Wearable, Energy-Opportunistic Vision Sensing for Walking Speed Estimation

Andres Gomez*, Lukas Sigrist*, Thomas Schalch*, Luca Benini *†, Lothar Thiele*
*D-ITET, ETH Zurich, 8092 Zurich, Switzerland
Email: {gomeza, sigristl, schalcht, lbenini, thiele}@ethz.ch
†DEIS, University of Bologna, Bologna, Italy
Email: firstname.lastname@unibo.it

***Abstract*—State-of-the-art wearable systems are typically performance-constrained, battery-based devices which can, at most, reach self-sustainability using energy harvesting and aggressive duty-cycling. In this work, we present a wearable vision sensor node which can reliably execute computationally-intensive computer-vision algorithms in an energy-opportunistic fashion. By leveraging a burst-generation scheme, the proposed system can efficiently provide the energy guarantees required for tasks with temporal dependencies, even under highly variable harvesting conditions. By mounting the node on a user's glasses, the node is able to acquire a sequence of images and determine the user's walking speed, requiring only a small solar panel and capacitor. Both hardware and software have been fully optimized for ultra-low power consumption and high performance. Extensive experimental results show the energy node's energy proportionality and the accuracy of its walking speed estimation.**

I. Introduction

Over the past decade, there has been a substantial research effort to reduce the power consumption of electronic devices. Many techniques have been developed to reduce the average power consumption of battery-powered devices in an attempt to prolong their lifetimes. Energy harvesting, though a mature technology, has only been successfully deployed in large scale systems where size is not a limiting factor. In the wearable domain, however, a source's physical dimensions are just as important, if not more, than its harvesting capacity. Furthermore, wearable sources such as kinetic, piezoelectric and photovoltaic are subject to very rapidly changing environmental conditions and harvesting rates, making efficient operation a veritable challenge [3]. Traditional approaches that use large storage elements are not feasible in this domain because they are expensive, have short lifetimes and cannot easily scale with power hungry applications such as computer vision. Hence, innovative energy harvesting concepts are needed to power the increasing number in the Internet of Things (IoT) and wearable devices in an efficient, low-cost, long-term, self-sustainable manner.

Transiently powered systems operate efficiently in adverse harvesting conditions, requiring only limited storage capacity and input power to reliably execute power-hungry applications. The recently-proposed Energy Management Unit (EMU) [8] allows a transient system to operate in an energy-proportional manner. As opposed to traditional duty-cycling, an EMU-based system is energy driven, meaning that as the available energy increases, so does application's execution rate. This would allow devices to operate reliably and efficiently in the wide input power range found in typical wearable sources. Furthermore, they do not require expensive and bulky storage devices that would render them unwearable.

Vision sensors, which acquire and process images, are typically power-hungry devices which require substantial computational resources. For this reason it was not until recently that it became feasible to have batteryless vision sensors in a wearable form factor, thanks to new paradigms in energy harvesting systems. Wearable vision sensors are a nascient field, because compared to traditional low power sensors like accelerometers, vision sensors offer much more information which could be used to accurately determine not only the step count, but other variables as well. In [13], for example, authors partly use vision data to determine the user's context.

Vision sensors have been studied for many years, and there has been a significant research effort to use them for optical navigation of unmanned aerial vehicles (UAV) or robots. V. More et al. proposed a visual navigation system for UAVs based on optical flow estimation using the Lucas-Kanade method and an ultrasonic sensor [15]. To account for the high computational effort for the Lucas-Kanade method, the system runs on a on-board linux computer (Odroid-U3), which consumes up to 10 W. Computing the optical flow with the Lucas-Kanade method can also be done on computationally constrained microcontrollers such as the Atmel ATmega2560. K. Schneider et al. [17] took this approach and used the same Stonyman vision chip from Centeye as in this work. However, low power microcontrollers have very limited memories and in their case it could only handle camera resolutions up to 28×28 pixels due to the memory intensive Lucas-Kanade method.

In general, vision sensors either require too much power or offer too little performance to design functional batteryless devices. In [16], for example, a stereo vision system for a micro aerial vehicle capable of processing 640x480 frames at 60 fps, requiring 5 W and 50 grams for the FPGA subsystem alone. In [9], the authors use a PX4FLOW optical flow sensor to provide velocity and position estimation at high update rates for mobile robot navigation. The system is not only vision-based, but also includes a gyroscope and an ultrasonic sensor. The power consumption of this system is specified as 575 mW. To design a batteryless system with these power requirements would require a solar panel with an area of 100's cm^2, which is no longer wearable.

