2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

A Synchronized Visual-Inertial Sensor System with FPGA Pre-Processing for Accurate Real-Time SLAM

Janosch Nikolic, Joern Rehder, Michael Burri, Pascal Gohl,
Stefan Leutenegger, Paul T. Furgale and Roland Siegwart[1]

***Abstract*— Robust, accurate pose estimation and mapping at real-time in six dimensions is a primary need of mobile robots, in particular flying Micro Aerial Vehicles (MAVs), which still perform their impressive maneuvers mostly in controlled environments. This work presents a visual-inertial sensor unit aimed at effortless deployment on robots in order to equip them with robust real-time Simultaneous Localization and Mapping (SLAM) capabilities, and to facilitate research on this important topic at a low entry barrier.**

Up to four cameras are interfaced through a modern ARM-FPGA system, along with an Inertial Measurement Unit (IMU) providing high-quality rate gyro and accelerometer measurements, calibrated and hardware-synchronized with the images. This facilitates a tight fusion of visual and inertial cues that leads to a level of robustness and accuracy which is difficult to achieve with purely visual SLAM systems. In addition to raw data, the sensor head provides FPGA-pre-processed data such as visual keypoints, reducing the computational complexity of SLAM algorithms significantly and enabling employment on resource-constrained platforms.

Sensor selection, hardware and firmware design, as well as intrinsic and extrinsic calibration are addressed in this work. Results from a tightly coupled reference visual-inertial motion estimation framework demonstrate the capabilities of the presented system.

***Index Terms*— Visual-Inertial Motion Estimation, SLAM, Camera, IMU, FPGA, Calibration, Sensor Fusion.**

I. INTRODUCTION

Many mobile robots require on-board localization and mapping capabilities in order to operate truly autonomously. Control, path planning, and decision making rely on a timely and accurate map of the robots surroundings and on an estimate of the state of the system within this map.

Accordingly, Simultaneous Localization and Mapping (SLAM) has been an active topic of research for decades [1]. Tremendous advances led to successful employments of SLAM systems on all sorts of platforms operating in diverse environments. Different interoceptive and exteroceptive sensors such as 2D and 3D laser scanners, wheel odometry, cameras, inertial sensors, ultrasonic range finders, and radar, amongst others, provide the necessary data.

Yet it is often a challenge to equip a platform with a reliable and accurate real-time SLAM or state estimation system that fulfills payload, power, and cost constraints. A "plug-and-play" SLAM solution that achieves all requirements and runs robustly under the given conditions is seldom readily available, and thus significant engineering efforts often have to be undertaken.

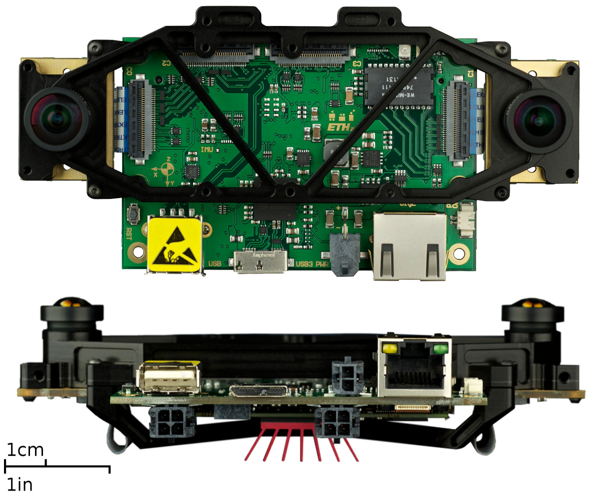

Fig. 1: The *SLAM Sensor* unit in a fronto-parallel "stereo" configuration (front- and side-view). The sensor interfaces up to four cameras and incorporates a time-synchronized and calibrated inertial measurement system. Access to high quality raw- and pre-processed data is provided through simple interfaces.

Visual SLAM systems that rely on cameras have received particular attention from the robotics and computer vision communities. A vast amount of data from low-cost, lightweight cameras enables incredibly powerful SLAM or structure-from-motion (SfM) systems that perform accurate, large-scale localization and (dense) mapping in real-time [2], [3]. However, SLAM algorithms that rely only on visual cues are often difficult to employ in practice. Dynamic motion, a lack of visible texture, and the need for precise structure and motion estimates under such conditions often renders purely visual SLAM inapplicable.

Augmenting visual SLAM systems with inertial sensors tackles exactly these issues. MEMS Inertial Measurement Units (IMUs) provide valuable measurements of angular velocity and linear acceleration. In *tight combination* with visual cues, this can lead to more robust and accurate motion estimation systems that are able to operate in *less controlled*, *sparsely textured*, and *poorly illuminated* scenes while undergoing dynamic motion. However, this requires all sensors to be well calibrated, rigidly connected, and precisely time-synchronized.

