Localized Imaging and Mapping for Underwater Fuel Storage Basins – 18364

Jerry Hsiung *, Andrew Tallaksen *, Lawrence Papincak *, Sudharshan Suresh *, Heather Jones *,
William Whittaker *, Michael Kaess *
* Carnegie Mellon University

ABSTRACT

Current methods for inspection of spent nuclear fuel storage basins involve lowering a single camera for visual inspection of walls and other structures. We present a localized inspection solution where the images are automatically annotated by localization information and a 3D model of the inspected area is generated. The system consists of an underwater sensor pod containing a stereo pair of cameras, light source, inertial measurement unit, and a pressure sensor. The sensors are time synchronized to provide precise measurements. We describe both the sensor pod and the algorithms that keep the pod localized. Preliminary results from in-air testing of a prototype are presented.

INTRODUCTION

Spent nuclear fuel is stored underwater in fuel storage basins, which provide cooling and radiation shielding. This includes the concrete L-Basin at the Savannah River Site. L-Basin must be regularly inspected to ensure structural integrity. Based on observation of exposed exterior walls of L-Basin, there are known cracks through which water seepage occurs, but it is difficult to identify the corresponding sites of these cracks inside the basin. The current inspection method involves lowering a single camera, either on a stick or a string, to visually inspect the basin walls. Unfortunately, the current method suffers from several drawbacks. It is difficult for human operators to keep track of the precise location of a camera, especially when (in the case of a camera on a string), their control over the camera is limited. The presence of algae growing in L-Basin also complicates the inspection task, as an autofocus camera will tend to focus on floating algae, resulting in blurry images of the intended inspection targets. Given these shortcomings, a better inspection solution is needed.

This paper presents a localized inspection solution designed for the L-Basin inspection application featuring a stereo pair of cameras. The system uses a robotics technique, simultaneous localization and mapping (SLAM) to build up a map of the inspected region while determining the sensors' position and orientation relative to this map. An inertial measurement unit is used to improve the positioning. Precise synchronization between the cameras and the inertial measurement unit is an especially advantageous feature of the solution.

The system features cameras with large sensors relative to the camera size, making them particularly suited to underwater imaging. Using data from both cameras taken at the same time, a 3D reconstruction of the viewed scene can be built. The system is also designed such that it will not focus on algae.

By precisely tracking camera positions, the system can determine what areas were covered and whether there are gaps in the data. The constructed surface maps can also be stored and compared across multiple inspections, making it easy for operators to track change over time. This is particularly valuable when trying to understand basin structural integrity.

The objective of the presented system is to provide high fidelity 3D models of surfaces (e.g. concrete walls) or objects to be inspected. The 3D model will allow for visual inspection by a remote operator, automatic anomaly detection (e.g. cracks in concrete), and automatic detection of changes when compared to prior models from earlier inspections.

In what follows, we first describe the sensor pod system, followed by the algorithms to provide localized imaging and mapping. We then show results from preliminary in-air experimental evaluation, before concluding.

DESCRIPTION

The purpose of the sensor pod is to house all the sensors that will be used for inspection safely in an underwater environment. The sensor pod consists of an underwater housing, a sensor assembly, an external underwater light and a shore cable providing power and a data connection to a top-side or off-site control station. The control station consists of a power supply and a laptop for data processing and recording.

The underwater housing is shown in Fig. 1 and consists of a polycarbonate tube with two end plates that provide a watertight seal. The sensor assembly (see Fig. 2 for details) is fixed on the rear end plate and a viewport made of optically clear acrylic is epoxied on the opposite end of the polycarbonate tube. Four wing nuts press a circular aluminum plate that prevents axial movement of the polycarbonate tube off of the o-ring seal on the rear end plate. Three through holes for the power and Ethernet cable, underwater light cable, and pressure sensor are located on the rear end plate.

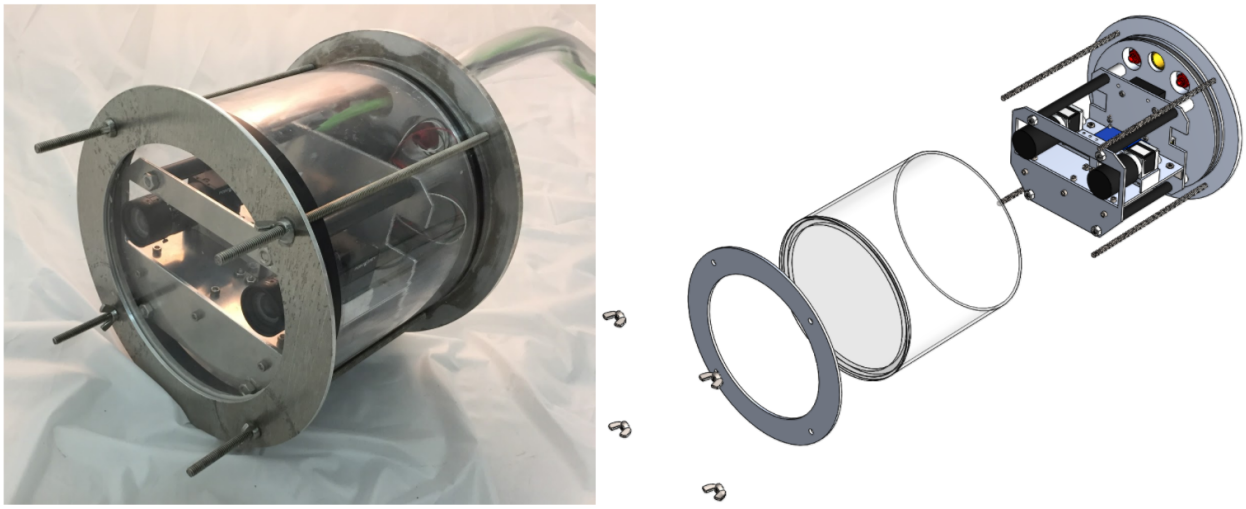

Fig. 1. As-built underwater sensor pod (left) and exploded CAD design of sensor pod (right). Wingnuts press a circular aluminum plate that presses the polycarbonate tube over the o-ring seal on the rear aluminum plate.