In this work, we describe the design and implementation of an energy-opportunistic, wearable vision sensor node capable of executing computationally intensive tasks with temporal

dependencies. Thanks to its EMU-based design, it can reliably and efficiently acquire and process images to estimate the user's walking speed in a wide variety of harvesting scenarios. Our proposed vision sensor has an average active power consumption of 6.85 mW, but requires only 100's of μW's to begin making velocity estimations. Furthermore, it can reach up to 5.8 velocity estimations per second, and has a motion estimation error of 1.4 % of the distance traveled.

II. Preliminaries

Transiently powered systems are energy harvesting-based systems which can operate reliably with very limited energy storage and very adverse harvesting conditions. They are also called batteryless because their energy storage is required to be as small as possible and their behavior is entirely energy-driven. How big the storage element is depends entirely on the application and its energy needs, but as opposed to battery-based designs, the system cannot buffer the energy required for 100's or even 1000's of iterations. The challenges to build such systems are many, especially given that volatile energy sources might never be able to directly power the system. In the recently proposed energy harvesting scheme called Dynamic Energy Burst Scaling (DEBS), it is argued that for these systems to operate reliably and efficiently, they have to accumulate harvested energy until enough is available for the execution of one single atomic task, also called a burst. Afterwards, the system must be shut down completely until enough energy is accumulated for the next burst execution. The time interval between two bursts depends on the currently available input power. This type of operation directly leads to three challenges for the design of transiently powered systems:

Constraint (1) ***Minimum Energy Guarantee****:* The energy harvester cannot directly power the system. To guarantee the execution of atomic tasks, the storage device should provide this *minimum energy availability*.

Constraint (2) ***Temporal Independence****:* There is no control over the length of the time interval between two bursts, since this only depends on the currently available input power. The application needs therefore to be split into *separate bursts with no temporal dependencies*.

Constraint (3) ***Non-Volatility****:* Between two bursts, the system is completely shut down and all peripherals are powered off. Therefore, if an application cycle is split among different bursts, there is a need for using *non-volatile memory* (NVM) technologies to retain the system's state between bursts. Even if an application cycle fits in a single burst, long-term logging requires NVM.

To the best of our knowledge, these constraints have prevented system designers from building reliable harvesting-based wearable image vision sensors. In our walking speed estimation scenario, we overcome these restrictions with DEBS, and we can obtain accurate, high performance vision sensing by leveraging existing motion estimation algorithms. The rest of this section gives a brief overview of the two areas that are combined in our proposed system.

A. Dynamic Energy Burst Scaling (DEBS)

Typical cyber-physical systems consist of a microcontroller with its peripherals like memories, sensors or transceivers. Microcontrollers can usually operate in a wide voltage range, but they are most efficient at minimum supply voltages. The peripherals can have completely different voltage requirements. Often, the highest minimum voltage is then taken for the whole application design to avoid converter losses for multiple voltage domains. If the application is divided into several tasks in which only some of the peripherals are used, each task has its own optimal supply voltage for lowest energy consumption. The idea of *Dynamic Energy Burst Scaling (DEBS)* is to supply each task with its optimal supply voltage, or in other words to track the load's optimal power point in order to minimize its energy consumption [8].

The advantage of DEBS is that the power point of the load and the source are completely decoupled. This allows harvesting the maximal possible power from the energy harvesting source, whereas each task is supplied with the lowest possible supply voltage to minimize the energy consumption for each task.

B. Motion Estimation

Optical flow is a fundamental concept in visual perception firstly described by Gibson in 1950 [7]. The optical flow describes the apparent velocities of movements of brightness patterns in the perceived image [10]. Those movements of brightness patterns may be caused by relative motion between the captured objects and the observer or by motion of single objects within the scenery. Expressing the optical flow in an image sequence in an appropriate mathematical way can be a very useful tool for many applications like flow measurements of fluids [2], motion segmentation [6] or video compression [1]. In this work, the concept of optical flow is used to extract information about the movement of the observer while capturing a static scenery.

III. Transient Vision Sensor

We propose a transiently powered vision sensor to be attached to a user's glasses to estimate and log the walking speed of the person. Figure 1 shows the generic function of a visual velocity estimation sensor node, which can be divided into three parts: *Image Acquisition*, *Processing* and *Storage*.