This work makes a step towards a *general-purpose SLAM*

[1] Janosch Nikolic and Joern Rehder contributed equally to this work. All authors are with the ETH, the Swiss Federal Institute of Technology Zurich, Autonomous Systems Lab (www.asl.ethz.ch), Tannenstrasse 3, CLA, CH-8092 Zurich, Switzerland.

system by providing these capabilities. The sensor head evolved through the development of several prototypes and was tested in many applications, for instance in a coal fired power plant [4] or on a car [5]. Fig. 1 shows our final hardware iteration.

The remainder of this article is organized as follows: in Section III, we outline the design concept, FPGA-pre-processing (see Section III-C), and the calibration of such a visual-inertial sensor unit (see Section IV). We provide an overview of our reference tightly coupled visual-inertial motion estimation framework in Section V, which we use in Section VI to illustrate the capabilities of the sensor system.

II. RELATED WORK

There exist different FPGA vision systems particularly geared to robotics. The GIMME platform [6] is similar in scope to this hardware in that it computes visual interest points on an FPGA and transmits those to a host system in order to bring visual pose estimation to platforms with computational and power constraints. However, it is a purely visual sensor setup and hence does not require elaborated synchronization or calibration between different types of sensors.

Another system that employs inertial sensors has been developed by the DLR [7]. In this system, a general purpose computer and an FPGA are closely interleaved in order to enable ego-motion estimation and depth computation on a handheld device. In contrast to our setup, cameras and inertial sensors are not as tightly integrated into the system, and images appear to be timestamped at the start of sensor exposure, resulting in a varying, exposure dependent offset to IMU measurements. Furthermore, its weight might prohibit application on very payload-constrained platforms.

As heterogeneous sensor systems for motion estimation and localization become increasingly popular, spatial calibration has attracted some attention and resulted in a variety of frameworks [8]–[10]. More recently, the importance of accurate synchronization of the sensors became apparent and was addressed in [8], [11], [12]. While this work makes use of the calibration presented in [12] to determine the transformation between cameras and IMU and to determine fixed delays present when polling inertial data, its approach to the problem is exactly antithetic: rather than connecting a set of stand-alone sensors to a general purpose computer and calibrating for potentially time-variant time-offsets afterwards, we pursued a tight integration of all hardware components with a central unit capable of concurrent triggering and polling of all sensors.

III. THE VISUAL-INERTIAL SLAM SENSOR

This section outlines important design concepts and "lessons learned" throughout the development of three successive prototypes that led to the sensor presented here.

Subsection III-A provides a conceptual overview of the sensor. Subsection III-B describes a synchronization method that guarantees ideal alignment of all sensors in time. Subsection III-C describes the FPGA implementation of image processing operations such as keypoint detection to reduce CPU-load of successive SLAM software.

A. Sensor Design Concept

At the core of the SLAM sensor, we employ a modern XILINX Zynq System-on-Chip (SoC), a device that combines FPGA resources with a dual ARM Cortex-A9 on a *single chip*. Hardware programmability allows a direct, lowest-level interface to the CMOS imagers and inertial sensors, enabling precise synchronization and a reliable data acquisition process.

At the same time, the chip offers a powerful, industry standard CPU running Linux. This facilitates simple development and efficient execution of processes that are time-consuming to implement on an FPGA (e.g. host-communication or even a simple SLAM framework). In contrast to previous prototypes which featured a XILINX Spartan-6 FPGA - Intel ATOM combination, this also offers a better integration and a higher bandwidth between logic and CPU. Fig. 2 gives an overview of the hardware architecture, and Table I summarizes the technical specifications of the sensor unit.

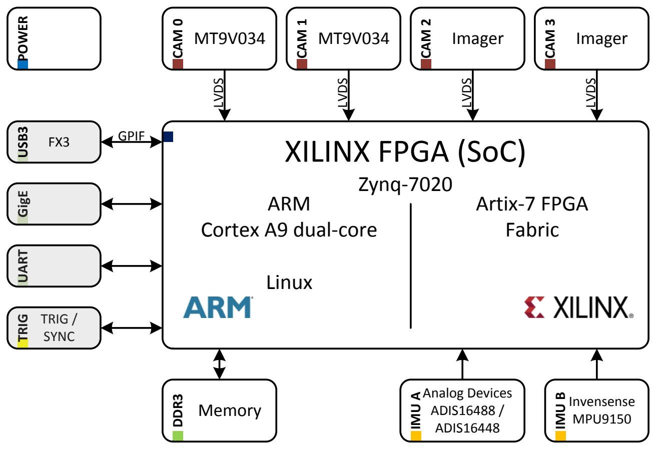

Fig. 2: Block-diagram of the SLAM sensor hardware architecture. Camera chips and inertial sensors interface the ARM-FPGA system-on-chip directly. Standard interfaces provide fast access to the data provided by the module.

