IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received June 18, 2021, accepted July 11, 2021, date of publication July 20, 2021, date of current version July 28, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3098856

Multiple Master-Slave FPGA Architecture of a Stereo Visual Odometry

CHIANG-HENG CHIEN[1], CHEN-CHIEN JAMES HSU[2], (Senior Member, IEEE), AND CHIANG-JU CHIEN[3], (Member, IEEE)
[1]Electrical and Computer Engineering, Brown University, Providence, RI 02912, USA
[2]Department of Electrical Engineering, National Taiwan Normal University, Taipei 10610, Taiwan
[3]Department of Electronic Engineering, Huafan University, New Taipei City 22301, Taiwan

Corresponding author: Chiang-Ju Chien (cjc@cc.hfu.edu.tw)

This work was financially supported by the Ministry of Science and Technology, Taiwan, under Grant no. MOST 109-2221-E-211-003, and the "Chinese Language and Technology Center" of National Taiwan Normal University (NTNU) from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan, and Ministry of Science and Technology, Taiwan, under Grants no. MOST 109-2634-F-003-006 and MOST 109-2634-F-003-007 through Pervasive Artificial Intelligence Research (PAIR) Labs.

ABSTRACT Estimating relative camera pose is the key problem of visual odometry (VO). To achieve better efficiency, sparse key-points are usually relied on for the estimation. Yet, feature extraction and matching are still computationally demanding, hindering the overall VO from real-time processing. Exploiting the superiorities of an FPGA in terms of high efficiency, low power consumption, and low cost, this paper proposes a multiple master-slave FPGA architecture for an SIFT-based stereo VO. The master-slave design enables high reconfiguration for the data throughputs among various modules. These modules include SIFT, matching, pose estimation, and their corresponding controllers. In the SIFT module, hardware implemented image pyramid is proposed, where scales are determined off-line via a minimization approach. Local linear exhausted search (LES) matching is considered for both the stereo and the frame matching. In the pose estimation module, a novel hardware design of deriving closest orthogonal matrix for 3D-3D correspondences of relative pose estimation is proposed. Experimental results show that 33.2 fps can be achieved using KITTI dataset without the need of a large number of hardware resources. The proposed reconfigurable design also facilitates its expansions of adopting CCD cameras as well as developing SLAM and other applications.

INDEX TERMS Master-slave hardware architecture, visual odometry, FPGA, SIFT, Avalon bus.

I. INTRODUCTION

Characterized by low cost, low power consumption and rich information, cameras are becoming the highly interested sensors for many visual autonomous systems. In the wake of increasing need for visual navigation, VO, which provides relative camera poses from two successive images, is critical in any vision-based robotic systems. Because of its efficiency and reliability, feature-based VO is one of the preferable approaches among the research communities. It estimates relative pose geometrically using extracted features and their correspondences between frames. Features such as FAST [1] or SURF [2] are widely used. However, despite of their efficiency, the feature descriptions could be unstable in scenarios where huge illumination changes or rotation changes appeared in the environments, compared to SIFT [3]. Yet SIFT features are encoded by 128 dimensions of descriptors, requiring heavy computational burden and thus the applicability in real-time computer vision areas is limited.

In addition to feature extraction, feature matching is another time consuming process. In the case of stereo VO, not only features of the stereo images are needed to be matched to determine disparities, features between sequential frames are also required to be matched for relative pose estimation. Dealing with correspondences of thousands of features in one image to another would inevitably drags down the overall efficiency, which posts an unfavorable condition for real-life, real-time application. Although some approaches such as KLT tracking [4] and FLANN [5] provide faster matching computation, they sacrifice robustness to reduce computational cost. Furthermore, considering the efficiency of data acquisitions from memories, [4] and [5] do not drastically reduce the computational time if parallel computing design is used.

Based on either 2D-2D, 2D-3D, or 3D-3D point correspondences between the current and the previous frame,

The associate editor coordinating the review of this manuscript and approving it for publication was Heng Wang.

respective algorithms like essential matrix, PnP [6], and ICP [6], are responsible for providing 6-DOF relative camera poses. These approaches only require linear algebra computation such as singular value decomposition (SVD), and thus the computational cost is relatively low. However, considering real-time processing, data delays from the aforementioned feature extraction and matching make a VO incapable of meeting up such a requirement. As a result, there are more and more approaches implementing a VO system on parallel computing platforms. In fact, according to [7], using either Digital Signal Processors (DSP), Field Programmable Gate Arrays (FPGA) or Graphical Processing Unit (GPU) or their heterogeneous approach on computer vision is a desirable choice. A GPU+CPU heterogeneous design of a VO [8] is an example, which yields better performances than CPU-only approach. Nevertheless, extensive comparisons made in [7] that considers balances between efficiency, data throughput, and power consumption, show that FPGA provides outstanding advantages over DSP and GPU. This is also supported by [10] that System-on-Chip (SoC) like FPGA or Application-Specific Integrated Circuits (ASIC) prototype implementation strategy demonstrates significant benefits. More specifically, [9] compared the use of desktop GPU, FPGA, multi-core DSP, single and multiple core CPU in visual navigation including visual odometry. The results show that FPGA obviously surpassed other computing platforms. Therefore, leveraging FPGA for VO systems have been emerging in the literatures [11].

In [12], a unified framework for feature-based localization system called EUDOXUS was proposed based on an FPGA. It involves a visual-inertial odometry (VIO) frontend and a simultaneous localization and mapping (SLAM) backend. Such a large system requires large memory capacities, and thus to fit the overall system on one FPGA chip, [12] developed a holistic approach to optimize the on-chip memory and the logic usage. To demonstrate the flexibility of [12], not only a Zynq FPGA was used for evaluations, but high-end Vertex-7 FPGA was also employed. According to its experiments summarized in [9], VIO frontend has around 4.8x speedup compared to CPU. Although a good performance was given, [12] used multi-state constraint Kalman filter (MSCKF) to estimates poses, which contributes inferior performances compared to non-filtering family, as stated in [13].

Another work [14] named Navion was proposed to provide a fully integrated stereo non-filtering based VIO system on an ASIC using lightweight Shi-Tomasi corners. It also incorporates a graph optimization backend to regularize both poses and 3D features. To reduce on-chip memory size, images were compressed and both structured and unstructured sparsity of features were exploited. As a result, a total of $4.1\times$ memory saving was achieved. Evaluating the system on a dataset, [14] offers 20 fps while consuming only a little power.

Similar to [14], a binocular VIO algorithm-hardware co-design approach was proposed [10], aiming at trading off algorithm performance and hardware resource. Shi-Tomasi corners were extracted as features in the system, and ARM software was introduced to obtain optimal parameters for the algorithm prior to developing a fully integrated chip. This minimizes the hardware resources. However, [14] and [10] employed an inertial measurement unit (IMU) sensor that led to higher development complexity than using only one or two cameras. Without an IMU, hardware resources expenditure can be further reduced.

Aside by full hardware implementation of a VO system mentioned above, hardware-software (HW/SW) co-design is another intriguing approach for accelerating VOs. One of the main advantages of using such an architecture is that, since some operations are not easily to be implemented in a hardware, software can be prior to be involved in for performing those operations before full-hardware implementation is achieved, which is demonstrated by [10]. Moreover, HW/SW co-design characterizes high flexibility, meaning that potential applications being integrated to the HW/SW co-design system can be easily carried off. Additionally, hardware modules that require less parallelism can be taken over by software, reducing the overall hardware resources. This facilitates the developments on low level FPGA where bounded hardware resources are given. For example, in [15], a HW/SW co-design architecture of a stereo VO was released for rover navigation on Mars. It built five distinct VO pipelines using different feature extractions and matching methods, where the most reliable pipeline used Harris corner detection and SIFT descriptors based on its comprehensive evaluations. Features are matched based on both the similarities and the image locations, and camera motion was estimated using absolute orientation. To account for the I/O scenarios and HW/SW co-design, [15] developed a communication scheme between programmable logic and processing system via Xillibus. Because of the combination of hardware and software, [15] does not depend on hardware resources. Nonetheless, its speed is yet to be improved, which provides nearly 1 frame per second (fps).

Using ground robots, [16] fused cameras, IMU, and wheel odometry sensors to construct a HW/SW co-design VIO system. Time-costly feature extraction and matching was accelerated by an FPGA, while an ARM software was in responsible for the floating point arithmetic computation required in pose estimation. Sadly, the runtime is far from satisfactory, and no intensive evaluations were provided to validate the system performances. Particularly when only 200 features were used, the robustness of [16] is yet to be proved.