To enable accurate 3D modeling, two high resolution (5M pixels, 2448x2048 pixels) 0.169m CMOS color cameras are combined into a stereo configuration. The field of view is 56.9° x 43.9°, a compromise between providing high resolution surface information and wider field of view useful for navigation. For surfaces at 1m distance, a single pixel images a 0.003m by 0.003m area. The cameras are mounted to a solid aluminum block to ensure rigidity, simplifying camera calibration. The baseline is adjustable for experimentation with different interocular distances between 0.06m and 0.12m.

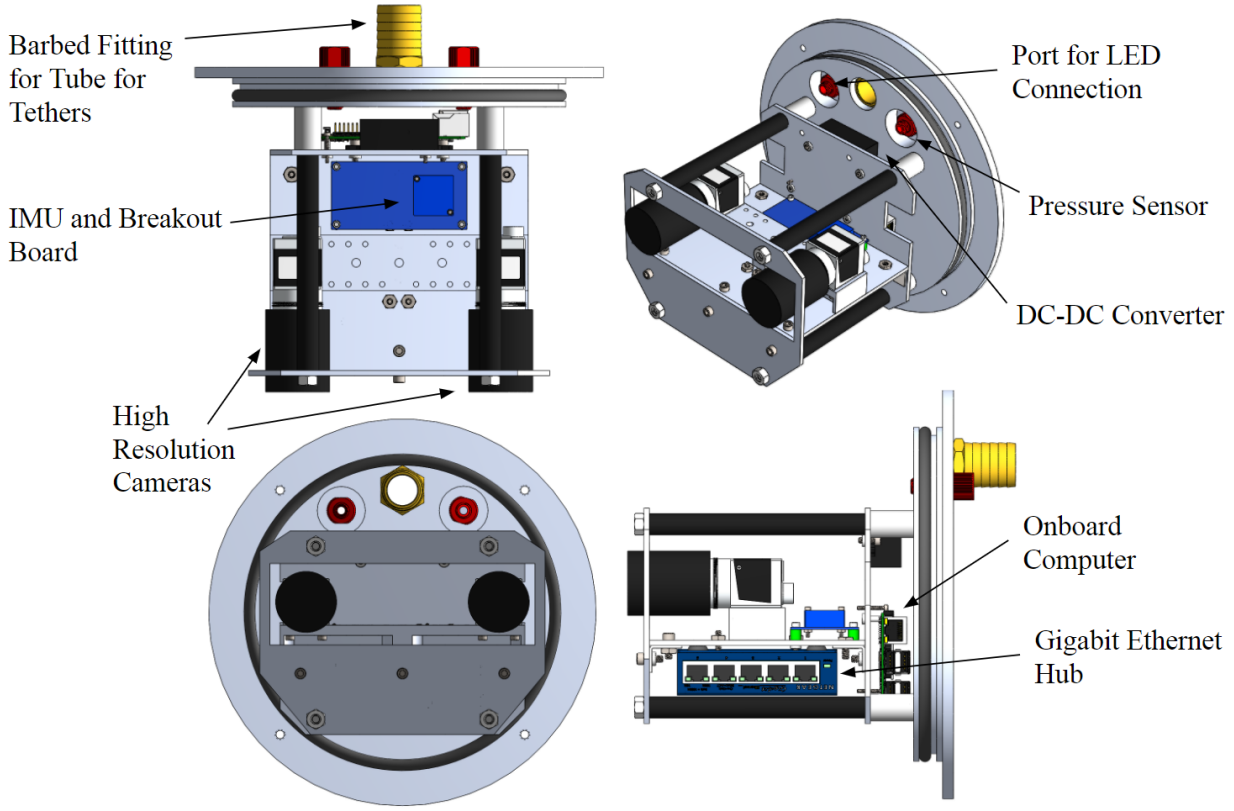

Fig. 2. Sensor assembly inside the underwater housing

The shore cable provides power and a gigabit Ethernet data connection. Therefore, no onboard battery is needed, and data recording and processing is done top-side or off-site. Ethernet allows for a maximum cable length of 100m. While USB3 would provide for a higher bandwidth, cable length is limited to a few meters, which is insufficient even for top-side operation. Gigabit Ethernet is still fast enough to support 10 frames per second from both cameras at their highest resolution (using raw Bayer pattern color coding) in addition to data from the other sensors. A 5-port gigabit Ethernet hub directly connects the two cameras and the onboard computer with the shore cable. The onboard computer interfaces the inertial and pressure sensors to the network connection to the surface.

An underwater LED light provides selective illumination with adjustable brightness. The LED is connected by a cable to the main underwater housing. This allows the light to be placed at a sufficient distance from the cameras to reduce backscatter from any particulates in the water column. The LED is controlled from the onboard computer using a pulse width modulated signal to adjust brightness.