1) Visual Subsystem: The SLAM sensor offers *four* camera extension ports. The ability to rely on several cameras is crucial for many real-world applications. Even when wide Field-of-View (FoV) optics are used, a single camera may still point into a direction where keypoint tracking is difficult (lack of texture, temporary obstruction of the FoV, bad illumination). With the option to use four cameras simultaneously, configurations such as the combination of a "fronto-parallel" stereo pair and two fish-eye modules are quickly realized.

In the current configuration, camera chips were selected according to their *low-light sensitivity* and *global shutter* functionality. Aptina's MT9V034 CMOS sensors offer good performance and a direct interface to the FPGA through

Basic Characteristics	Value/Characteristic	Unit
Mass (for diff. configurations)		
1 cam+MPU	60	g
2 cams+mount+ADIS16448	130	g
4 cams+mount+ADIS16488	185	g
Embedded Processing	XILINX Zynq 7020	
Processor	2xARM Cortex A9	
FPGA	ARTIX-7	
Interfaces	GigE, USB2/3	
Camera System		
Sensors	Aptina MT9V034	
Shutter type	global	
Opt. resolution	752×480	pixel
Max. frame rate	60	fps
Inertial System (ADIS16488)		
Rate Gyroscope		
Measurement Range	±1000	°/s
Noise Density	0.007	°/s $Hz^{-1/2}$
Accelerometer		
Measurement Range	±177	ms^{-2}
Noise Density	$0.66 \cdot 10^{-3}$	ms^{-2} $Hz^{-1/2}$
Max. sampling rate	2.4	kHz

TABLE I: Overview of the SLAM sensors technical specifications. High quality sensors that perform well in low-light scenarios and when undergoing dynamic motion are integrated in the sensor unit. The module's relatively low weight facilitates employment on payload-constrained platforms.

LVDS ports. By default, Lensagon lenses of the type BM2820 (122° diagonal FoV) or BM2420 (132° diagonal FoV) are used.

In addition, a FLIR Tau 2 *thermal imager* can be connected, which then occupies one of the camera ports. Similar to the camera modules, it directly interfaces with the Zynq providing time-synchronized digital (14 bit dynamic range) thermal images to the host.

2) Inertial Subsystem: The current prototype allows two options with regard to the IMU subsystem. By default, each camera module is fitted with a low-cost MEMS IMU offering a triple axis gyroscope, accelerometer, and magnetometer in a single package. The MPU-9150 was selected due to its high range in both angular rates and acceleration. Chip internal filtering and processing are switched off, and only raw data is used.

In addition, a factory-calibrated MEMS IMU system from the ADIS family of Analog Devices can be connected. The ADIS16448 and ADIS16488 are equipped with higher quality gyroscopes and accelerometers, and they are factory-calibrated over a large scale and temperature range. Depending on the application, one can trade-off sensor weight versus accuracy of the inertial subsystem.

B. Sensor Synchronization and Data Acquisition

We configure the image sensors for external triggering. At the same time, the inertial sensors are polled for data acquisition. As stated earlier, accurate synchronization of different sensors was the driving motivation for a tight integration in hardware. It is an established fact in photogrammetry, that images should be timestamped by their mid-exposure time, and in previous work [12], it could be shown that neglecting image exposure time in timestamping data has an observable effect, which suggests that it could adversely affect image-based state estimation. We made the design choice to not correct for the exposure time in timestamping images, but to account for the exposure time when triggering the sensors. This way, the middle of the exposure times will still be equally spaced despite varying lighting conditions, which exhibits certain advantages when representing states in a time-discrete manner. Fig. 3 illustrates the synchronization scheme in comparison with periodically triggering, where varying lighting conditions result in exposure midpoints that are not equally spaced.

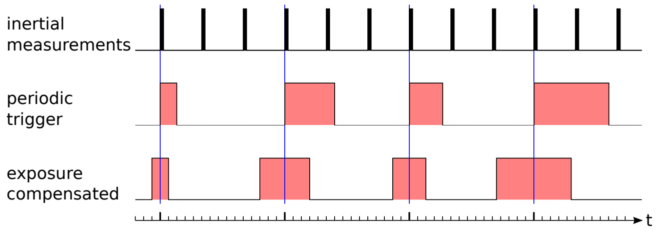

Fig. 3: This timing diagram shows strictly periodic polling of an IMU as well as two schemes of camera synchronization, where high levels mark exposure times. Triggering the camera at the instance an inertial measurement is retrieved is a common approach to synchronization. However, the exposure is asymmetrical with respect to the inertial measurement. By taking varying exposure into account and shifting each triggering instance accordingly, significantly improved synchronization can be achieved, as demonstrated in Fig. 6.