Alike [10], [12], and [16] where an additional IMU is employed, [17] also built a VIO system in a HW/SW co-design manner using FAST features. However, unlike [10], [12], and [16], the processing of the IMU as well as the synchronization between the camera and the IMU were taken over by software, while the rest of the system was managed by hardware. This downsized the development complexity. Besides, it targeted the optimization of the VIO algorithm to reach ultra-low latency and power, where only 2.2 mW energy was relied on for the system. Its experiments using a dataset exhibited that the pose estimation accuracy was comparable to pure ARM-based implementation, while a speedup of 10 times faster than an ARM was achieved. Nevertheless, it fell into the same drawback appeared in [12] by using extended Kalman filter (EKF) for providing pose outputs.

In all of these HW/SW co-design works, it shows promising benefits of adopting such a design for VO systems. Albeit many advantages brought by a HW/SW co-design, the difference of operational frequencies between software and hardware sometimes lowers the overall performances and scales up the development complexity. For instance, in [17], its use of FPGA and ARM respectively operate at 70 MHz and 1.2 GHz. As a consequence, this paper builds on our previous work [18] and carries out a multiple master-slave HW/SW co-design stereo VO architecture using SIFT features on an Altera FPGA board. Below is the summary of the contributions and findings of this paper:

A. ARCHITECTURE

A multiple master-slave HW/SW architecture is constructed. SIFT feature extraction, matching, and pose estimation are achieved by hardware, whereas an Altera's Intel soft core Nios II is only responsible for managing data transfer between hardware modules and off-chip memories. Additional controller modules are designed to coordinate data read from or written to the off-chip memories via Avalon Bus.

B. HARDWARE IMPLEMENTED PARALLEL IMAGE PYRAMID

The FPGA-implemented SIFT used in this paper is borrowed from the state-of-the-art method [19], and a hardware implemented parallel image pyramid is proposed. To determine proper scales for each level of the pyramid, a constrained objective function is designed based on the differences of Gaussian image intensities obtained by software and hardware. To minimize the objective function, a genetic evolutionary algorithm [20], a heuristic optimization method capable of dealing with complex constrained problems via evolutionary computation including crossover, mutation, and selection, is employed.

C. FEATURE MATCHING HARDWARE MODULE

Because of the Avalon Bus burst mode, the core of the matching module proposed in [21] is extended to allow efficient and reliable local linear exhaustive search matching. The stereo and frame-to-frame matching are achieved based on both the geometrical image locations and the SIFT descriptors.

D. POSE ESTIMATION HARDWARE MODULE

3D-3D iterative closest point (ICP) registration method [22], [23] is used for pose estimation. To avoid large memory size required by SVD for deriving nearest orthogonal matrix, this paper proposes a technique that incorporates Denman-Beavers (DB) algorithm [29] as a matrix square root solver and Taylor approximation to obtain fast and accurate result.

E. HIGH SPEED AND LOW ESTIMATION ERROR

Using 640×480 VGA images cropped from images of the KITTI stereo dataset [24], the proposed VO achieves 33.2 fps without consuming lots of hardware resources, outperforming the methods mentioned above. Furthermore, compared to a desktop PC, the proposed system provides a speedup of $4{,}955\times$. The average estimation errors in terms of relative pose error (RPE) and absolute trajectory error (ATE) [25] are 0.66 and 1.084 meters, respectively.

F. FIRST PAPER PROPOSING MASTER-SLAVE FPGA ARCHITECTURE ON VISUAL ODOMETRY

To the best of our knowledge, this is the first paper that designs and applies multiple master-slave HW/SW co-design FPGA architecture on a VO. The design not only reduces heavy dependencies of hardware on the software computations appeared in the above-mentioned publications, but it also shapes the system in flexible and reconfigurable manners, allowing better integration to other potential navigation applications such as SLAM.

The rest of this paper is organized as follows. Section II gives the overview of the multiple master-slave architecture for the VO algorithm. In Section III, the derivation of scales for SIFT parallel image pyramid is described, while the SIFT and its control module are presented in Section IV. The local matching, pose estimation, and their corresponding control modules are respectively shown in Section V and VI. Finally, the experiments are provided in Section VII, and Section VIII concludes this paper with discussions on the future works of the proposed approach.

II. OVERVIEW OF THE PROPOSED FPGA ARCHITECTURE

Multiple master-slave HW/SW co-design FPGA architecture for a VO proposed in this paper is shown in Figure. 1. The hardware includes an on-chip Altera DE2i-150 core and off-chip memories. The former consists of on-chip modules, i.e., an SDRAM controller as well as SIFT Top, LES Top, ICP Top modules and their corresponding controllers. The latter, on the other hand, includes a Flash and two SDRAM memories. Because the I/O ports are shared by both of the two SDRAMs, these two memories are together controlled by the Nios II soft core built on the chip.

Basically, the SIFT Top, LES Top, ICP Top modules are coordinated by the Nios II processor via Avalon Bus. Data are read from or written to the SDRAMs which are also controlled by the Nios II. The read/write modes between on-chip controller modules and the Nios II processor follows the rules of Avalon Bus in a master-slave manner. The master-slave mechanism is depicted by the arrows between the Avalon Bus and controller modules in Figure. 1, where the direction of the arrows points from master to slave. Take SIFT Controller as an example. When it reads image data and parameters from the SDRAMs to the SIFT Top module

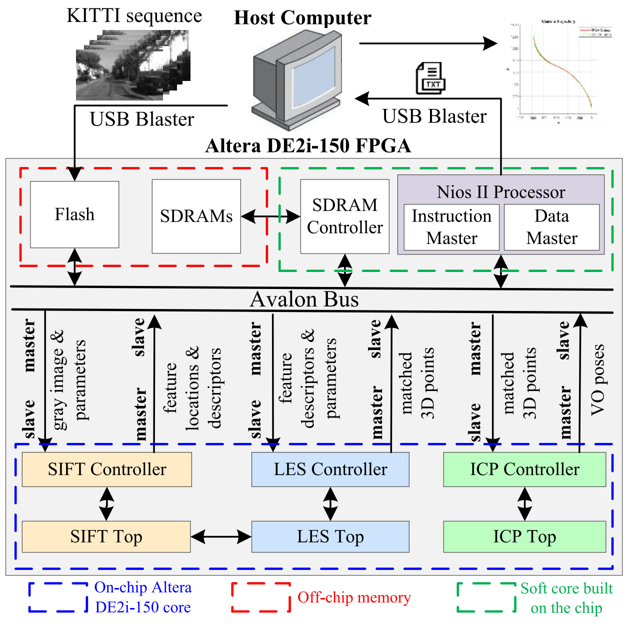

FIGURE 1. **The architecture of the proposed multiple master-slave HW/SW co-design VO system.**

via Avalon Bus and Nios II, Nios II is the master while the SIFT Controller is the slave, respectively. Conversely, when the SIFT Controller is required to store SIFT features back to the SDRAMs, Nios II now becomes the slave while the SIFT Controller is the master. By using such a structure, the overall architecture is attached with high flexibility and reconfigurability. In other words, as long as the master and slave interfaces are specifically designed, any top module shown in Figure. 1 can be easily replaced without adjusting other modules. For instance, the SIFT Top module can be replaced by FPGA-based FAST algorithm without the need to reorganize the matching and pose estimation modules. Moreover, as hardware modules are managed by the Nios II processor, pipeline computation can be easily achieved.

Details of how the proposed system is carried out is described as follows.

1) Prior to the beginning of the VO, a sequence of stereo gray images is written to the Flash memory from the host computer via Nios II host file system using a USB cable.
2) Generally, when the VO starts, the first left image data is written from the Flash memory to the SDRAMs. Nios II processor (master) initializes and transfers image data from the SDRAMs to the SIFT Controller (slave) via Avalon Bus. Then, the SIFT Controller passes the image data to the SIFT Top module for extracting SIFT features. After it is finished, SIFT Controller (master) writes feature locations and descriptors from the SIFT Top module back to the SDRAMs via Avalon Bus. This allows Nios II (slave) to receive the storage addresses. Subsequently, the right image is fed from the Flash to the SDRAMs, and repeats the SIFT algorithm same as the above.
3) Until SIFT features are completely extracted from the left and right images, Nios II (master) starts to pass feature locations and descriptors addresses via Avalon Bus to the LES Controller (slave). At this moment, local LES stereo matching is able to perform stereo matching by the LES Top module. The correspondences are formulated as 3D points. Alike the SIFT features, these points are also written to the SDRAMs by the LES Controller (master) via Avalon Bus. Nios II (slave) receives the 3D points data addresses, facilitating the execution of the same procedure for the next stereo images.
4) As long as two stereo images are completely gone through SIFT and matching, the LES Controller (slave) is initialized by the Nios II (master) to match frame-to-frame features. This step is not only similar to the previous step, but the same hardware modules are also used. Followed by the need to store 3D correspondences, Nios II (slave) receives the storage addresses from the LES Controller (master).
5) With the two sets of 3D correspondences, Nios II (master) transfers the data to the ICP Controller (slave) via Avalon Bus. The ICP Top module is subsequently triggered to give relative pose estimations. The poses are stored to the SDRAMs from the ICP Controller (master) via Avalon Bus, where the poses outputs can be taken by the Nios II (slave).
6) Finally, the whole process repeats until all the images from the Flash memory are completely performed. At this moment, Nios II processor is allowed to pack all the poses associated to each frame as a compressed text file using the zip file system. The compressed file is then transferred to the host computer via the USB cable for displaying the trajectory results.

