for testing the AutoRally platform at the limit of its performance. On both tracks, the vehicle can regularly reach speeds in excess of 20 miles per hour.

Figure 3.4: (Top) The GTARF at the main campus of Georgia Tech. (Bottom) The second GTARF at the GTRI campus.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Results

The following results come from recorded data in which the system was performing nominally with GPS and IMU. This data was recorded over a three minute test run at the main campus GTARF at speeds of 5-7 m/s (this is just past the slip point of the wheels). The nominal system is considered to be ground truth in the following experiments. The first test was to evaluate the accuracy of both wheel odometry and VO when input into the factor graph. Both of these components compute relative transforms from one position variable to the next, so their error can be found by comparing the computed transform and the ground truth transform. The results of this comparison are show in figures 4.1 and 4.2.

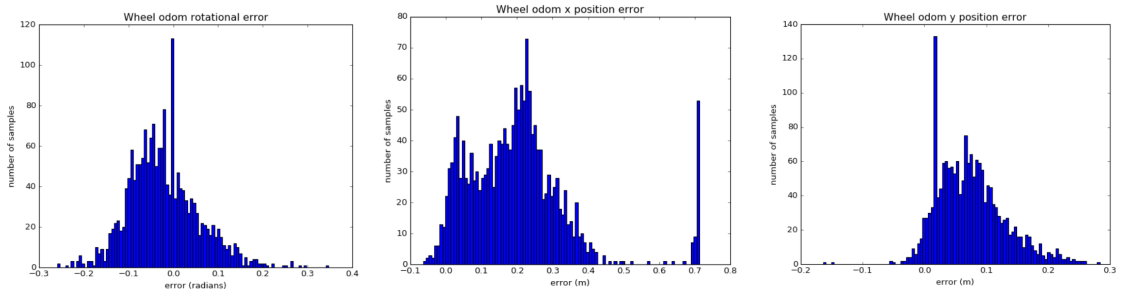

Figure 4.1: (Left) The yaw, (center) x, and (right) y error over an wheel odometry factor compared to ground truth.

4.2 Discussion

In figure 4.1, we see the expected bias in the estimates. In the test run, the vehicle is traveling counter-clockwise which causes the estimates to be consistently under for y and over for yaw by not accounting for wheel slip. This is expected given such a simplistic physical model. There has been other work that expands on odometry modeling that would greatly increase the accuracy

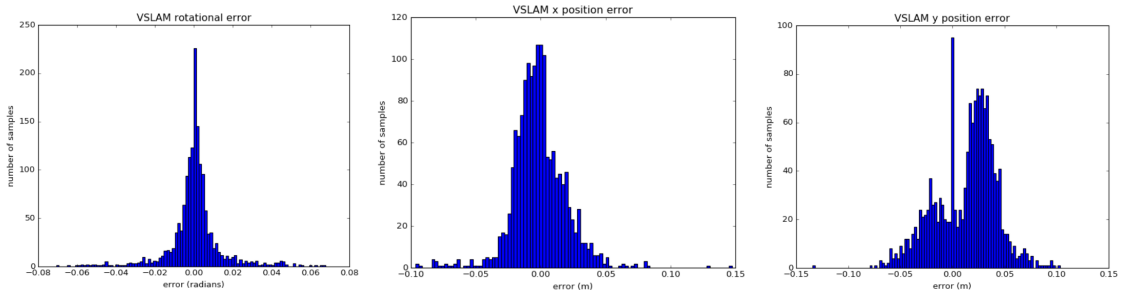

Figure 4.2: (Left) The yaw, (center) x, and (right) y error over an visual odometry factor compared to ground truth.

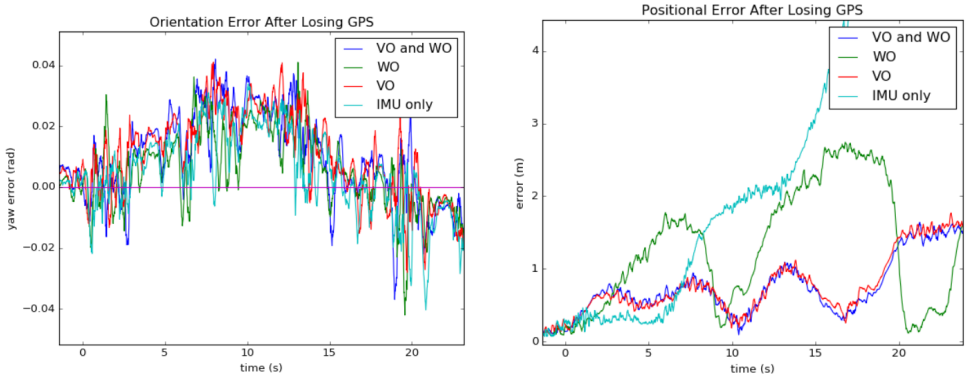

Figure 4.3: (Left) The yaw error and (right) the x-y positional error of the system integrating information when no GPS is available. WO is wheel odometry and IMU, VO is visual odometry and IMU, WO and VO is all three, and IMU only includes data from the IMU.

and reduce bias [25, 26]. On the other hand, figure 4.2 shows that the VSLAM pipeline gave significantly better estimates in all three measurements. However, what it gains in accuracy it loses in reliability.

Significant effort was put into making the VSLAM pipeline more robust. Neither of the open-source implementations of SVO or ORB SLAM was able to produce reliable estimates, even after tuning. This is thought to be because the environment is a particularly challenging one for VO/VSLAM for several reasons. The textures on the track are very self-similar. The track surface itself has very few reliable features. This means that the only consistent features that can be extracted frame to frame are far from the vehicle. Additionally, given the low perspective of the cameras along with the high speeds, a large portion of the image moves a substantial amount between frames. Coupled with a very short stereo baseline, the only reliable features cannot be used

for positional estimation. Although VSLAM is largely considered a theoretically solved problem, the experience on this project shows there is still much work to be done on the engineering and reliability side of the problem.