Note that also the inertial measurements may exhibit a delay. This delay is in general fixed and can be a combination of communication, filter and logic delays. Section IV will detail on estimating this delay, which is compensated for in the same way the exposure delay is addressed, by moving the moment when a polling request is initiated with respect to the point in time when the measurement is timestamped. As part of the results section, Fig. 6 reproduces an experiment from [12]. The results demonstrate that the delays can be accounted for in the sensor data acquisition, thereby improving the synchronization between sensors significantly.

C. FPGA Accelerated Image Processing

As depicted in Fig. 7, the detection of interest points consumes a significant share of the processing time in the state estimation pipeline. At the same time, many interest point detectors operate on a rather confined neighborhood of pixels and can be implemented exclusively using fixed-point arithmetic, which renders them well suited for an implementation as dedicated logic blocks inside an FPGA. For this project, a fixed-point version of the Harris corner detector [13] as well as the FAST corner detector [14] have been implemented. While the resources of the FPGA used in the setup are not sufficient to integrate them both at the same time, it is possible to load the FPGA with different configurations depending on the requirements of the experiment. Note that the quantities reported in Table II have been acquired for an earlier prototype based on the Xilinx XC6SLX45T.

Harris Corner Detection: The Harris corner detector is based on an approximation of the auto-correlation function for small image patches. With I_x denoting the derivative in x-direction of the image intensity at pixel $x+u, y+v$, and $w(u,v)$ denoting a weighted averaging function, the approximated local auto-correlation is calculated as [13]

$$\mathbf{A}(x,y)=\sum_u\sum_v w(u,v)\left[\begin{array}{cc} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{array}\right]. \quad (1)$$

With $\mathbf{A}$ and a weighting factor k, the corner response function r is calculated as

$$r = |\mathbf{A}| - k\ \mathrm{tr}(\mathbf{A})^2. \quad (2)$$

Larger positive values of this function correspond to corner regions, while negative results indicate edges. Flat regions trigger a small response. Examining this function reveals pixel differencing operations, cascaded multiplications as well as local averaging. Fig. 4 depicts the FPGA implementation of the corner score function. Derivatives of image intensities are computed by means of Sobel filters, while local averaging is performed by convolution with a Gaussian kernel. As in [6], weighting the Trace of the matrix in the cost function has been realized by a bit shift operation. Individual blocks like Sobel and Gaussian filters as well as the multipliers in the pipeline operate at higher frequency than the pipeline itself—25 MHz and 125 Mhz respectively—allowing for the re-utilization of resources. Furthermore, by making use of the separability properties of Sobel and Gaussian filters, resource utilization can further be reduced. The resulting resource utilization is shown in Table II. The maximum clock rate is limited and thus imposes upper bounds on the degree to which resources can be shared. However, the pixel rate of the sensors used in this sensor setup allows for a excessive re-utilization of resources, resulting in a core that can be conveniently duplicated for four cameras without exceeding the area of the FPGA.

FAST Corner Detection: The FAST corner detector is a heuristically motivated approach to interest point detection, which compares intensities of image points on a circle around the point in question. It identifies a pixel as an interest point based on the number of pixels in a segment that is either coherently lighter or darker than the central element. In [14], different scores for nonmaximum suppression are proposed. Taking the mere number of coherent intensity comparisons can be efficiently implemented, but results in a rather coarsely quantized score. On the other hand, considering the sum of absolute differences (SAD) of this segment with the center pixel yields finer granularity in the score at the expense of occupying a larger area on-chip. In this project, both scores have been implemented with the resource utilization displayed in Table II. Fig. 5 illustrates a detail of the implementation as a block diagram, which depicts the path testing for lighter pixels, which is duplicated for the test for darker pixels. The central and surrounding pixel, grouped in sets of four consecutive elements, feed into the block. The design heavily employs identical blocks, which are only shown in a number sufficient to convey the underlying interconnection principles. As for the Harris implementation, individual components of the detector are clocked at a higher rate than the overall pipeline, resulting in a reduction in resource utilization. To this end, the comparison with the central pixel is executed in four clock cycles, decreasing the number of comparators that operate on image data. Counting of coherent segment lengths is done for each potential starting point of the segment in parallel. The appropriate signal connecting the counting units with the registers holding the intensity comparisons are represented by a routing network block in the schematic. Per clock cycle, each segment length counter evaluates four comparisons. To this end, the counter block depicted in Fig. 5 determines the position of the first zero in the 4 bit segment, and accumulates these. Once the coherency of a segment is interrupted, further accumulations are blocked. In order to determine the maximum coherent segment length from the parallel counter units, a recursive comparator structure has been implemented. The comparison for darker is implemented accordingly and results from the two paths which are fused using an additional comparator stage. The figure does not depict the extraction of the central pixel and the surrounding circle that precedes the block shown, as well as the non-maximum suppression succeeding the block. Note that Fig. 5 depicts the case where the mere segment length is employed.