In addition to the stereo cameras, state estimation relies on a MEMS inertial measurement unit (IMU) to provide linear acceleration and rotational velocity in three axes, as well as a pressure sensor to provide a depth measurement. Data from the stereo cameras and IMU are fused together in a tightly coupled smoothing framework, allowing online calibration of sensor biases as well as refinement of projective geometry observed by the cameras. While the pressure sensor provides an absolute measurement on the downward axis, a camera/IMU fusion algorithm estimates the gravity direction, providing an absolute measurement on pitch and roll angles. Therefore, the state estimate can only drift in three degrees of freedom: Forward, sideways, and yaw. Drift can be significantly reduced or even eliminated using modern SLAM techniques, depending on the specific sensor trajectory.

Sensor synchronization is essential toward accurate localization and modeling. In addition to the usual synchronization of the left and right camera of the stereo pair, the IMU measurements also need to be timestamped appropriately. All three sensors have their own internal clocks; however, those will drift with respect to each other over time. Synchronization is provided by the left camera triggering the right camera whenever it takes an image, facilitated by a connection between the two cameras as shown in the wiring diagram in Fig. 3. Upon each triggered image, the right camera generates a reset signal for the IMU, which resets an internal counter that is used to timestamp its measurements. In combination, with some logic, it is possible to estimate the time drift between left camera and IMU, and also be robust to dropped frames from any of the sensors.

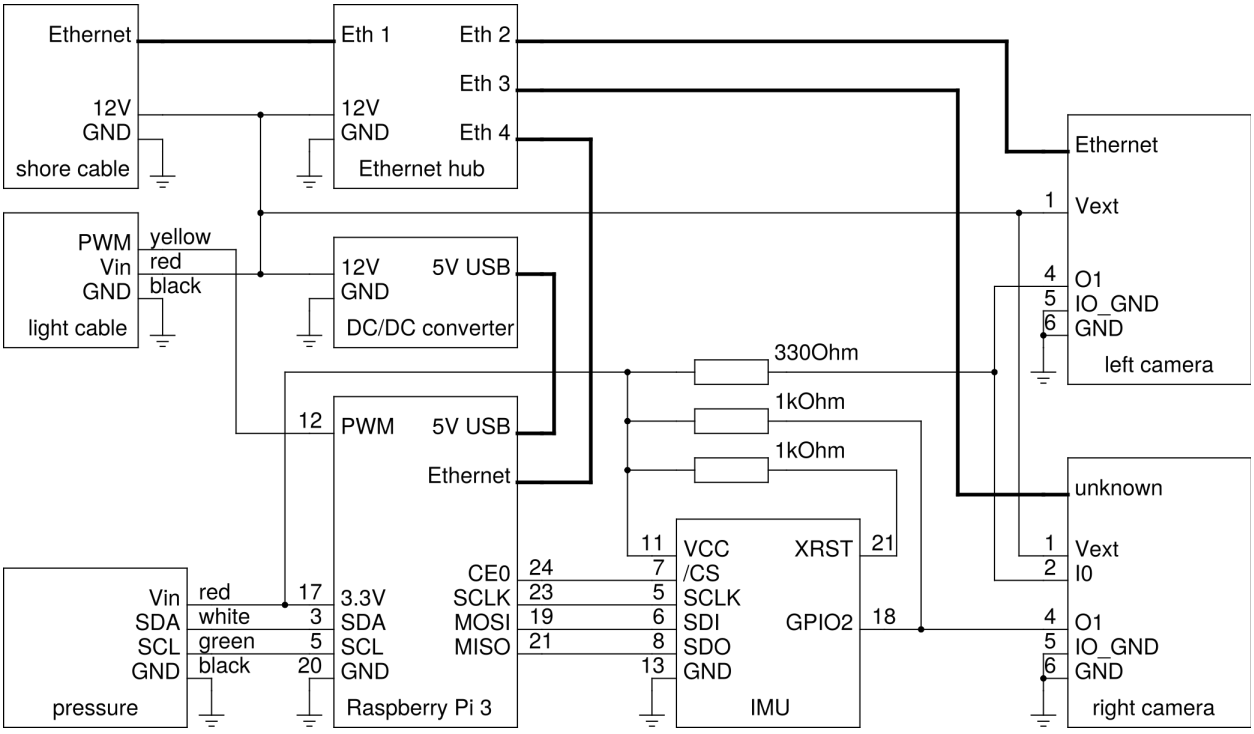

Fig. 3. Sensor assembly wiring diagram. The components include two Pointgrey GigE BFLY-PGE-50S5C-C machine vision cameras, an Epson G364PDC0 inertial measurement unit (IMU), a BlueRobotics Bar30 pressure sensor, and a BlueRobotics Lumen Subsea Light.

ALGORITHMS

Visual odometry (VO) and inertial navigation are two important technologies for mobile robotics and simultaneous localization and mapping (SLAM). The former uses algorithms to calculate frame-to-frame rigid body transformation with respect to cameras, and the latter uses inertial measurement units (IMU) along with other navigation sensors (e.g. GPS) to measure robot trajectories. In the past, VO and inertial navigation have been developed in the computer vision community and robotics community separately. Also, due to hardware and computational constraints, the fusion of camera and inertial sensors was difficult to achieve. Recent advancements in camera hardware and inertial measurement units (IMU) have facilitated research efforts to develop sensor fusion technologies combining the two. Moreover, novel probabilistic frameworks and optimization techniques using factor graphs have permitted SLAM algorithms to perform more efficiently and in real-time [9], [10], [4]. These advancements in both

hardware and in the underlying theory have prompted greater interest in visual-inertial navigation (VIN) by the robotics community.