1) Image Acquisition and Compensation: If the vision sensor is attached to someone's glasses, a camera needs to face down onto the floor to capture the area in front of the person. According to *Constraint (1)*, it is not possible to acquire a continuous sequence of images, since the system is not continuously powered. But the smallest number of images for being able to recover a displacement is two. This means that at the very least, each burst needs to guarantee at least the energy required to acquire and compensate two pictures in sequence. Therefore, two images with a well-known, constant time difference need to be acquired within the same burst to meet the temporal condition formulated in *Constraint (2)*.

2) Processing: Motion Estimation: Using the two acquired image frames, a velocity estimation needs to be calculated. This is done using a motion estimation algorithm which will be discussed below. The result of the motion estimation algorithm is a so-called optical flow field, a vector field in which every vector indicates the displacement of the corresponding part of the image between the two frames. This optical flow field is then reduced to one single displacement vector indicating the

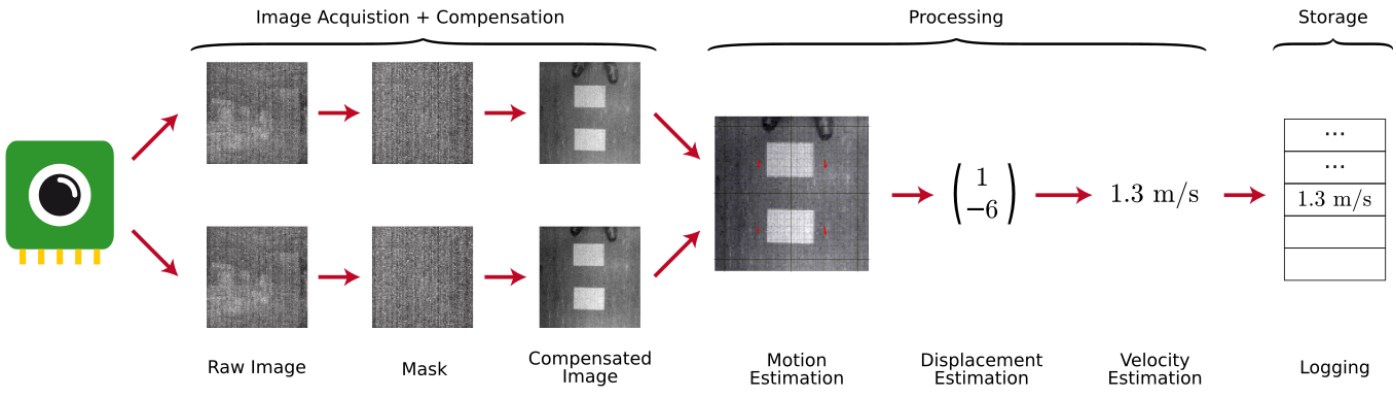

Fig. 1: Overview of the proposed energy-opportunistic walking speed estimation sensor.

displacement between the two frames. Together with the user's height, camera view angle and image acquistion times, this displacement can be scaled to a final velocity estimation.

In this work, we implement a block-based algorithm. This heuristic estimates the motion between two images by comparing them for all possible displacements and determining where the images "fit best". Such a method only works, if there is a pure translation and only a negligibly small rotation between the images. To relax this condition, one could think of dividing the image into blocks and then search for each block the position in the next image where the block "fits best". The smaller the block size, the more robust the method is to rotations. The result of the algorithm is an independent translation vector for each block, which represents an estimate for the optical flow of the pixels within the considered block. Block-based methods are not restricted to small displacements like the differential methods and are therefore able to recognize displacements of many pixels.

3) Storage: The final velocity estimation will be immediately saved to the external FRAM memory, following *Constraint (3)*. Though rich data sensors produce large volumes of data, due to the limited power budget and available FRAM, we will only save the processing results. This reduces the storage requirements from 200 Kbits for two image acquisitions to only 2 Bytes to hold the displacement vector. This means that a 4 Mbit FRAM memory, which at the time of writing is the largest commercially available, would last 30 days when performing one velocity estimation every 10 seconds.