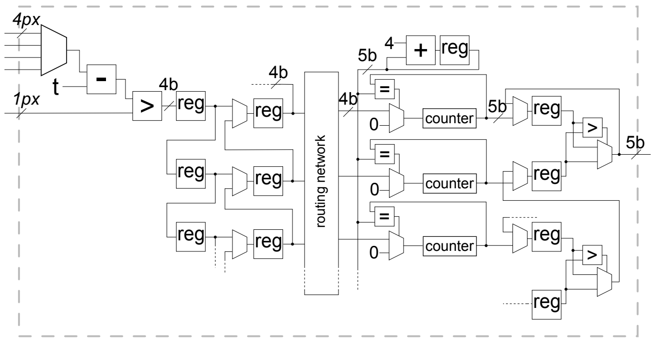

Fig. 5: Logic diagram of a detail of the fast implementation. By reusing blocks, the area footprint of the core can be reduced significantly.

	RAMB16B	DSP48A	Slices
Harris	17 (14%)	8 (13%)	774 (11%)
FAST	5 (4%)	0 (0%)	1,124 (16%)
FAST+SAD	5 (4%)	0 (0%)	1,913 (28%)

TABLE II: Resource utilization of the implemented interest point detectors for a WVGA image on a Xilinx Spartan 6 architecture. The number in brackets indicates the device utilization for a Xilinx XC6SLX45T.

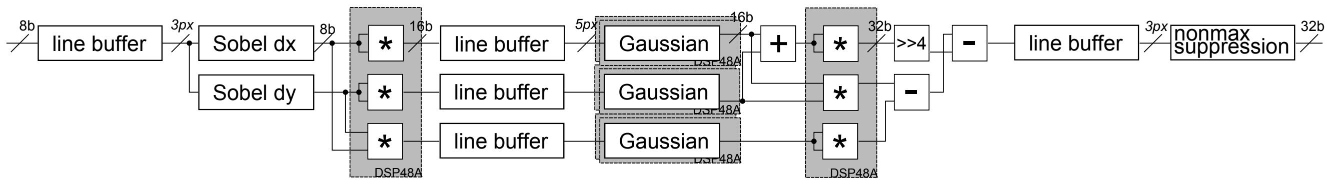

Fig. 4: Block diagram illustrating the implementation of the Harris corner detector. For improved readability, only the width of the topmost path is shown, which also applies to any other path in the same column. Note that bit widths are shown in oblique font, while line buffer widths in terms of numbers of pixels of the respective input bit width are displayed in italics. Unless marked otherwise, bit widths propagate through blocks. For operations that potentially lead to an overflow, a saturation operation is performed. Each gray box illustrates the location of a single DSP slice within the processing pipeline.

IV. CALIBRATION

In order to achieve accurate motion estimates, the sensor setup needs to be calibrated. As a factory calibrated IMU is employed in the setup, the remaining quantities that need to be estimated are

- the camera intrinsics,
- the extrinsics of the stereo setup,
- the transformation between the cameras and the IMU,
- and the fixed time delay between camera and IMU measurements.

The camera intrinsics and stereo extrinsics are determined from a set of stills of a checkerboard using the well-established camera calibration toolbox by Bouguet[1]. The toolbox is based on the pinhole camera model and employs the radial-tangential distortion model established by Brown [15].

The transformations between the cameras and the IMU as well as the time delay is estimated using a unified framework, presented in our previous work [12] and specifically developed to estimate filter and communication delays to deliver optimal synchronization of sensors. The framework is based on the idea of parameterizing time-variant quantities as B-splines—introduced in detail in [16]—and solving for these as well as a set of time-invariant calibration parameters in a batch optimal fashion. Apart from requiring fewer parameters when fusing measurements of significantly different rates such as images and inertial data, this approach allows for an accurate estimation of the fixed time delay between camera and IMU. Like other frameworks [9], [10], the calibration procedure requires waving the setup in front of a checkerboard, while exciting all rotational degrees of freedom sufficiently in order to render the displacement of camera and IMU well observable. We also experimented with incorporating the calibration for the stereo extrinsics directly into the unified calibration framework, but observed degraded performance when used in visual-inertial SLAM, an explanation to which may be that the setups between calibration and SLAM vary (mostly as far as scene depth is concerned).