III. PARALLEL IMAGE PYRAMID

The FPGA-based SIFT algorithm released in [19] efficiently gives promising features from an image. However, to further boost its performance, we propose a new structure of an image pyramid as shown in Figure. 2 (a) and (b). The former and the latter respectively show the software and hardware implemented image pyramid. In this paper, 6 levels of images are used. In software, Gaussian blur is processed to create Gaussian images $G_{s,0}, \ldots, G_{s,5}$ sequentially using their corresponding scales $\sigma_{s,0}, \ldots, \sigma_{s,5}$. Each scale is computed based on the base scale $\sigma_{s,0}$. In contrast, the parallel assignments of scales to each level of image $G_{h,0}, \ldots, G_{h,5}$ in hardware have to be well-defined. If improper scales are used, the Gaussian blurring outcomes computed by hardware could share low similarities to the ones computed by software. This could implicitly lead to feature unreliability. Hence, this paper formulates such an issue as a two-step optimization problem, and a genetic algorithm is used as the solver to find the optimal scales for the hardware implemented parallel image pyramid, namely, $\sigma_{h,0}, \ldots, \sigma_{h,5}$.

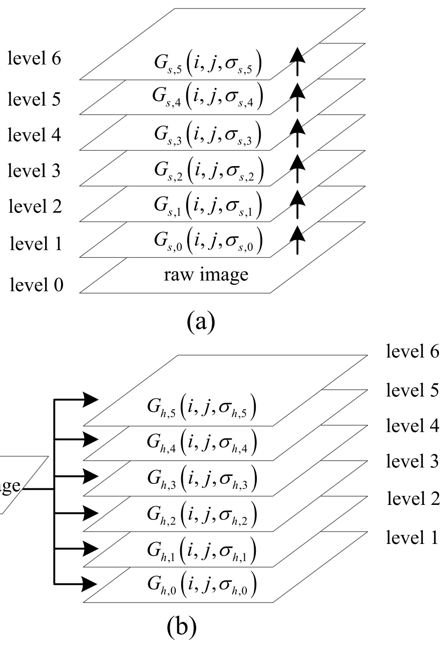
level 6
level 5
level 4
level 3
level 2
level 1
level 0
$G_{s,5}(i,j,\sigma_{s,5})$
$G_{s,4}(i,j,\sigma_{s,4})$
$G_{s,3}(i,j,\sigma_{s,3})$
$G_{s,2}(i,j,\sigma_{s,2})$
$G_{s,1}(i,j,\sigma_{s,1})$
$G_{s,0}(i,j,\sigma_{s,0})$
raw image
(a)
raw image
level 0
$G_{h,5}(i,j,\sigma_{h,5})$
$G_{h,4}(i,j,\sigma_{h,4})$
$G_{h,3}(i,j,\sigma_{h,3})$
$G_{h,2}(i,j,\sigma_{h,2})$
$G_{h,1}(i,j,\sigma_{h,1})$
$G_{h,0}(i,j,\sigma_{h,0})$
level 6
level 5
level 4
level 3
level 2
level 1
(b)

FIGURE 2. Image pyramid implemented by software and hardware. (a) Sequential Gaussian blurring in software. (b) Parallel image pyramid proposed in this paper.

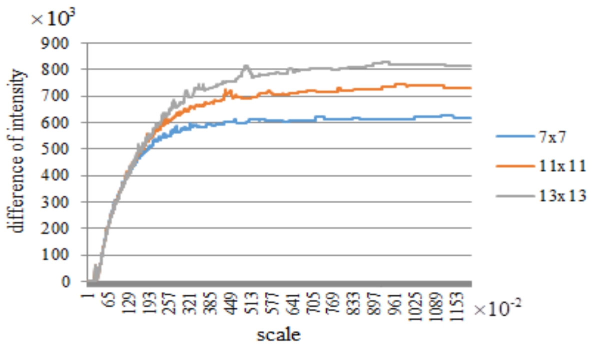

FIGURE 3. Blurring effectiveness with respect to scales using different mask sizes.

1) **First Step**: To define the scale values for most of the scenarios, we use numerous images from datasets like KITTI and TUM [25] with various scenes. The images are calibrated and rectified, and all the image size is restricted to 640 × 480. Then, the scale of top level of the image $\sigma_{s,5}$ that shows the largest blurring effectiveness with respect to the raw image is derived using different mask sizes including 7×7, 11×11, and 13×13. The result is presented in Figure. 3, where the difference of intensities between the original image and the blurring image with respective to the scale using three masks is given. It can be clearly seen that all the three masks are able to provide maximal blurring with σ_5 approximate be 10. However, the larger the mask size, the more blurring is contributed. This is intuitive because larger mask convolves more image data with a Gaussian kernel. However, instead of using a 13×13 mask, considering the limited memory usage in hardware, this paper uses 7×7 masks identical to [19]. As a result, the main purpose in this step is to find the scales of every image level in software, where $\sigma_{s,5}$ maps to $\sigma_{h,5} = 10$ that gives the minimal relative scales between levels with respect to the original SIFT algorithm. Moreover, the problem is subject to an inequality where the error of Gaussian blur images between $G_{s,5}$ and $G_{h,5}$ is smaller than a predefined threshold T. This formulates a constrained optimization problem that solves the optimal values of $\sigma_{s,0}$ and k, denoted by $\sigma^*_{s,0}$ and k^*, respectively, as shown below.

$$\sigma^*_{s,0}, k^* = \underset{\sigma_{s,0},k}{\arg\min} \sum_{n=0}^{4} \left(\left|\Delta\hat{\sigma}_{n+1,n}\right| - \left|\Delta\sigma_{n+1,n}\right|\right)$$
$$s.t. \sum_{j=1}^{640}\sum_{i=1}^{480} \left|G_{s,5} - G_{h,5}\right| < T,$$
$$\sigma_{h,5} = 10 \tag{1}$$

where k, defined as follows, is the variable of $\sigma_{s,n}$ that determines the extent of scaling with respect to the base scale in level n of the pyramid.

$$\sigma_{s,n} = \sqrt{\left(k^n\sigma_{s,0}\right)^2 - \left(k^{n-1}\sigma_{s,0}\right)^2} = \sigma_{s,0}k^{n-1}\sqrt{k^2-1} \tag{2}$$

In (1), $\Delta\hat{\sigma}_{n+1,n}$ and $\Delta\sigma_{n+1,n}$ are respectively the relative scale between the level $n+1$ and n of the original SIFT and software (Figure. 2 (a)), namely,

$$\Delta\sigma_{n+1,n} = \sigma_{s,n+1} - \sigma_{s,n}. \tag{3}$$

Hence, given $\sigma_{h,5} = 10$, the constrained minimization of (1) focuses on the optimal continuity of the scales of the levels of image pyramid by giving proper values of $\sigma_{s,0}$ and k, accordingly yielding $\sigma_{s,1}, \ldots, \sigma_{s,5}$. As a result, even though $\sigma_{s,0}$ is not 1.6 as suggested by the author of the SIFT algorithm, and the blurring effectiveness of the final Gaussian image $G_{s,5}$ is not as much as the original SIFT does, the continuity of scales among all Gaussian images is similar. This allows better construction of difference of Gaussian for interest points extraction in the SIFT algorithm.

2) **Second Step**: With the optimal base scale and k, the second step targets the derivation of a set of optimal scales from level 2 to level 5 used in hardware, i.e., $\sigma_{h,1}, \sigma_{h,2}, \ldots, \sigma_{h,4}$, given $\sigma_{h,5} = 10$ and $\sigma_{s,0} = \sigma_{h,0}$. To do so, we minimize the error summing up the differences of Gaussian image intensities from level 2 to level 5 obtained by the software and the hardware in the pyramid. This formulates a non-constrained optimization for obtaining the optimality of $\sigma_{h,1}, \ldots, \sigma_{h,4}$ denoted as $\sigma^*_{h,1}, \ldots, \sigma^*_{h,4}$ shown below.

$$\sigma^*_{h,1}, \ldots, \sigma^*_{h,4} = \underset{\sigma_{h,1},\ldots,\sigma_{h,4}}{\arg\min} \sum_{m=2}^{5}\sum_{j=1}^{640}\sum_{i=1}^{480} \left|G_{s,\mathrm{m}} - G_{h,\mathrm{m}}\right| \tag{4}$$

Using a genetic algorithm, two optimization problems can be solved easily, and the parameters used by the parallel hardware image pyramid can therefore be assigned prior to implementation.