Vision and inertial information are complementary to each other in SLAM applications. For instance, cameras are exteroceptive sensors which collect dense appearance information of the surroundings. This rich information is useful for 3D reconstruction, damage detection, and obstacle avoidance. Moving cameras enable us to infer the three-dimensional structures of the geometric world, which can further be utilized for calculating long term odometry. On the contrary, IMUs are interoceptive sensors that measure linear acceleration and angular velocity at a high rate. However, the inherent noises and time-varying biases have made IMUs difficult to use standalone and can only be trusted for short-term state estimate. The dual nature of visual and inertial information complements each other, and provide results for superior state estimation than from either one alone. However, fusing the two is nontrivial. In addition to highly nonlinear sensor models, the algorithms need to accurately and consistently account for sensor noises and time-varying biases. Recent works in VIN have addressed these issues and demonstrated impressive results.

The current state-of-the art VIN systems can be broadly categorized into filtering-based and smoothing-based methods [1], [2], [3]. Filtering-based systems such as MSC-KF [1] are derived from extended Kalman filter. Such systems achieve high frame-rate performances because of its simple prediction and correction steps. However, because these methods perform single linearized update for nonlinear systems, they generally suffer from accumulated linearization errors. In smoothing-based methods, sensor measurements and states are formulated using Gaussian assumption in a graphical model called a factor graph [4]. Such formulations store past states and measurements in a unified framework for optimization, solving the maximum a posteriori (MAP) estimation for nonlinear least squares (NLS). An example showing a typical landmark-based SLAM factor graph is shown in Fig. 4. Smoothing-based methods provide better accuracy in nonlinear systems because they iteratively solve for the minimum of an optimization problem. However, more computational resources are required. In fact, extended Kalman filter is a special case of the factor graph smoothing-based method, where it only keeps one past state and performs one update step.

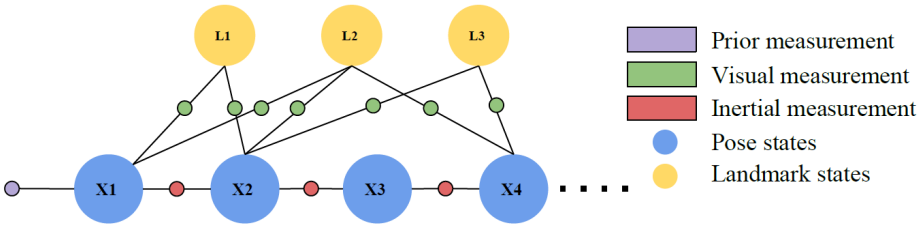

Fig. 4. A factor graph is a bipartite graph. The larger circles representing “variables”, which are connected to each other by smaller circles representing “factors”. The variables in a typical factor graph represent states we want to estimate, while factors are measurements that relate different states. Optimization tools such as iSAM2 utilizes factor graphs to perform inferences and solve for maximum a posteriori (MAP) estimation for a nonlinear least squares (NLS) problem.

In either filtering-based or smoothing-based system, there are two ways to fuse visual information with inertial information. The two different methods are described as loosely-coupled or tightly-coupled. In loosely-coupled VIN systems, visual information is incorporated into the states as odometry information. This is achieved by using VO algorithm either with feature-based methods such as ORB-SLAM [14], [15], Libviso2 [11] or direct methods such as LSD-SLAM [12]. Fig. 5 shows the factor graph formulation

of a typical loosely-coupled VIN, where VO measurement is incorporated as a state-to-state measurement.

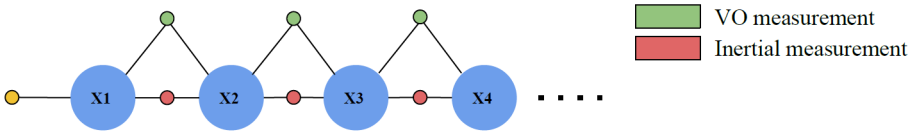

Fig. 5. This figure shows an example factor graph of a typical loosely-coupled VIN. The red factors are IMU measurements and the green factors are VO measurement. There are no landmarks information since visual information is precomputed into state-to-state odometry constraint.

In tightly-coupled systems, visual information is directly incorporated into the factor graph and jointly optimized with inertial information. Fig. 4 shows an example of a typical tightly-coupled system, where the green factors represent visual landmark measurements, and the red factors represent inertial measurements. In addition to raw measurements, tightly-coupled systems incorporate visual landmarks in the states, as represented by yellow circles. The additional visual landmark states provide consistent and more accurate estimation compared to loosely-coupled frameworks.

In general, VIN algorithms consist of a frontend and a backend module as shown in Fig. 6. In a loosely-coupled VIN, the frontend module is the VO algorithm as discussed previously. In a tightly-coupled VIN, the frontend of VIN performs data associations, which receives visual information from the cameras and computes features such as points, lines, or planes [5], [6]. These features are matched between frames and provide input to the backend module. The backend module combines visual features and inertial information to perform joint optimization using optimizers such as iSAM2 [9] and g2o [16].