IV. System Architecture

Figure 2 shows an overview of the hardware implementation of the wearable vision sensor. The hardware can be divided into two parts according to their functionalities. First, an Energy Management Unit (EMU) is responsible for energy harvesting and power management. The EMU accumulates the harvested energy from a solar panel in an energy buffer and controls the energy burst generation for the transiently powered vision sensor. The second part is an Image Processing Unit (IPU), which can acquire images, process the captured images and store data in Non-Volatile Memory.

A. Image Processing Unit (IPU)

Microcontroller: Due to the computationally intensive nature of the velocity estimation and low power requirements, the Cortex M4F family of processors was selected. In this work, we use the *MSP432P401R* (MSP432) [11]. The MSP432 can be clocked up to 48 MHz and is currently the lowest powered MCU featuring the ARM Cortex-M4F architecture [12].

Centeye Stonyman Vision Chip: The Stonyman vision chip from Centeye [4] was chosen as the image sensor. Its ultra-low power consumption (around 2mW at 3V supply voltage [18]) makes it unique amongst comparable commercially available image sensors and makes it a popular choice for ultra-low-power vision sensing projects. Examples using the Stonyman vision chip are iShadow (a wearable, real-time mobile gaze tracker [14]), KinetiSee (a wearable camera acquisition system with a kinetic harvester [18]) or a vision-based space landing control for a miniature tailed robot [19].

FRAM Chip: FRAM chips are ideal for non-volatile memory applications that require frequent and fast read/write operations of small amounts of data. The FRAM chip FM25V10 with 1 MBits storage capacity is used in this project [5]. The memory chip can be accessed over SPI with clock frequencies of up to 40 MHz.

B. Energy Management Unit (EMU)

The Energy Management Unit (EMU), first proposed in [8], harvests energy from a source (e.g. a solar panel), stores it in an energy buffer and controls the energy burst generation for the vision sensor. The heart of the EMU is a BQ25505

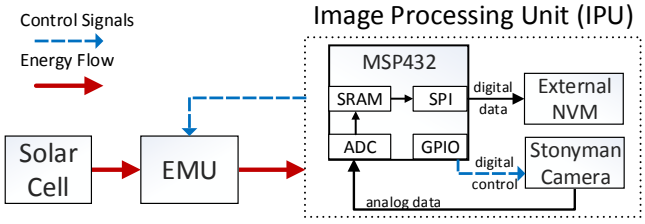

Fig. 2: Architecture of the wearable vision sensor.

energy harvesting chip, which features a built-in boost converter that can accumulate the harvested energy in a capacitor independently from the source's voltage. The input impedance of the chip is continuously adjusted to track the source's optimal power point. The load is supplied with the desired voltage by a buck converter (TPS62740) with digitally adjustable output voltage. Hence, the power points of the source and the load are completely decoupled.

The EMU is designed for use in transiently powered systems. The energy burst generation is implemented as follows. The load (in this case the IPU) needs to configure the size of the next energy burst, as well as the current optimal output voltage level. The output voltage level can directly be adjusted digitally by the load using the interface at the buck converter. Between two energy bursts, the output voltage is kept at the minimal operating voltage of the load MCU in deep sleep. If the MCU is woken up to execute the next task, the minimal voltage level for the current task is set by the MCU. The EMU needs to know the size of the next energy burst for triggering the wake-up of the load MCU and therefore the execution of the next task. For that, the EMU uses a *power_OK* signal to trigger the load. It changes to high as soon as a configurable voltage threshold at the energy buffer is reached. The load MCU can configure this voltage threshold according to the requested size of the next energy burst. The *power_OK* signal of the energy harvesting chip is then used to wake up the load MCU by a GPIO interrupt. To power our vision node, the capacitor's voltage level V_{cap} must stay within $V_{min} = 3.0\,\text{V}$ and $V_{max} = 5.1\,\text{V}$, which are the limits imposed by the harvesting chip and the buck converter. The amount of energy stored between V_{min} and V_{max}, which guarantees a burst's energy, depends on the capacitance as follows:

$$E_{burst} = \frac{1}{2}C(V_{max} - V_{min})^2 \tag{1}$$

Using this formula, the size of the energy buffer capacitor was determined to be 150 μF such that it can guarantee an energy burst of 1.3 mJ. This total energy corresponds to one complete velocity estimation, including all of the tasks characterized in Sec. V-A.