IV. SIFT CONTROLLER AND SIFT TOP MODULES

With the proper parameters of image pyramid, the SIFT Top module along with its corresponding SIFT Controller module can be developed. The architecture is shown in Figure. 4. The SIFT Controller module consists of several modules including Master and Slave Interface, SIFT Slave, and SIFT Interface Control. When it is initialized by the Nios II processor, SDRAMs addresses of gray images and hardware settings such as image size, addresses of output SIFT features, are received by Slave Interface to the SIFT Slave module. The SIFT Slave module coordinates the input data to either a DCFIFO named DCFIFO#1 in the Interface Control module or the finite state machine (FSM) control. Because SDRAMs have a limited frequency bandwidth, reading image data from it while simultaneously writing feature data to it would not allow all the data being read or written continuously at once. Furthermore, the SIFT module in [19] does not have a data valid signal design, meaning that input data discontinuity cannot be handled. Hence, the DCFIFO#1, with 32-bit input and 8-bit output, acts as a buffer, storing image data for the SIFT Top module to read. Since SIFT Top module reads only one fourth of the data width from the DCFIFO#1 at one clock cycle, with substantial large size of DCFIFO#1 the SIFT module is able to receive continuous gray image data. We leave the addition of a data valid signal to the SIFT Top module as one of our future works, as discussed in section VIII.

As for the FSM control, it manages the data flow between the SIFT Controller, SIFT Top, and Nios II processor in a FSM manner. Whether the data is required to be read from SDRAMs or be written to the SDRAMs is decided by the FSM Control which sends a read/write signal to the SIFT Slave module for allocating the associated data address of SDRAMs.

As long as DCFIFO#1 has certain amount of data, SIFT Controller sends a permission to the SIFT Top, allowing the SIFT Top to read image data from DCFIFO#1. Once the SIFT Top module extracts one feature, the feature data including 128 dimensions of SIFT descriptors as well as feature locations, is stored in another DCFIFO named DCFIFO#2. Detecting any data stored in DCFIFO#2, the FSM control module enables writing the feature to the SDRAMs by Master Interface via Avalon Bus based on the corresponding addresses provided by the SIFT Slave module.

Composed of SIFT, Buffer 1312 to 32, and SIFT LES Controller modules, the SIFT Top module extracts SIFT features based on the input image data. The SIFT LES Controller is in charge of coordinating the data input and output of the SIFT Top, as well as the communication with the LES Top module. Similar to our previous work [18], one dimension of a SIFT feature descriptor is encoded by 10 bits, and thus

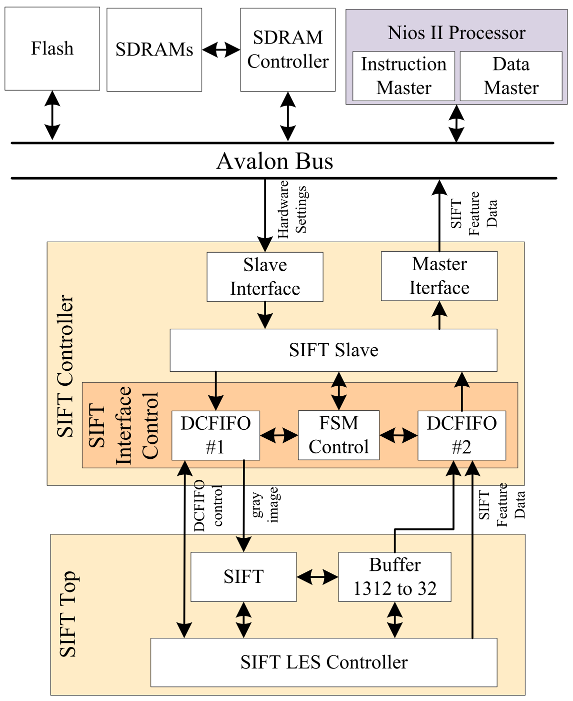

FIGURE 4. SIFT Controller and SIFT Top modules.

a descriptor is 1280-bit long. Concatenated with its image location encoded by 32-bit, a feature is represented totally by 1312 bits. However, DCFIFO#2 used in the SIFT Controller module is 32-bit input and 32-bit output. Therefore, a Buffer 1312 to 32 module is required for sequentially transferring 32-bit data 41 times to the DCFIFO#2. As the SIFT Top completes the extraction, a 1-bit terminal signal packed in a 32-bit data is transferred to the DCFIFO#2. This allows Nios II processor to acknowledge when it is required to pass the SDRAMs storage addresses to the LES modules.

Note that the frequency of Nios II processor is 50 MHz, and SDRAMs as well as SIFT Controller operates at 94 MHz. As for the SIFT Top module, it operates at 100 MHz. Although the low frequency used in the Nios II processor could decrease the overall system efficiency, in the proposed master-slave design system, Nios II only deals with data transfer and hardware module coordination, and thus it does not potentially influence the system performances. Also, the asynchronization of SIFT Controller and SIFT Top shows the need of using DCFIFOs rather than SCFIFOs.

V. LES CONTROLLER AND LES TOP MODULES

Analogous to SIFT Controller and SIFT Top modules, LES Controller and LES Top modules have respectively similar architectures, as shown in Figure. 5. As SIFT features are completely extracted from the stereo images, Nios II processor initializes the LES Controller module by transferring SDRAMs feature data addresses and the number of features. These data are received by Slave Interface to the Match Slave module via Avalon Bus, and are subsequently stored in a DCFIFO named DCFIFO#3. Alike the FSM Control in the

SIFT Interface Control module, FSM Control in Figure. 5 is also in charge of controlling the read/write of data between modules and the SDRAMs. No matter it is a stereo matching or a frame-to-frame matching, a feature from one image is first read from the SDRAMs, followed by all the features from another image being read also from the SDRAMs. This enables the LES Top module to start matching the features.

Different from Figure. 4 where there is a controller in the SIFT Top module, LES Top modules does not have one. Instead, the Buffer 32 to 1312 first packs the 32-bit data read from DCFIFO#3 into a 1312-bit data including a 1280-bit feature descriptor and a 32-bit feature location. The feature location is passed to the Coordinate Hold module, while the whole 1312-bit feature data is given to the LES Matching module. In the LES Matching module, local LES matching algorithm is used. The locality of stereo matching is constrained by the epipolar geometry. However, since SIFT features are stored sequentially from left to right and top to down of the image, it is not able to directly retrieve features satisfying epipolar constraints using only the SDRAMs addresses. In other words, the locations of features are known only after it is read from the SDRAMs. As a result, taking stereo matching for example, to achieve local matching, we sequentially read the feature according to the first right image feature address to the one that has the location exceeding a certain window size of the left feature location. Only the right feature with locations satisfying the epipolar constraint is employed for LES matching. As long as a right feature with location exceeding a certain window size of the left feature location, the rest of the right features are not necessarily read, for they are definitely not the candidates on the epipolar line. Such a mechanism also applies to frame-to-frame matching, where the locality constraint is a window of a feature rather than a epipolar line. Using local LES matching, unrelated features are avoided for correspondences, thereby increasing both computational efficiency and matching reliability.

As for the Coordinate Hold module, feature locations are hold, waiting for the instruction from the LES Matching module. Only if a pair of stereo features is successfully matched does the Coordinate Hold compute the corresponding 3D points. The Euclidean space dimension (x, y, z) of a 3D point is encoded respectively by 12-bit, 11-bit, and 12-bit, and all three dimensions are signed binary numbers. Therefore, a buffer is included in the Coordinate Hold module that decomposes 45-bit 3D point data into two 32-bit data written to DCFIFO#4.