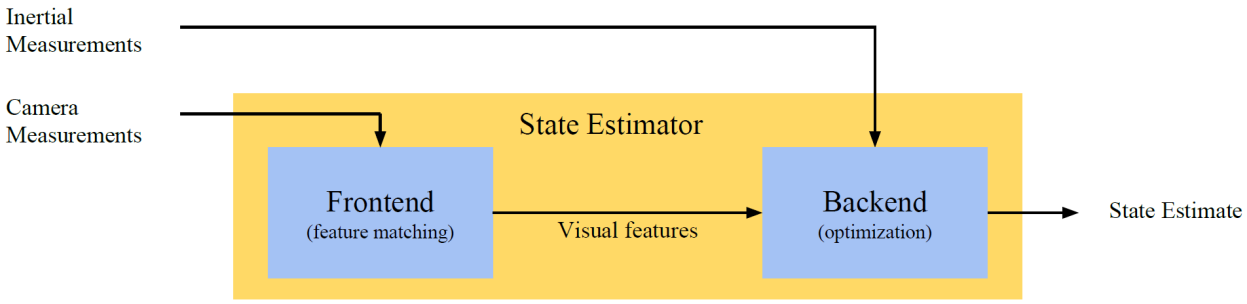

Fig. 6. This graph shows the general framework of a VIN system, which includes a frontend and a backend modules. The frontend module computes visual features and the backend modules combine these features with inertial measurements for joint optimization.

Proposed Approach

In this work, we present a sparse, tightly-coupled fixed-lag smoothing VIN system for stereo cameras and an IMU. Our fixed-lag smoothing based approach maintains a fixed number of past states in the factor graph enabling it to bound the computation complexity while maintaining higher accuracy than traditional filtering-based systems. The following sections describe the frontend and backend modules of our system in detail. As shown earlier in Fig. 6, the frontend module performs data association while the backend module performs the joint optimization.

Frontend Module

The stereo camera system provides images at a rate of 10Hz. Upon receiving raw images, stereo rectification is performed to obtain undistorted stereo images. We then adopt a sparse feature-based method using Shi-Tomasi corner detector and Kanade-Lucas-Tomasi (KLT) feature-tracking algorithm [7] to track features in temporal images as well as stereo images. To ensure sufficient space between features, a separation distance of 20 pixels is used to evenly spread features across the entire image. Each feature is associated with a unique identification number, pixel location, and the frame in which it is detected. Mismatched features are rejected using the Random Sample Consensus (RANSAC) in the final step. Our frontend pipeline is lightweight and requires minimal computational resources. Using 300-500 features, our frontend module can process 80-100 frames per second, which surpasses the frame rate of standard cameras. Fig. 7 shows the frontend block diagram and an example output of the frontend module.

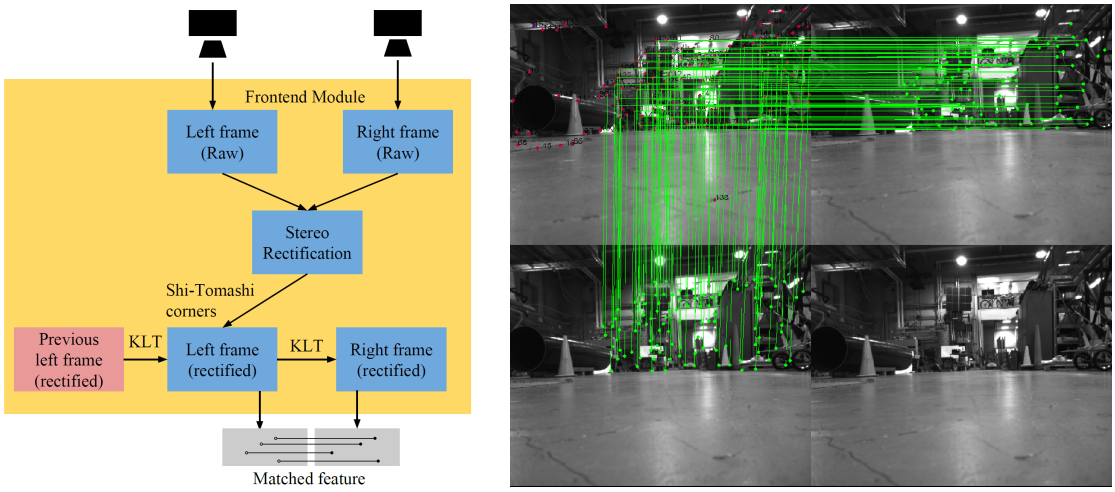

Fig. 7. Left) The block diagram shows the pipeline of our frontend module. After receiving raw stereo images, first stereo rectification is performed. We then use Shi-Tomashi and KLT to track temporal as well as stereo features. Right) The output of our frontend modules includes temporal feature matching using Kanade-Lucas-Tomasi (KLT) tracking and stereo feature matching. The top-left image shows left camera frame, and the other three frames are right camera frames. The top horizontal green lines show stereo matching with epipolar constraints, while the vertical green lines show disparity between each feature match.

Backend Module

The backend module performs nonlinear optimization for a tightly-coupled, fixed-lag smoothing-based estimation objective. We formulate this optimization as a factor graph as shown in Fig. 8.

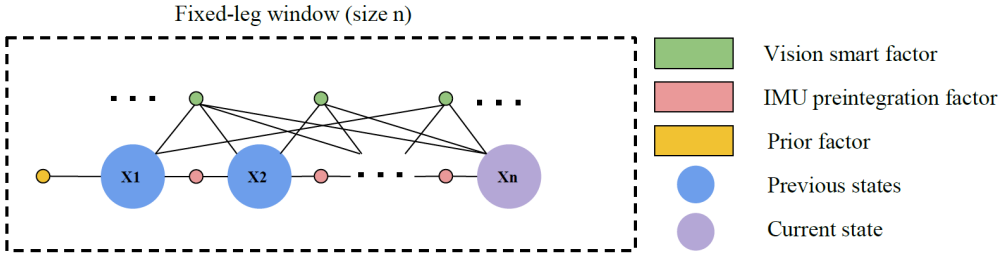

Fig. 8. Our factor graph formulation consists of three types of factors: prior factor, smart factors, and IMU preintegration factors. The only states we are estimating are poses, velocities, and biases.