The calibration process describes the position and orientation of the IMU with respect to the world frame in continuous-time, which also includes a continuous-time representation of respective derivatives (velocity, acceleration, and angular velocity). Furthermore, both accelerometer and gyroscope biases—modeled as random walks—are represented as continuous-time quantities. The calibration may then be formulated as a batch optimization that combines reprojection error $\mathbf{e}_y$ of checkerboard corners with errors on the acceleration $\mathbf{e}_\alpha$ and $\mathbf{e}_\omega$, as well as terms concerning the compliance of the biases with the random walk processes ($\mathbf{e}_{b_a}$ and $\mathbf{e}_{b_\omega}$).

[1]Available at http://www.vision.caltech.edu/bouguetj/calib_doc/

V. VISUAL-INERTIAL MOTION ESTIMATION

Since the sensor was designed to perform real-time visual-inertial motion estimation, we applied our framework [5] to an outdoor dataset. In short, the method is inspired by recent advances in purely vision-based SLAM that solve a sparse non-linear least-squares problem. Such approaches optimize the reprojection error of a fairly large number of landmarks as observed by various camera frames. Our method tightly integrates inertial measurements into the cost function J by combining reprojection error $\mathbf{e}_\mathrm{r}$ with an IMU error term $\mathbf{e}_\mathrm{s}$ obtained from propagation using standard IMU kinematics in-between successive image frames:

$$J(\mathbf{x}) := \sum_{i=1}^{I}\sum_{k=1}^{K}\sum_{j\in\mathcal{J}(i,k)} {\mathbf{e}_\mathrm{r}^{i,j,k}}^T \mathbf{W}_\mathrm{r}^{i,j,k}\mathbf{e}_\mathrm{r}^{i,j,k} + \sum_{k=1}^{K-1} {\mathbf{e}_\mathrm{s}^{k}}^T \mathbf{W}_\mathrm{s}^{k}\mathbf{e}_\mathrm{s}^{k}. \tag{3}$$

$\mathbf{x}$ denotes the variables to be estimated, composed of the states at all camera snapshot time steps k, as well as all the 3D positions of the landmarks. Note that the states cover not only 6D poses, but also the velocity as well as biases of both gyroscope and accelerometer sensors. The velocity is needed for state propagation in-between time steps, and the slowly varying biases are tracked in order to remove them from gyro and accelerometer readings. i stands for the camera index of the sensor assembly, and j for the landmark index. Landmarks visible in the i^{th} camera are summarized in the set $\mathcal{J}(i,k)$. Furthermore, $\mathbf{W}_\mathrm{r}^{i,j,k}$ denotes the information matrix of reprojection errors related to detection uncertainty in the image plane. Finally, $\mathbf{W}_\mathrm{s}^{k}$ represents the information of the k^{th} IMU error, as obtained from the IMU sensor noise models, provided by the manufacturer (see Table I). We furthermore include the extrinsic calibration of the cameras in the optimization.

This fully probabilistically motivated batch optimization problem over all cameras and IMU measurements quickly grows intractable. We therefore bound the optimization window by applying the concept of marginalization. This allows us to keep a fixed number of keyframes that are arbitrarily spaced in time and that are still related to each other with (linearized) IMU error terms. Consequently, drift during stand-still is avoided, and nevertheless we are able to track dynamic motions.

VI. RESULTS

A. *Sensor Synchronization*

The box plot in Fig. 6 depicts the effect of exposure-compensated sensor synchronization in comparison to a synchronization scheme, where the camera trigger is temporally aligned with polling the IMU. For each synchronization paradigm, we collected about ten datasets for three fixed exposure times by dynamically moving the sensor setup in front of a checkerboard. The algorithm outlined in Section IV was used to estimate the time-offset between the measurements. The figure clearly shows the exposure dependency of the inter-sensor delay for the synchronization where the camera trigger events are equally spaced in time. In addition, a fixed offset becomes apparent, which can be estimated when extrapolating the graph for zero exposure time. As detailed on in Section III-B, the sensor setup compensates for the exposure as well as for the fixed time-offset, resulting in an average inter-sensor delay of only about 7 μs.