The instruction from the LES Matching module is also given to the Buffer 1312 to 32 module when stereo matching is processing, where the matched left feature data is written to DCFIFO#4. Note that for frame-to-frame matching, it does not need to compute 3D points in Coordinate Hold, and the Buffer 1312 to 32 module is not used. Rather than writing the outputs of the Buffer 1312 to 32 module to DCFIFO#4, frame-to-frame matching only has to write the 3D matches. Therefore, a multiplexer is involved in LES Top that selects the inputs from either Coordinate Hold or Buffer 1312 to 32

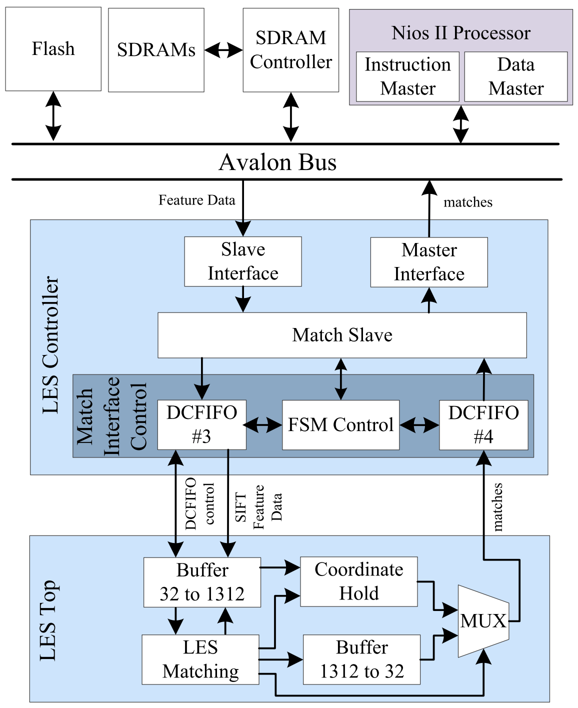

FIGURE 5. LES controller and LES top modules.

modules respectively for frame-to-frame matching and stereo matching.

Alike the SIFT Controller, once there is data in DCFIFO#4, FSM Control sends a write request to Match Slave, and the matches are then written to the SDRAMs by Master Interface via Avalon Bus. A terminal signal is also written to the DCFIFO#4 when the matching finishes. Hence, Nios II processor is able to continuously detect the value of the terminal signal SDRAMs address to investigate whether the it is allowed to retrieve the matching results and proceed to the next step.

Overall, during matching, Nios II processor is only responsible for sending and receiving SDRAMs addresses as well as hardware settings. Therefore, the LES Top module can be processed at 200 MHz, while the LES Controller operates at the same frequency as SIFT Controller which is 96 MHz.

VI. LES CONTROLLER AND LES TOP MODULES

The controller module for the pose estimation has very high similarities to the SIFT and the LES Controller modules. The ICP Controller connected with the ICP Top module is presented in Figure. 6. The ICP Top module contains a ICP FSM Control, governing the operations of the rest of the modules of the ICP Top. Because each 3D point is 45-bit and the ICP algorithm requires lots of multiplications and divisions, data could be loss during integer arithmetic computation. Thus, each point is enlarged to 64 bits. However, there could be hundreds or even thousands of 3D matches, to handle such a large amount of data requires lots of memory storage. To avoid such a problem, we do not use on-chip RAM

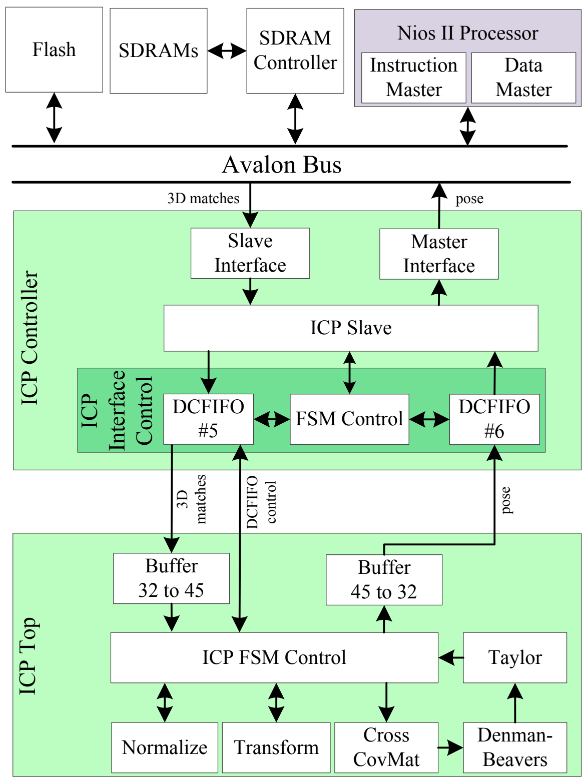

FIGURE 6. ICP Control and ICP Top modules.

to store all the 3D matches data when they are processed by either Normalize or Transform module. Instead, the outputs of these two modules are stored back to the SDRAMs through a Buffer 45 to 32 module as well as DCFIFO#6. Likewise, the inputs are read from the SDRAMs through DCFIFO#5 and Buffer 32 to 45. This is quite different from SIFT and LES blocks where data would not be read again once it is stored in the SDRAMs. Although such a read/write procedure in the ICP block could consume extra time, the burst read/write mode service offered by Avalon Bus still allows the system to perform efficiently. Furthermore, a large amount of on-chip hardware resources can be saved.

Providing frame-to-frame 3D points correspondences by the LES block, in the beginning of the ICP block, the Nios II processor transfers SDRAMs addresses of 3D points and output poses as well as the number of matches to the ICP Controller by Slave Interface via Avalon Bus. The matches are stored in DCFIFO#5 where it is later read by the ICP Top module. When DCFIFO#5 has at least two 32-bit data, a pair of 32-bit is enveloped to a 45-bit three dimensional point data by the Buffer 32 to 45 module. The Normalize module is used to normalize all the 3D matches, and the output is later returned to the ICP FSM Control module before being stored to the SDRAMs through DCFIFO#6. Note that in the Normalize module, the matches are read twice from the SDRAMs. The first time is read so that all matched points can be summed up. Another time is read to divide the summation for normalization.

After normalization, the normalized matches are read from the SDRAMs to the Transform module. The Transform module is responsible for transforming the 3D points in the current camera coordination to the ones in the previous camera coordination given a camera pose. In this paper, the initial pose of the ICP algorithm is defined based on constant motion assumption of visual odometry. As soon as a point is transformed, it is stored back to the SDRAMs.

Followed by the Transform module, the Cross CovMat module is used to derive cross covariance matrix. After that, the camera rotation matrix can be obtained by finding the closest orthogonal matrix of the cross covariance matrix in order for the rotation matrix to meet the properties of special orthogonality SO(3). Generally in the ICP algorithm, SVD is used. However, carrying out SVD on an FPGA requires large memory size to achieve precise integer arithmetic and the performance degrades if operational frequency is not high enough, as can be seen from [27]. Furthermore, SVD is only used to compute the nearest orthogonal matrix in the proposed system even though it is a powerful tool. Therefore, this paper proposes a hardware oriented iterative approach algorithm based on [28] that approximates the closest orthogonal matrix $\mathbf{M}_{COM}$ of $\mathbf{M}$ using

$$\mathbf{M}_{COM} = \mathbf{M}(\sqrt{\mathbf{M}^T\mathbf{M}})^{-1}. \tag{5}$$

The tricky part of using (5) is the square root of $\mathbf{M}^T\mathbf{M}$. Among all the possible solutions, non-closed-form methods such as Newton's iterative algorithm, Meini's algorithm [29], Denman-Beavers (D-B) algorithm, eigenvalue approach, and so on, appear to be easier for hardware implementation due to its simple use of matrix addition, multiplication, and inversion. Newton's method requires a good initial guess, and thus the solution is likely to diverge. Meini's and D-B algorithms are preferable choices, for they not only provide satisfactory convergence rate, the computations are also simple. Nevertheless, investigating both algorithms it is found that Meini's approach demands 4 matrix multiplications in each iteration, while D-B algorithm does not need any multiplication. Instead, only matrix additions are required. Considering integer computation used in the proposed VO system, matrix multiplication could indirectly enlarge integer arithmetic errors, not to mention that Meini's method is an iterative algorithm. Hence, this paper utilizes D-B method for the matrix square root, without the need of intricate floating point arithmetic in hardware. Consequently, the expression in (5) can be rewritten to

$$\mathbf{M}_{COM,p} = \mathbf{M}(\sqrt{\mathbf{M}_p^T\mathbf{M}_p})^{-1}, \tag{6}$$

where p is the iteration index of the D-B algorithm,

$$\mathbf{M}_{p+1}^T\mathbf{M}_{p+1} = \frac{1}{2}\left(\mathbf{M}_p^T\mathbf{M}_p + \mathbf{N}_p^{-1}\right) \tag{7}$$

$$\mathbf{N}_{p+1} = \frac{1}{2}\left(\mathbf{N}_p + (\mathbf{M}_p^T\mathbf{M}_p)^{-1}\right), \tag{8}$$

and when $p = 0$, $\mathbf{M}_p$ and $\mathbf{N}_p$ be the initial cross covariance matrix computed in the Cross CovMat module and the identity matrix, respectively.