C. Firmware

1) Image Acquisition: The MSP432 uses PWM timers to generate specific control signals optimized for ultra-low-power acquisition. The ADC runs in parallel and continuously samples and digitizes the analog output signal for each pixel. The CPU is used to copy the digitized pixel values from the ADC memory to SRAM, using an interrupt that is triggered every time an ADC cycle is completed. An ADC clock frequency of 8 MHz and a CPU frequency of 24 MHz are needed to achieve a frame rate of 37.5 fps. The internal reference voltage V_{ref} is set to the lowest possible value of 1.2V and an ADC resolution of 10 bits is used. Because the voltage level of the analog output signal is between 0V and 0.3V for indoor light conditions, the two most significant bits of the sampled pixel values can be truncated. This leads to a 8-bit resolution of the pixel values between 0V and 0.3V and has the advantage that every pixel value can be stored in an unsigned integer variable with a length of 8 bits.

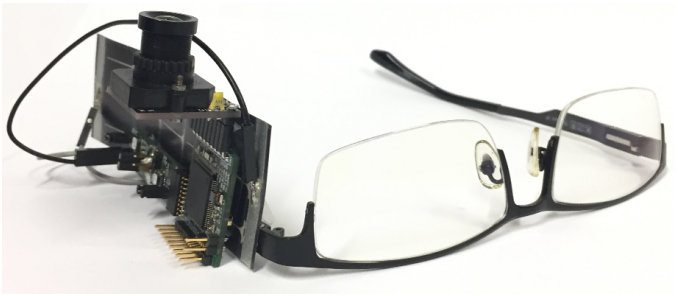

Fig. 3: Image of the wearable, batteryless vision sensor prototype with a flexible solar panel mounted on conventional glasses.

FPN Compensation on the MSP432: Illuminating the whole vision chip with an uniform intensity should theoretically produce the same analog output voltage for all pixels. Due to process variations, each pixel exhibits an individual, constant offset in the output voltage, also called Fixed Pattern Noise (FPN) [4]. In our particular case, the FPN is constant, which means a mask I_{mask} needs to be characterized only once and can then be stored for future use. The FPN can be removed from a raw image I_{raw} after the acquisition, by subtracting I_{raw} from the mask I_{mask}. While this could possibly be done during acquisition, it will be shown empirically in Sec. V-A that doing so will consume more energy than when doing it sequentially.

2) Power Management: The vision sensor shall be powered transiently, driven by energy bursts from the EMU. Between two bursts, all peripherals must be shut down and the microcontroller must go into the deepest sleep mode to minimize the losses between energy bursts. The Stonyman vision chip supports a software shut-down and can be connected directly to the supply voltage. The FRAM chip is connected to the supply voltage via a load switch (TPS22960) controlled by GPIO pins of the MSP432. This allows the MCU to activate the memory only when needed.

The MSP432 is optimized for low-power applications and features various low-power modes (LPMs) [12]. In the deepest low-power mode (LPM4.5), all peripherals including the CPU and the SRAM bank are disabled. During deep sleep, the GPIOs can be configured to lock their previously configured state which will be kept as long as there is power. This important feature guarantees a consistent interface between EMU and IPU also during deep sleep and further allows waking up the IPU by toggling a appropriately configured GPIO pin besides the usual the reset pin. The current consumption in LPM4.5 is only 25 nA [11]. Therefore, this LPM is ideal for the use in a transiently powered system. Because also the volatile internal SRAM banks are disabled in LPM4.5, the system state has to be restored after wakeup. In our implementation, the system state is stored in the FRAM chip. The state basically consists of a simple finite state machine to keep track of the next available FRAM memory location.

In order to wake up the MSP432 from LPM4.5, the EMU toggles a *power_OK* signal, causing a GPIO interrupt on the MCU. Four digital lines are used for configuring the buck converter in the EMU. Voltages from 1.8 V to 3.3 V in steps

of 100 mV can be requested. Two additional digital lines are used to configure the next burst size. The EMU uses this information for setting the energy threshold for sending the *power_OK* signal. All these six digital lines are driven by GPIO pins of the MSP432.

V. Experimental Evaluation

In this section, we will present the experimental evaluation of our batteryless, wearable vision sensor prototype. The evaluation is organized as follows: first, the energy consumption and the execution time of all tasks are characterized. Those measurements are used to optimize the tasks for lowest energy consumption and for configuring the Energy Management Unit. The energy proportionality and efficiency is tested under low power harvesting scenarios. Next, the accuracy of the block-based motion estimation algorithm is also evaluated. Finally, the velocity estimation sensor is tested in a real-world scenario.