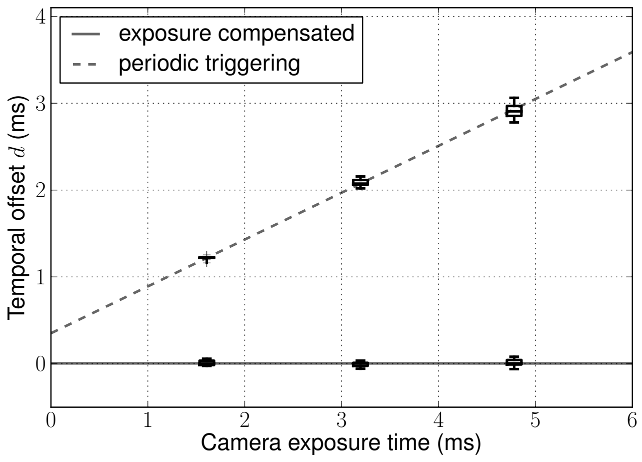

Fig. 6: Results for compensating relative delays of camera and IMU. The dotted line marks the estimated time offset between camera and IMU for a synchronization scheme, where the camera is triggered periodically and the timestamp represents the trigger time. This paradigm clearly results in an exposure dependent delay. Note that there also exists a fixed time-offset, which is induced by filter and communication delays in the IMU and can be estimated by extrapolating for zero exposure time. Our setup compensates for both types of delay, resulting in an almost perfect synchronization with an average estimated delay of only about 7 μs.

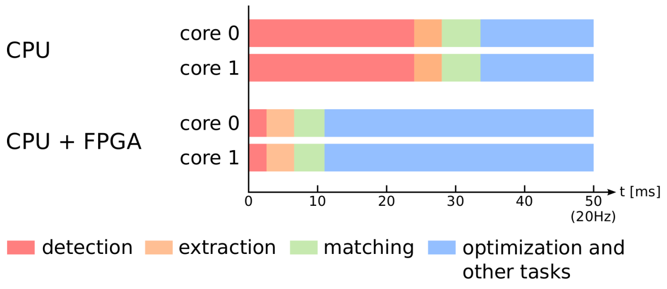

Fig. 7: Profiling for visual-inertial SLAM with and without FPGA accelerated keypoint detection on a Core2Duo. Detection complexity is directly related to camera resolution and consumes a significant amount of time. Outsourcing this operation to the FPGA frees up resources and thus enables processing on resource-constrained platforms, larger optimization windows, or other tasks.

B. *Timing*

Figure 7 shows profiling results for the visual-inertial SLAM system. Timings were generated on our flying platform equipped with a Core2Duo host computer. The sensor assembly was operated in a two-camera configuration, with both cameras running at 20 Hz, and with an IMU rate of 200 Hz.

The most expensive operation in this configuration is keypoint detection using an SSE-accelerated CPU implementation of Harris corners, followed by optimization in the visual-inertial SLAM backend algorithm. With an optimization window of more than five keyframes, the optimization is not able to finish in time and starts dropping frames. Using the FPGA for corner detection resolves this issue.

The computational complexity of the detection further grows when camera resolution or frame rate is increased, or when more cameras are integrated. Outsourcing detection to the FPGA thus significantly reduces CPU load. The remaining parts of the visual-inertial motion estimation algorithm are then largely independent of the system's hardware configuration.

C. *Visual-Inertial Motion Estimation Evaluation*

We recorded a dataset walking around the ETH main building. The sequence contains changing illumination, varying depth, and dynamic objects such as people and cars. The length of the trajectory was 700 m. Two video streams were captured at 20 Hz and the IMU at 200 Hz. Processing was performed with the algorithm outlined in Section V.

Fig. 8 shows the trajectory and structure reconstruction manually overlaid onto an orthophoto. The position error at the end of the trajectory amounts to 5 m laterally and 1 m vertically, thus about 0.7 % of the distance traveled. Note that no loop-closure constraint was applied when reaching the point of origin.

VII. CONCLUSION AND OUTLOOK

This work presented the design of a time-synchronized, calibrated sensor head which is targeted at mobile robotic applications in need of accurate, robust, real-time pose