Each state x_i is modeled as a variable node in the graph while each measurement residual is modeled as a factor node. Since each state x_i in the factor graph corresponds to a camera frame I_i, our backend module needs to be able to perform optimization at the frame rate of the camera. Consider a fixed-lag window of n states, the full state vector χ is defined as collections of camera states n_i as:

$$\chi = [x_1, \dots, x_n]$$
$$x_i = [p_i, v_i, b_i]$$

Within each state x_i, p_i is the 3D pose; v_i is the linear velocity; and b_i is the acceleration and angular velocity biases. Each visual feature also known as a landmark is modeled as a measurement factor called smart factor [13]. A smart factor internally estimates its landmark position but only constraints poses where the landmark is visible. The use of smart factors as opposed to modeling each landmark in the state vector significantly reduces the dimension of the full state vector, and therefore improves efficiency. Between consecutive states, because an IMU is operating at a much higher rate, there are numerous IMU measurements. Instead of modeling each inertial measurement as one factor, we employed the preintegration theory by Forster et al. [2]. It accumulates inertial measurements into a single relative pose constraints called an IMU preintegration factor, which also reduces computational complexity in optimization. Interested readers may refer to [2] for complete mathematical derivation of the preintegration theory.

At every camera frame, we jointly optimize visual and inertial information within a single optimization framework. We assume measurement and motions models with added Gaussian noise. The resulting optimization is a maximum a posteriori (MAP) estimation of the posterior probability:

$$\begin{aligned} p(\chi_k|Z_k) &\propto p(\chi_0)p(Z_k|\chi_k) \\ &= p(\chi_0) \prod_{(i,j)\in I_k} p(z_{ij}^{imu}|x_i, x_j) \prod_{i\in I_k} \prod_{l\in C_i} p(z_{il}^{cam}|x_i) \end{aligned} \quad \text{(Eq. 1)}$$

which can be further reduced to nonlinear least squares minimization by taking the negative logarithm:

$$\begin{aligned} \chi^* &= \underset{\chi_k}{arg\min} -\log\big(p(\chi_k|Z_k)\big) \\ &= \underset{\chi_k}{arg\min} \ \|r_0\|^2_{\Sigma_0} + \sum_{(i,j)\in I_k} \left\|r_{i_{ij}}\right\|^2_{\Sigma_{ij}} + \sum_{i\in I_k} \sum_{l\in C_i} \|r_{C_{il}}\|^2_{\Sigma_c} \end{aligned} \quad \text{(Eq. 2)}$$

where, r represents the residual errors, and Σ the corresponding covariance matrices. The MAP estimate in (Eq. 2) consists of three terms. The first term corresponds to the prior, the second to the inertial measurements, and the third to the visual measurements.

EXPERIMENTAL EVALUATION

Implementation Details

We implemented our VIN using OpenCV library for the frontend module and GTSAM library [4] for the backend module. The computation of the MAP estimate in (Eq. 2) is based on Levenberg-Marquardt (LM) algorithm. Because our VIN uses fixed-lag smoothing, full optimization using LM still promises real-time operation. Our factor graph formulation is also suitable for using state-of-the-art incremental smoothing approach, iSAM2 [9], which exploits matrix sparsity and the fact that new measurements only affect local variables. For control and mapping applications where high- rate state estimates are required, our VIN is capable of performing state estimate at IMU rates even if optimization occurs at camera rates. Between two states x_i, x_j, we apply inertial propagation since IMU measurement is reliable in the short-term. This is assuming the optimizer has correctly estimated the biases after optimization.

Results and Analysis

To validate our VIN, we tested on our custom datasets and the European Robotics Challenge (EuRoC) dataset [8]. Our own dataset has camera images recorded at 10Hz with a resolution of 640x512, and the IMU recorded at 250Hz. Our custom dataset provides basic validation by moving the sensing device in a smooth manner in a well-textured environment. We also use a public dataset, the EuRoC dataset, for benchmarking our VIN system. It consists of a series of trajectories ranging from easy to difficult scenarios such as poor lighting conditions or fast motions. Fig. 9 shows an example from the EuRoC dataset (Machine Hall 01 dataset) with relevant statistics in TABLE I. This particular dataset has a total length of 80.6 meters and total time of 183 seconds. Its environment is feature rich, and thus provides sufficient texture for the stereo vision sensor.