Figure 4.3 shows the error of the system when operating without GPS. As expected, the orientation error is quite stable and is dominated by the IMU given that angular rates can be directly measured with low noise. When looking at the positional error over time, it is more obvious the effect the other measurements have on the estimate. Because the wheel odometry measurements are biased, they quickly pull the estimate away. However, by directly measuring the velocities VO and WO cap the growth of the linear velocity errors. This manifests as much slower growth of error over time. VO allows the system to stay within approximately one meter of the ground truth for 15 seconds. Potentially, this allows for safer operation where the system can come to a stop autonomously.

CHAPTER 5
CONCLUSION

This paper proposed a comprehensive system for state estimation built on the idea of integrating all available information into a single estimate. By itself, the results were not reliable enough to confidently operate without GPS. However, this project has laid the foundation for a modular state estimation system. Future work can capitalize on this and replace subsystems with more accurate and reliable ones, thus improving the system as a whole.

REFERENCES

[1] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1433–1440.

[2] P. Drews, G. Williams, B. Goldfain, E. A. Theodorou, and J. M. Rehg, "Aggressive deep driving: Model predictive control with a cnn cost model," *arXiv preprint arXiv:1707.05303*, 2017.

[3] Y. Pan, C.-A. Cheng, K. Saigol, K. Lee, X. Yan, E. Theodorou, and B. Boots, "Agile off-road autonomous driving using end-to-end deep imitation learning," *CoRR*, vol. abs/1709.07174, 2017.

[4] S. J. Julier and J. K. Uhlmann, *A New Extension of the Kalman Filter to Nonlinear Systems*. 1999, vol. 3068.

[5] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[6] S. Rezaei and R. Sengupta, "Kalman filter-based integration of dgps and vehicle sensors for localization," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 6, pp. 1080–1088, 2007.

[7] T. Chu, N. Guo, S. Backn, and D. Akos, "Monocular camera/imu/gnss integration for ground vehicle navigation in challenging gnss environments," *Sensors*, vol. 12, no. 3, p. 3162, 2012.

[8] S. B. Kim, J. C. Bazin, H. K. Lee, K. H. Choi, and S. Y. Park, "Ground vehicle navigation in harsh urban conditions by integrating inertial navigation system, global positioning system, odometer and vision data," *IET Radar, Sonar and Navigation*, vol. 5, no. 8, 2011.

[9] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, vol. 2, 1999, 1322–1328 vol.2, ISBN: 1050-4729.

[10] N. Funk, N. Alatur, R. Deuber, F. Gonon, N. Messikommer, J. Nubert, M. Patriarca, S. Schaefer, D. Scotoni, and N. Bnger, "Autonomous electric race car design," *arXiv preprint arXiv:1711.00548*, 2017.

[11] F. Dellaert, "Factor graphs and gtsam: A hands-on introduction," 2017.

[12] M. Kaess, A. Ranganathan, and F. Dellaert, "Isam: Incremental smoothing and mapping," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1365–1378, 2008.

[13] M. Kaess, V. Ila, R. Roberts, and F. Dellaert, "The bayes tree: An algorithmic foundation for probabilistic robot mapping," in *Algorithmic Foundations of Robotics IX: Selected Contributions of the Ninth International Workshop on the Algorithmic Foundations of Robotics*, D. Hsu, V. Isler, J.-C. Latombe, and M. C. Lin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 157–173, ISBN: 978-3-642-17452-0.

[14] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "Isam2: Incremental smoothing and mapping using the bayes tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.

[15] J. Dong, J. G. Burnham, B. Boots, G. Rains, and F. Dellaert, "4d crop monitoring: Spatio-temporal reconstruction for agriculture," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3878–3885.

[16] C. Forster, M. Pizzoli, D. Scaramuzza, and Ieee, "Svo: Fast semi-direct monocular visual odometry," in *2014 Ieee International Conference on Robotics and Automation*, ser. IEEE International Conference on Robotics and Automation ICRA. New York: Ieee, 2014, pp. 15–22, ISBN: 978-1-4799-3685-4.

[17] F. Fraundorfer and D. Scaramuzza, "Visual odometry part ii: Matching, robustness, optimization, and applications," *Ieee Robotics and Automation Magazine*, vol. 19, no. 2, pp. 78–90, 2012.

[18] C. Forster, Z. C. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," *Ieee Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2017.

[19] R. Mur-Artal, J. M. M. Montiel, and J. D. Tards, "Orb-slam: A versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[20] R. Mur-Artal and J. D. Tardos, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. PP, no. 99, pp. 1–8, 2017.

[21] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: An open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, Kobe, Japan, p. 5.

[22] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "Imu preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," in *Robotics: Science and Systems 2015*, 2015.

[23] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2017.

[24] Y. Pan, K. Saigol, and E. A. Theodorou, "Belief space stochastic control under unknown dynamics," in *2017 American Control Conference (ACC)*, pp. 3764–3770.

[25] Y. Pan, X. Yan, E. A. Theodorou, and B. Boots, "Prediction under uncertainty in sparse spectrum Gaussian processes with applications to filtering and control," in *Proceedings of the 34th International Conference on Machine Learning*, D. Precup and Y. W. Teh, Eds., ser. Proceedings of Machine Learning Research, vol. 70, International Convention Centre, Sydney, Australia: PMLR, 2017, pp. 2760–2768.

[26] C. You and P. Tsiotras, "Vehicle modeling and parameter estimation using adaptive limited memory joint-state ukf," in *2017 American Control Conference (ACC)*, pp. 322–327.