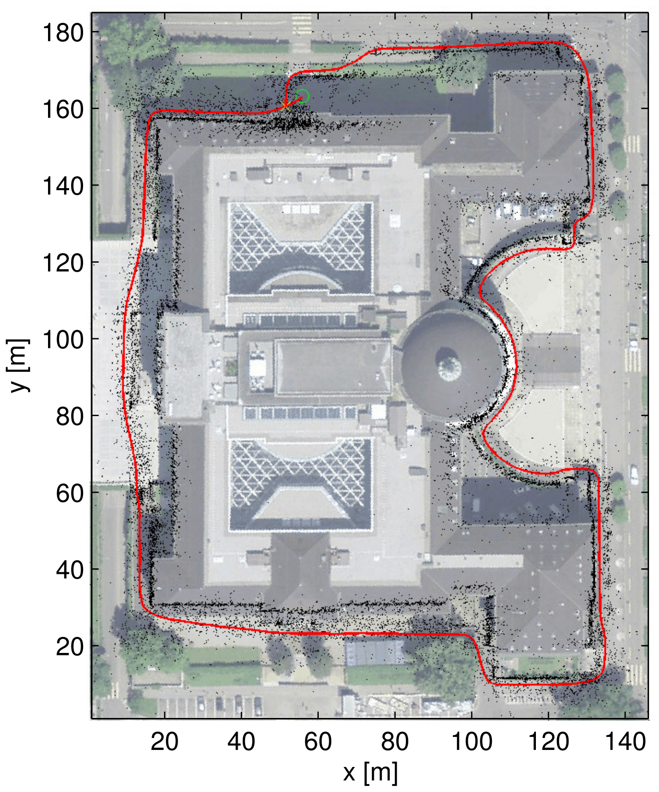

Fig. 8: Reconstructed trajectory (red) and estimated landmarks (black) for a hand-held sequence with the SLAM system in stereo configuration. A distance of 700 m around the ETH main building was covered, and drift accumulates to approximately 5 m laterally and 1 m vertically.

estimation and mapping in uncontrolled environments. Hardware synchronization includes compensation for variable shutter opening, resulting in provably virtually zero time offset between images and IMU measurements. Low-level image processing tasks such as keypoint detection were implemented in programmable hardware in order to speed up processing and free CPU resources. The measurements taken by the presented sensor head were finally fed to a tightly-coupled real-time visual-inertial SLAM framework, the output of which demonstrated the capabilities of the sensor head.

The modular design is ready for integration of higher resolution imagers. Our future activities will on the one hand focus on integration on different platforms ranging from fixed-wing unmanned aircraft to legged robots. On the other hand, we plan to port a light-weight visual-inertial SLAM solution onto the ARM of the sensorhead, in order to obtain a true "SLAM in a box" module.

ACKNOWLEDGMENT

The research leading to these results has received funding from *armasuisse Science and Technology*, project No. 050-23, research contract No. 8003501880.

This project also received funding from the *Swiss Commission for Technology and Innovation (CTI)*, project No. 13394.1 PFFLE-NM (Visual-Inertial 3D Navigation and Mapping Sensor), and from the European Commission's Seventh Framework Program under grant agreement nr. 285417 (ICARUS), nr. 600958 (SHERPA), and nr. 231143 (ECHORD/TUAV).

The authors would also like to thank Markus Bühler, Dario Fenner and Fabio Diem for mechanical design and fabrication, and Simon Lynen for support in driver development.

REFERENCES

[1] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *Robotics Automation Magazine, IEEE*, vol. 13, no. 2, pp. 99–110, 2006.

[2] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

[3] H. Strasdat, A. Davison, J. M. M. Montiel, and K. Konolige, "Double window optimisation for constant time visual slam," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011, pp. 2352–2359.

[4] J. Nikolic, M. Burri, J. Rehder, S. Leutenegger, C. Huerzeler, and R. Siegwart, "A uav system for inspection of industrial facilities," in *Aerospace Conference, 2013 IEEE*. IEEE, 2013, pp. 1–8.

[5] 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.

[6] C. Ahlberg, J. Lidholm, F. Ekstrand, G. Spampinato, M. Ekstrom, and L. Asplund, "Gimme-a general image multiview manipulation engine," in *Reconfigurable Computing and FPGAs (ReConFig), 2011 International Conference on*. IEEE, 2011, pp. 129–134.

[7] K. Schmid and H. Hirschmüller, "Stereo vision and imu based real-time ego-motion and depth image computation on a handheld device," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany, May 6-10 2013.

[8] M. Fleps, E. Mair, O. Ruepp, M. Suppa, and D. Burschka, "Optimization based IMU camera calibration," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 3297–3304.

[9] J. Kelly and G. Sukhatme, "Fast relative pose calibration for visual and inertial sensors," in *Experimental Robotics*. Springer, 2009, pp. 515–524.

[10] F. Mirzaei and S. Roumeliotis, "A kalman filter-based algorithm for IMU-camera calibration: Observability analysis and performance evaluation," *Robotics, IEEE Transactions on*, vol. 24, no. 5, pp. 1143–1156, 2008.

[11] J. Kelly and G. S. Sukhatme, "A general framework for temporal calibration of multiple proprioceptive and exteroceptive sensors," in *12th International Symposium on Experimental Robotics, 2010*, Delhi, India, Dec 2010.

[12] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[13] C. Harris and M. Stephens, "A combined corner and edge detector." in *Alvey vision conference*, vol. 15. Manchester, UK, 1988, p. 50.

[14] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Proceedings of the 9th European conference on Computer Vision-Volume Part I*. Springer-Verlag, 2006, pp. 430–443.

[15] D. C. Brown, "Close-range camera calibration," *Photogrammetric engineering*, vol. 37, no. 8, pp. 855–866, 1971.

[16] P. T. Furgale, T. D. Barfoot, and G. Sibley, "Continuous-time batch estimation using temporal basis functions," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, St. Paul, MN, 14-18 May 2012, pp. 2088–2095.