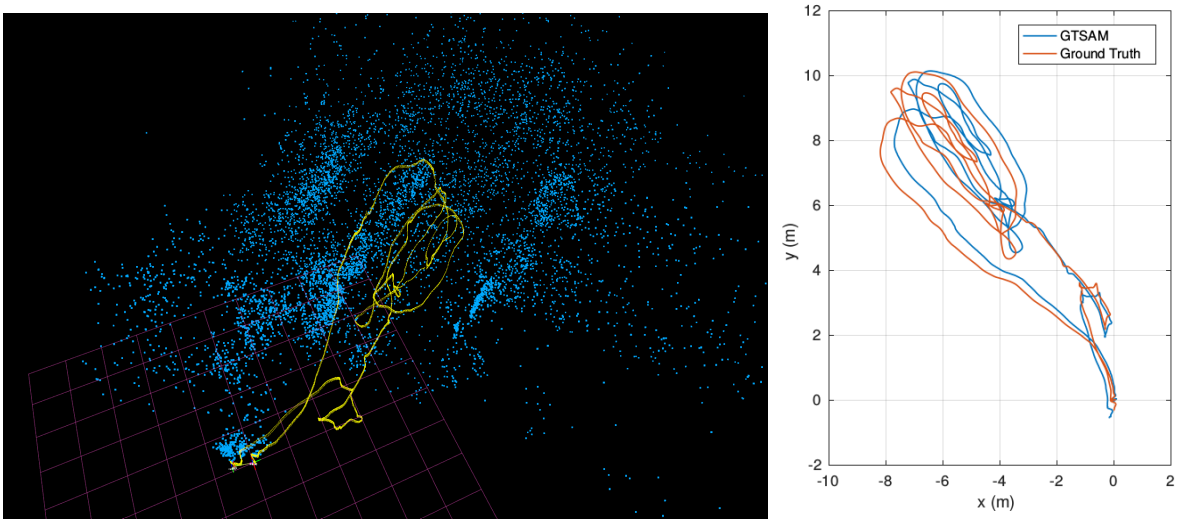

Fig. 9. (Left) Result of our VIN on EuRoC dataset "Machine Hall 01". The yellow line represents the trajectory of our VIN. The blue dots are point cloud representing the feature points used by the algorithm. (Right) Comparison between our result vs. ground truth. There is a small rotational difference so the two trajectories do not perfectly align.

TABLE I. Machine Hall 01 Dataset Statistics

		RMSE	Mean	Median	Std	Min	Max
Absolute Trajectory Error (ATE)	Positional (m)	0.130672	0.122982	0.117527	0.044164	0.025472	0.237720
Relative Pose Error (RPE)	Translational (m)	0.037452	0.031576	0.026708	0.020140	0.001040	0.164382
	Rotational (°)	0.651658	0.505816	0.406956	0.410863	0.016415	3.729380

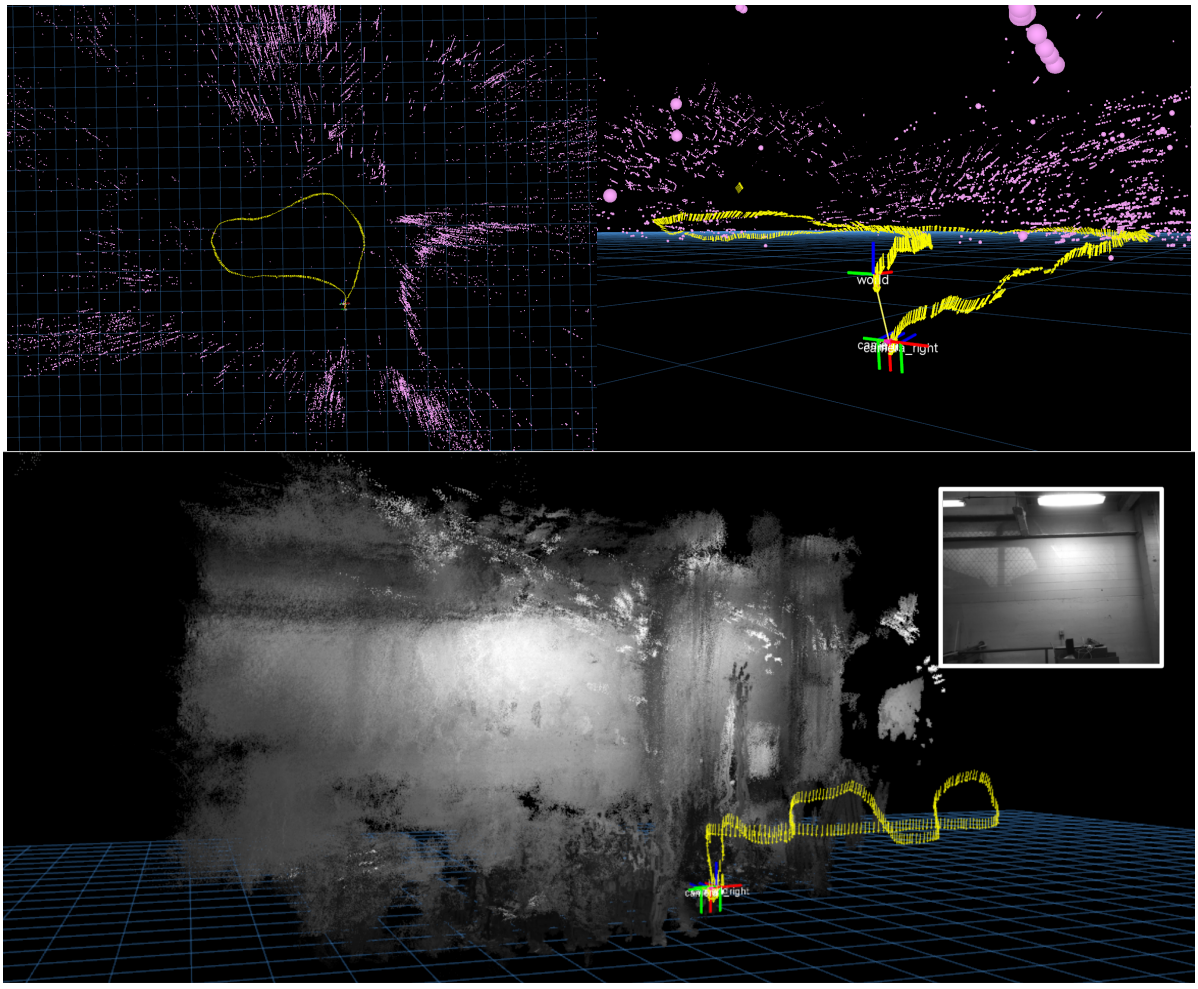

Fig. 10. (Top-left) Result of our custom dataset, where we walked in a 25m loop while constantly rotating the sensor device. The yellow line represents the trajectory of our VIN. The pink dots are point cloud representing the feature points used by the algorithm. (Top-right) The result of final translational error of 0.43m. This results in a 1.72% error. (Bottom) Combining mapping of a wall with stereo disparity and odometry.