However, [29] conducted numerical experiments and reported that Meini's method is the fastest and reliable method. It is true that DB method does not converge as fast as Meini's, thereby increasing the risk of accumulating integer arithmetic error as more iteration is used. Nonetheless, as mentioned earlier, implementing Meini's method on hardware could lose data during matrix computations. Henceforth, we still use D-B algorithm in this paper, but we additionally employ a Taylor series approach for approximating the nearest orthogonal matrix after $\mathbf{M}_{COM,p}$, with p be an arbitrary integer obtained in (5), is already nearly orthogonal in order to speed up the approximation procedure. This reformulates (6) as

$$\mathbf{M}_{COM,p+1-\lambda} = \lambda \mathbf{M}_{COM,p} + (1-\lambda)\,\mathbf{M}_{TS}, \qquad (9)$$

where λ is set to 1 initially such that the closest orthogonal matrix of $\mathbf{M}$ is computed by (6). When the orthogonality of $\mathbf{M}_{COM,p}$ with p be an arbitrary integer satisfies the following 1-norm residual criteria,

$$\left\|\mathbf{Z}_p\right\| = \left\|\mathbf{M}_{COM,p}^T \mathbf{M}_{COM,p} - \mathbf{I}\right\| < \delta, \qquad (10)$$

λ becomes 0, enforcing $\mathbf{M}_{COM,p}$ to be further approximated to $\mathbf{M}_{COM,p+1}$ by means of $\mathbf{M}_{TS}$ which is derived using the residual in (10) and a Taylor series approximation to the second order,

$$\begin{aligned} \mathbf{M}_{TS} &= \mathbf{M}_{COM,p}(\mathbf{M}_{COM,p}^T \mathbf{M}_{COM,p})^{-1/2} \\ &= \mathbf{M}_{COM,p}(\mathbf{I}+\mathbf{Z}_p)^{-1/2} \\ &\approx \mathbf{M}_{COM,p}\left(\mathbf{I} - \frac{1}{2}\mathbf{Z}_p + \frac{1}{8}\mathbf{Z}_p^T \mathbf{Z}_p\right). \qquad (11) \end{aligned}$$

In (11), when $\left\|\mathbf{Z}_p\right\| < 1$, the convergence of $\mathbf{M}_{TS}$ is locally uniform and normal; hence the value of δ must be set below 1. Using (9) with $\mathbf{M}_{COM,p}$ and $\mathbf{M}_{TS}$ defined respectively in (6) and (11), the hardware implemented closest orthogonal matrix solver can be easily achieved without the loss of data during integer arithmetic computation and the need of large on-chip memory capacity. Equations (6) and (11) are respectively done in the Denman-Beavers module as well as the Taylor module shown in Figure. 6. Note that since the translation matrix in an ICP algorithm is simple, it is implemented together in the Taylor module. Another notice is that the above-mentioned closest orthogonal matrix approach is different from our previous work [18] in that a variable λ is involved in, and the criteria of orthogonality given in (10) is proposed. Moreover, the hardware architecture is totally different as well.

After the camera pose is completely derived, the data is sent back to the SDRAMs via Buffer 45 to 32, DCFIFO#6, and Master Interface. Finally, Nios II is capable of retrieving data from the SDRAMs addresses of the pose.

VII. EXPERIMENTS

A. EXPERIMENTAL SETUP

To evaluate the proposed multiple master-slave HW/SW co-design FPGA design of a VO system, Altera DE2i-150 Cyclone IV FPGA board is employed. To compare the run-time of a VO with a general desktop PC, an i7 CPU of 4.2 GHz as well as a 4.00 GB RAM is involved in. Additionally, considering the operating speed, comparisons are made with various prior works including VO and VIO systems.

Regarding the proposed hardware implemented parallel image pyramid, parameters used in the genetic algorithm of the first and second steps of the optimization is provided in Table 1. Because the number of parameters used for optimization in the second step is greater than the first step, the required number of iterations of the second step problem is greater. Moreover, the terminal fitness value in the first and second steps are respectively the relative scale and sum of difference of Gaussian image intensities. Verifications of the optimization results are provided by means of peak signal-to-noise ratio (PSNR) between software and hardware, where images used in the optimization include the ones from the EuRoC dataset [30], KITTI dataset, TUM dataset, and [18]. The total number of images is 50, and the final optimal solution is taken from the average of all the optimality.

TABLE 1. Parameters used in the genetic algorithm.

Parameters	First Step	Second Step
Population	500	1000
Crossover Rate	0.4	0.3
Mutation Rate	0.1	0.1
Maximal Iterations	20	50
Terminal Fitness Value	1	75,000
Initial Parameter Values	$\sigma_{s,0} = 0.5$ k=1.5	$\sigma_{h,i} = \frac{10-\sigma_{s,0}+1}{4} i,\ i=1,\ldots,4$
Parameter Search Space	$\sigma_{s,0}$:[0.5,1.6] k :[1.2,2.5]	$\sigma_{h,1},\sigma_{h,2},\sigma_{h,3},\sigma_{h,4}$:[$\sigma_{s,0}$,10]

As for the robustness of the proposed system in terms of VO pose estimations, KITTI dataset is adopted due to its fruitful and challenging outdoor scenarios. The KITTI images are prior to be cropped into a resolution of 640×480. Additionally, due to limited storage of the Flash memory on the DE2i-150 FPGA board, the number of stereo images in one sequence loaded into the Flash memory is restricted to no more than 248 images. Therefore, the experiments conducted by the stereo VO in this paper performs only from the first image to the maximal 248th one. Metrics for the estimation accuracy include relative pose estimation (RPE) and absolution trajectory error (ATE). Two timestamps $\Delta t = 2$ and $\Delta t = 5$ of relative poses in RPE are investigated in the experiments. Also, to testify that the proposed hardware design achieves comparable results with respect to the PC software, estimation error between hardware and software is also concerned. Note that since we are unable to acknowledge what accuracy metrics were used in [10], [15] and [26], and the number of images is limited in our experiments, comparisons with prior works cannot unfortunately be made.

B. EXPERIMENTAL RESULTS AND ANALYSIS

1) HARDWARE IMPLEMENTED PARALLEL IMAGE PYRAMID

In the first step of the optimization problem, the best optimal solutions of $\sigma_{s,0}$ and k are respectively 0.7 and 1.258. Because the scale of the top level image in the parallel pyramid is fixed to 10, the base scale is not 1.6 as suggested by the original SIFT algorithm. However, the relative scales between levels are similar, as can be seen clearly from the scale curve shown in Figure. 7. To show that the base scale assigned as 1.6 is not a preferable case given $\sigma_{h,5} = 10$, Figure. 7 also plots the scale curve where $\sigma_{s,0} = 1.6$, where it does not hold high similarities with the original SIFT algorithm. Note that in Figure. 7, the scale of the top level image of the best solution does not exceed 1.5. It is recognized in Figure. 3 that the difference of intensities with respect to the original image when the scale is 1.5 is lower than the case when the scale is above 4. Therefore, approximating the scales of parallel image pyramid using software shown in Figure. 2 (a) imposes the reduced blurring effectiveness such that the condition of $\sigma_{h,5} = 10$ can be satisfied. Nevertheless, according to the author of the SIFT algorithm [3], as long as an image pyramid is built with proper relative scales, it suffices to reliable feature matching. Hence, the optimal solutions are used in the proposed hardware system.

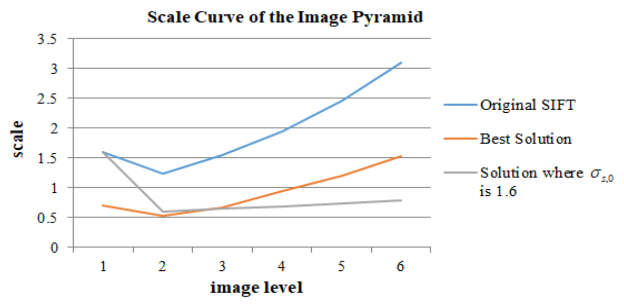

FIGURE 7. Scale curve of the image pyramid obtained in the first step of optimization.

As for the second step, the best scales of level 1 to level 4 of the hardware parallel image pyramid along with the PSNR value of each level of image is presented in Table 2, where $\sigma_{h,5} = 10$ and $\sigma_{h,0} = 0.7$ are obtained from the first step. The PSNR values shown in the table are all above 45, meaning that there is no significant difference of Gaussian image intensities provided by the software and hardware. Consequently, these scales are used for the hardware implemented parallel image pyramid.

2) RUNTIME EFFICIENCY

The efficiencies of the SIFT, LES, and ICP blocks in terms of fps compared to a desktop PC are shown in Table 3. The software PC implements SIFT, LES, and ICP from scratch without using packages such as OpenCV. Therefore, the

TABLE 2. Optimal scales of levels of parallel image pyramid and their PSNR valies.

	$\sigma_{h,0}$	$\sigma_{h,1}$	$\sigma_{h,2}$	$\sigma_{h,3}$	$\sigma_{h,4}$	$\sigma_{h,5}$
Optimal Values	0.7	0.86	1.1	1.46	2.12	10
PSNR (dB)	52.38	54.2	54.38	48.89	50	45.4

TABLE 3. Efficiency comparisons of the SIFT, LES, and ICP blocks compared to a desktop PC.