Our dataset mainly includes indoor data. To collect the data, we hand-carried the sensing device indoor in a well-textured environment from a known start and end location. Fig. 10 is one example where we walked in a loop of 25m, while constantly rotated the sensing device to test its robustness against rotations. The final translational error is 0.43m, which results in 1.72% error. Our VIN is also ready to be incorporated into other mapping algorithms. For example, from the stereo rectification algorithm from the frontend module, we can obtain the disparity map from the stereo sensing device. The bottom graph of Fig. 10 shows one example where we perform 3D mapping using the disparity map with our VIN.

CONCLUSION

We have presented a system for underwater inspection of spent nuclear fuel storage basins. We have described our custom sensor pod design that combines multiple complementary sensors and provides accurate time synchronization. We have further described our algorithmic approach applying state-of-the-art robotic technologies to provide localized images and 3D models. We have evaluated our approach in air, showing localization with low drift.

Next steps include testing with the sensor pod in an underwater environment. At first, the sensor pod will be moved manually on a stick to collect data. Future development will explore options for a free swimming, neutrally buoyant robot platform to extend the reach of the sensing areas not accessible by manual operation with a stick. This could be achieved through treads or wheels that move the robot on the floor or on the side of the basin as a wall crawler. Thrusters provide another option for mobility that could be used standalone or in tandem with the previous methods described.

Furthermore, we will address the mapping aspects in a simultaneous localization and mapping (SLAM) framework. This will address two goals: creating high resolution 3D map for inspection and using the maps on the fly for re-localization, further eliminating drift in the localization estimate.

Finally, a third camera will be integrated into the system that faces the surface of the water, with two LEDs placed close to the water surface. A wide-angle lens allows the camera to see the LEDs even at fairly shallow depth. Detection in the camera provides two angles to each LED, and when combined with inertial and pressure provide a global estimate for all six degrees of freedom. The two LEDs will be rigidly attached to a bar with a fixed baseline. They are controlled from the top-side laptop to provide robust detection as well as identification of each LED, even through several meters of water and interference from ceiling lights.

REFERENCES

[1] A. Mourikis and S. Roumeliotis, “A multi-state constraint Kalman filter for vision-aided inertial navigation,” in Proc. IEEE Int. Conf. Robotics and Automation, 2007, pp. 3565 – 3572.

[2] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, “IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation,” in IEEE Robotics: Science and Systems, 2015.

[3] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, K. Konolige, and R. Siegwart, “Keyframe-based visual-inertial SLAM using nonlinear optimization,” in Robotics Science and Systems (RSS), Berlin, Germany, 2013

[4] F. Dellaert. “Factor graphs and GTSAM: A hands-on introduction.” Technical Report GT-RIM-CP&R-2012-002, Georgia Institute of Technology, 2012.

[5] T. Liu and S. Shen, “Spline-based initialization of monocular visual-inertial state estimators at high altitude,” IEEE Robotics and Automation Letters, 2017

[6] M. Hsiao, E. Westman, G. Zhang, and M. Kaess, "Keyframe-based dense planar SLAM," in: Proc. International Conference on Robotics and Automation (ICRA), IEEE, 2017.

[7] S. Baker and I. Matthews, "Lucas-Kanade 20 years on: A unifying framework: part 1," Int'l J. Computer Vision, vol. 56, no. 3, pp. 221-255, 2004

[8] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," Int. J. Robot. Res., vol. 35, no. 10, pp. 1157–1163, Jan. 2016.

[9] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert (2012) "iSAM2: Incremental smoothing and mapping using the Bayes tree." The International Journal of Robotics Research 31(2): 216–235.

[10] M. Kaess, A. Ranganathan, and F. Dellaert, (2008). "iSAM: Incremental smoothing and mapping." IEEE Trans. Robotics, 24(6):1365–1378

[11] B. Kitt, A. Geiger, and H. Lategahn. "Visual odometry based on stereo image sequences with RANSAC-based outlier rejection scheme." In Intelligent Vehicles Symposium, June 2010.

[12] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in Proc. Eur. Conf. Comput. Vision, Zurich, Switzerland, Sep. 2014, pp. 834–849.

[13] L. Carlone, Z. Kira, C. Beall, V. Indelman, and F. Dellaert, "Eliminating conditionally independent sets in factor graphs: A unifying perspective based on smart factors," in Proc. IEEE Int. Conf. Robot. Autom., 2014.

[14] R. Mur-Artal, J. Montiel, and J. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," IEEE Transactions on Robotics, vol. 31, no. 5, pp. 1147–1163, 2015.

[15] R. Mur-Artal and J. Tardos, "ORB-SLAM2: An open-source SLAM system for monocular, stereo and RGB-D cameras," CoRR, vol. abs/1610.06475, 2016. [Online]. Available: http://arxiv.org/abs/1610.06475

[16] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. "g2o: A general framework for graph optimization." In Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA), 2011.