		Frequency (MHz)	FPGA runtime (fps)	PC runtime (fps)
SIFT block	SIFT Controller	96	147	0.026
	SIFT Top	100		
LES block	LES Controller	96	22.08	0.009
	LES Top	200		
ICP block	ICP Controller	96	208	1.67
	ICP Top	200		
Nios II processor		50	476	x
Overall System		x	33.2	0.0067

TABLE 4. Efficiency comparisons of the proposed system and other publications.

	No. of Tracked Features	Image Size	Platform	Time (fps)
[15]	1012 1190	512×384	Xilinx Virtex-6 + Intel CPU	1.31 1.77
[17]	128	N.A. (VGA)	Arria-10 FPGA + ARM Cortex-A9	30
[10]	200	752×480	Xilinx Kintex-7 XC7K355T + ARM Cortex-A15	20
[12]	N.A.	640×480	Zynq Ultrascale ZU9CG	22.4
		1280×720	Xilinx Virtex-7 XC7V690T + PC	31.9
[14]	35~150	752×480	Zynq-7000 + ARM Cortex-A9 CPU	20~142
[18]	300~400	640×480	Altera DE2i-150 built with a Nios II	6.25
Proposed Work	300~400	640×480	Altera DE2i-150 built with a Nios II	33.2

algorithm is fully operated in sequential manner. In the table, the runtime of the SIFT block is the scenario when 1,968 features are extracted. On the other hand, the LES block used for stereo and frame-to-frame matching performs 1968 to 2103 feature matching and 312 to 338 features, respectively. From the comparisons, it is clear that the proposed hardware based VO provides 43.2 fps, which is around a speedup of 4,955× compared to the PC.

Aside by comparing FPGA hardware with a software PC, Table 4 shows further efficiency comparisons between the proposed system and recent prior works including a HW/SW co-design feature based VO system [15] and various feature-based VIO systems including [10], [12], [14], and [17]. Although different dataset scenarios are used in

TABLE 5. Required hardware resources of the proposed VO system.

	LEs	Memory Bits	DSP Elements	DSP 9×9	DSP 18×18
SIFT Controller	56,655	294,912	0	0	0
SIFT Top	731	306,056	297	19	139
LES Controller	834	65,535	0	0	0
LES Top	3,936	0	0	0	0
ICP Controller	2,127	65,535	0	0	0
ICP Top	25,785	5,888	360	0	177
Nios II processor	5,061	2,583,552	0	0	0
Overall System	95,129	3,321,478	657	19	316
Capacity (%)	63.52	50	91.25	x	x

these methods, the image size is similar. Also, the number of tracked features among these approaches is also comparable, except [15] where huge amount of features is needed to be tracked due to little textures provided in the Mars rover environment. Since [15] proposed five different pipeline implementation, Table 4 gives the average outcome (1.31 fps) and the result of the least runtime required pipeline (1.77 fps). As for [14], its computational time depends on its running frequency, where the fastest case (142 fps) runs at its maximum frequency while the lowest one (20 fps) runs at the rate of which sensor data is captured. Given that a CCD camera compatible with the platform used in this paper runs at 133 MHz, we are able to firmly state that the proposed system can also be applied to online operation in 33.2 fps. Hence, consider real applications, the proposed method outperforms other existing works according to Table 4.

3) HARDWARE RESOURCE REQUIREMENTS

The on-chip resource usage of the hardware platform in this paper is listed in Table 5. The utilization of DSP elements comes from specific circuits loaded from Altera's packages, such as signed and unsigned dividers. From the table, it is clear that because of the off-chip memory that help storing feature data, the overall design is capable of fitting in one DE2i-150 board. Moreover, the amount of logic elements (LEs) takes around only 63.52% of the capacity. Therefore, the proposed design is not only highly reconfigurable, but also demands limited hardware resources, leaving substantial supports for future developments.

4) ROBUSTNESS OF THE PROPOSED VO SYSTEM

Using 11 sequences of the KITTI dataset, the proposed FPGA implemented VO provides accurate relative pose estimations, as can be seen from Table 6. Note that in our experiments only 248 images are performed due to limited Flash memory storage. The RPE metric demonstrates that the proposed FPGA-based VO is capable of reliably estimating the relative pose, yielding an average RPE error of 0.517 and 0.8 meters

TABLE 6. RPE and ATE of the proposed VO system.

Sequences	RPE (m) ($\Delta t = 2$)	RPE (m) ($\Delta t = 5$)	ATE (m)	Error w.r.t. PC
00	0.41	0.46	0.58	2.1%
01	1.73	1.21	1.911	4.3%
02	0.28	1.44	1.757	1.5%
03	0.14	0.84	1.791	2.2%
04	0.26	0.74	0.803	2.1%
05	0.94	1.32	1.08	3%
06	0.65	0.96	1.28	3.6%
07	0.23	0.27	0.51	2.8%
08	0.32	0.67	0.851	4.4%
09	0.11	0.21	0.47	1.6%
10	0.62	0.68	0.887	1.9%

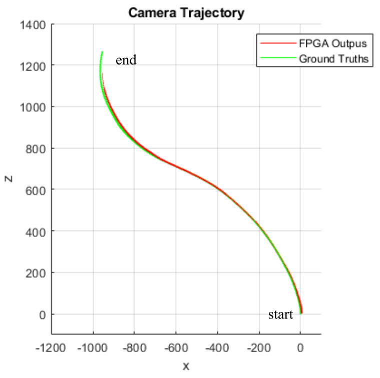

FIGURE 8. Trajectory comparisons of the sequence 09.

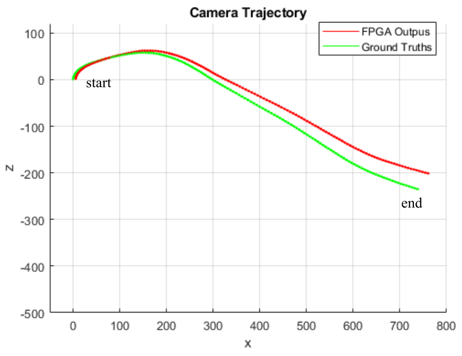

FIGURE 9. Trajectory comparisons of the sequence 10.

for all KITTI sequences when $\Delta t = 2$ and $\Delta t = 5$, respectively. ATE, in contrast, shows greater error because of the accumulated errors posted by the VO. Nevertheless, comparisons made with the PC software reveals that the proposed FPGA design still provides sufficient estimation accuracies, with an average of 2.7%.

As mentioned in Section II, as the ICP block finishes computing camera poses, the output data is packed in a compressed file by Nios II and is transferred to the host computer for display. Specifically, we use MATLAB to plot the trajectories. The following figures show selected estimation outcomes of sequence 09 and 10, where green and red trajectories respectively represent the ground truths and the FPGA outputs. In sequence 09, there is no significant shift from the ground truths to the estimations, yet the errors are accumulated clearly at the end of the trajectory. As for the sequence 10, the estimation drifts are gradually significant as the camera explores the environment. Nonetheless, according to Table 6, in terms of ATE, an average of 1.084 meters is given by the proposed hardware design.

VIII. CONCLUSION AND DISCUSSIONS

In this paper, a multiple master-slave HW/SW co-design architecture of a VO is proposed. The master-slave structure allows the overall system to be highly reconfigurable and easily managed. The use of a Nios II processor built in the hard core organizes the pipeline computation of hardware modules only, whereas primary computations are made in hardware. To determine scales of Gaussian images for hardware design of parallel image pyramid, genetic evolutionary algorithm is employed to optimize both the relative scale between levels with respect to the original SIFT algorithm, and the difference of intensities obtained by software and hardware. LES block for stereo and frame-to-frame correspondences is proposed to give robust and efficient local feature matching. Pose estimations are made by the 3D-3D points ICP algorithm, where a hardware design of a nearest orthogonal matrix algorithm is proposed to prevent the reliance on SVD. This saves lots of memory storage while simultaneously providing accurate integer arithmetic computation. Experiments show that 33.2 fps of runtime efficiency can be achieved without the need of huge hardware resources requirements. Also, using the KITTI dataset, the proposed VO system is capable of providing 0.66 and 1.084 meters of estimation errors in terms of RPE and ATE, respectively.

There are several challenges that remain open for future works. These include adding a data valid signal to the SIFT Top module such that the memory capacity of DCFIFO#1 shown in Figure 4 can be reduced. In addition, a hardware implemented RANSAC algorithm used to remove false feature correspondences in the LES Top module can help improve the ICP estimation accuracies. Furthermore, using a real stereo camera on the FPGA board allows the proposed system to perform in real environments. As demonstrated in our experiments, this also prevents the problem of limited storage of number of gray images in the SDRAM. Therefore, in our future works, it is worthwhile to further strengthen the proposed system by reducing the usage of hardware resources of the SIFT Top module, excluding outliers of feature matching in the LES Top module, and design an stereo camera input circuit for capturing images from the real environments.

REFERENCES

[1] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 105–119, Jan. 2010.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," *Comput. Vis. Image Understand.*, vol. 110, no. 3, pp. 309–432, Jun. 2008.

[3] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[4] C. Tomasi and T. Kanade, "Detection and tracking of point features," Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-CS-91132, 1991.

[5] M. Muja and D. G. Lowe, "Fast matching of binary features," in *Proc. 9th Conf. Comput. Robot Vis.*, May 2012, pp. 404–410.

[6] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[7] A. HajiRassouliha, A.-H. Taberner, M.-P. Nash, and P.-N.-F. Nielsen, "Suitability of recent hardware accelerators (DSPs, FPGAs, and GPUs) for computer vision and image processing algorithms," *Signal Process., Image Commun.*, vol. 68, pp. 101–109, 2018.

[8] C.-H. Lin, W.-Y. Wang, S.-H. Liu, C.-C. Hsu, and C.-H. Chien, "Heterogeneous implementation of a novel indirect visual odometry system," *IEEE Access*, vol. 7, pp. 34631–34644, 2019.

[9] B. Yu, J. Tang, and S. Liu, "On designing computing systems for autonomous vehicles: A PerceptIn case study," 2020, *arXiv:2011.06277*. [Online]. Available: http://arxiv.org/abs/2011.06277

[10] Z. Zhang, A. Suleiman, L. Carlone, V. Sze, and S. Karaman, "Visual-inertial odometry on chip: An algorithm-and-hardware co-design approach," in *Proc. Robot., Sci. Syst. XIII*, Cambridge, MA, USA, 2017.

[11] Z. Wan, B. Yu, T. Yuang Li, J. Tang, Y. Zhu, Y. Wang, A. Raychowdhury, and S. Liu, "A survey of FPGA-based robotic computing," 2020, *arXiv:2009.06034*. [Online]. Available: http://arxiv.org/abs/2009.06034

[12] Y. Gan, B. Yu, B. Tian, L. Xu, W. Hu, S. Liu, Q. Liu, Y. Zhang, J. Tang, and Y. Zhu, "Eudoxus: Characterizing and accelerating localization in autonomous machines," 2020, *arXiv:2012.01353*. [Online]. Available: http://arxiv.org/abs/2012.01353

[13] H. Strasdat, J. M. M. Montiel, and A. J. Davison, "Visual SLAM: Why filter?" *Image Vis. Comput.*, vol. 30, no. 2, pp. 65–77, 2012.

[14] A. Suleiman, Z. Zhang, L. Carlone, S. Karaman, and V. Sze, "Navion: A 2-mW fully integrated real-time visual-inertial odometry accelerator for autonomous navigation of nano drones," *IEEE J. Solid-State Circuits*, vol. 54, no. 4, pp. 1106–1119, Apr. 2019.

[15] G. Lentaris, I. Stamoulias, D. Soudris, and M. Lourakis, "HW/SW codesign and FPGA acceleration of visual odometry algorithms for rover navigation on mars," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 8, pp. 1563–1577, Aug. 2016.

[16] B. Liu, L. Li, and H. Liu, "SoC implementation of visual-inertial odometry for low-cost ground robots," *J. Phys., Conf. Ser.*, vol. 1453, Jan. 2020, Art. no. 012091.

[17] D. K. Mandal, S. Jandhyala, O. J. Omer, G. S. Kalsi, B. George, G. Neela, S. K. Rethinagiri, S. Subramoney, L. Hacking, J. Radford, E. Jones, B. Kuttanna, and H. Wang, "Visual inertial odometry at the edge: A hardware-software co-design approach for ultra-low latency and power," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2019, pp. 960–963.

[18] C.-H. Chien, C.-J. Chien, and C.-C. Hsu, "HW/SW co-design and FPGA acceleration of a feature-based visual odometry," in *Proc. 4th Int. Conf. Robot. Autom. Eng. (ICRAE)*, Nov. 2019, pp. 148–152.

[19] S.-A. Li, W.-Y. Wang, W.-Z. Pan, C.-C. J. Hsu, and C.-K. Lu, "FPGA-based hardware design for scale-invariant feature transform," *IEEE Access*, vol. 6, pp. 43850–43864, 2018.

[20] R. Datta and K. Deb, *Evolutionary Constained Optimization*. India: Springer, 2014.

[21] C.-H. Chien, C.-J. Chien, and C.-C. Hsu, "Hardware-software co-design of an image feature extraction and matching algorithm," in *Proc. 2nd Int. Conf. Intell. Auto. Syst. (ICoIAS)*, Feb. 2019, pp. 37–41.

[22] A. Nüchter, K. Lingemann, J. Hertzberg, and H. Surmann, "6D SLAM—3D mapping outdoor environments," *J. Field Robot.*, vol. 24, nos. 8–9, pp. 699–722, 2007.

[23] E. Denman and N. Beavers, "The matrix sign function and computations in systems," *Appl. Math. Comput.*, Vol. 2, pp. 63–94, Jan. 1976.

[24] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361.

[25] J. Sturm, W. Burgard, and D. Cremers, "Evaluating egomotion and structure-from-motion approaches using the TUM RGB-D benchmark," in *Proc. Workshop Color-Depth Camera Fusion Robot. IEEE/RJS Int. Conf. Intell. Robot Syst. (IROS)*, Oct. 2012, pp. 1–7.

[26] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 573–580.
[27] M. U. Torun, O. Yilmaz, and A. N. Akansu, "FPGA, GPU, and CPU implementations of Jacobi algorithm for eigenanalysis," *J. Parallel Distrib. Comput.*, vol. 96, pp. 172–180, Oct. 2016.
[28] N. J. Higham, "Computing the polar decomposition—With applications," *SIAM J. Sci. Stat. Comput.*, vol. 7, no. 4, pp. 1160–1174, Oct. 1986.
[29] B. Meini, "The matrix square root from a new functional perspective: Theoretical results and computational issues," Univ. Pisa, Pisa, Italy, Tech. Rep. 1455, 2003.
[30] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, W. Markus, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *Int. J. Robot. Res.*, vol. 35, no. 10, pp. 1157–1163, 2016.

CHIANG-HENG CHIEN received the B.S. and M.S. degrees in electrical engineering from the National Taiwan Normal University, Taipei, Taiwan, in 2015 and 2017, respectively. He is currently pursuing the Ph.D. degree with Brown University, Providence, RI, USA, working on differential geometry in computer vision applications and homotopy continuation on multiple-view pose estimation problems. He has been studying robot localization, evolutionary computation, SLAM, and FPGA design for more than six years. His research interests include multiple-view pose estimations, robot localization, visual SLAM algorithms, and FPGA hardware design.

CHEN-CHIEN JAMES HSU (Senior Member, IEEE) was born in Hsinchu, Taiwan. He received the B.S. degree in electronic engineering from the National Taiwan University of Science and Technology, Taipei, Taiwan, in 1987, the M.S. degree in control engineering from the National Chiao-Tung University, Hsinchu, in 1989, and the Ph.D. degree from the School of Microelectronic Engineering, Griffith University, Brisbane, QLD, Australia, in 1997.

He was a Systems Engineering with IBM Corporation, Taipei, for a period of three years, where he was responsible for information systems planning and application development, before commencing his Ph.D. studies. He joined the Department of Electronic Engineering, St. Johns University, Taipei, as an Assistant Professor, in 1997, and was appointed as an Associate Professor, in 2004. From 2006 to 2009, he was with the Department of Electrical Engineering, Tamkang University, Taipei. He is currently a Professor with the Department of Electrical Engineering, National Taiwan Normal University, Taipei. He is the author or coauthor of more than 200 refereed journal articles and conference papers. His current research interests include digital control systems, sensor applications, and the mobile robot navigation. He is an IET Fellow.

CHIANG-JU CHIEN (Member, IEEE) received the Ph.D. degree in electrical engineering from the National Taiwan University, Taipei, Taiwan, in 1992. Since 1993, he has been with the Department of Electronic Engineering, Huafan University, New Taipei City, Taiwan, where he is currently a Professor. He worked as the Chair of the Department of Electronic Engineering, Huafan University, from 2000 to 2005, the Secretary General, from 2005 to 2011, and the Dean of the College of Engineering and Management, Huafan University, from 2012 to 2015. Since 2015, he has been the Secretary General with Huafan University. His current research interests include iterative learning control, adaptive control, fuzzy-neuro systems, and control circuit design. He is a member of the Chinese Automatic Control Society.