A. Task Characterization

Image Acquistion: The image acquisition task acquires two images in a sequence and compensates for the fixed pattern noise (FPN). For minimizing the computational effort of the motion estimation algorithm, the time difference between the two acquired images should be minimal. Therefore, the vision sensor's maximal frame rate of 37.5 frames per second was chosen for this application. To achieve this frame rate, the CPU frequency has to be at least 24 MHz. Using a CPU frequency of 48 MHz even allows to compensate for the FPN on-line during acquisition, while still achieving the same frame rate. The time and energy characterization can be see in Table I. The evaluation reveals that configuration with the lower CPU frequency is beneficial in a transiently powered system, since the overall energy consumption is 42% lower.

TABLE I: Energy consumption and execution time of the image acquisition task with compensation.

FPN compensation	CPU Freq.	Executions	$E_{task,avg}$	$t_{task,avg}$
on-line	48 MHz	1152	930 μJ	**53 ms**
sequential	24 MHz	1829	**537 μJ**	61 ms

Image Processing: The processing task estimates the displacement between the two previously captured images by applying the block-matching algorithm to get an optical flow field, which is then further processed to estimate the final displacement value. For the task characterization, the block size of the block-matching algorithm is set to 48px and the search area to ± 3 and ± 8 in x- and y-direction respectively. The power supply voltage should be chosen as low as possible, which is 2.2 V in this case, since this is the minimal supply voltage for the MSP432 when using high clock frequencies. The only remaining variable parameter for the processing task is thus the CPU frequency. Table II compares the energy consumption and the execution time of the processing task for different CPU frequencies. As expected, higher CPU frequencies lead to shorter execution times. However, shorter execution times do not necessarily result in lower energy consumption, because the MSP432 MCU consumes much more power for higher clock frequencies [11]. As the results in Table II show, the energy consumption of the processing task is lowest at a CPU frequency of 24 MHz.

TABLE II: Energy consumption, execution time and power consumption of the processing task.

CPU Freq.	Executions	E_{task} (mean)	t_{task} (mean)	P_{task} (mean)
48 MHz	1079	757 μJ	**56 ms**	13.5 mW
24 MHz	1254	**635 μJ**	110 ms	5.8 mW
12 MHz	770	1371 μJ	388 ms	3.5 mW

Storage Task: For the storage task, the energy consumption of saving the motion estimation results to FRAM. Since the output of the motion estimation is only 2 B per iteration, this means that it takes 16 μs and 96 nJ to save one result.

B. Energy Efficiency and Proportionality

In order to evaluate the vision sensor's energy efficiency and proportionality, it is operated as a transiently powered system, using the Energy Management Unit (EMU) and the Image Processing Unit (IPU). The EMU harvests energy from a solar panel (flexible MP3-37 solar panel from PowerFilm with an area of 42 cm^2) that is exposed to constant illumination by a lamp. The experiment runs several times for different illumination levels. Each experiment lasts between 300 s (for high input powers) and 900 s (for low input powers) to ensure capturing a representative number of burst executions. The energy efficiency, calculated by measuring the energy harvested by the EMU and consumed by the vision sensor, can bee seen in Fig. 4. The same plot also shows the average number of estimations per minute which were recored during each experiment.

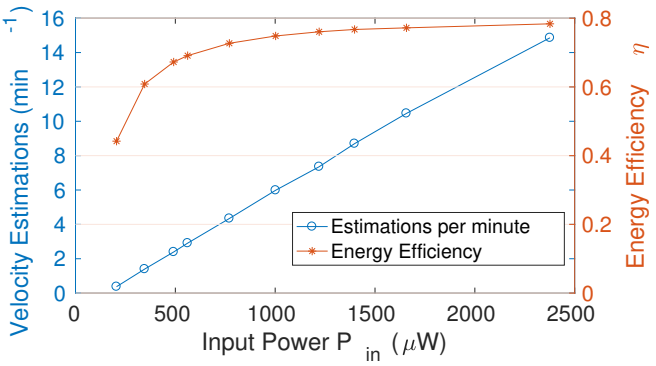

Fig. 4: Measured energy efficiency and execution rate as a function of the input power.

As expected, both velocity estimations per minute and the system's energy efficiency increase for higher input power levels. The number of velocity estimations is proportional to the input power, and the slope is determined by the average energy needed for one velocity estimation. The product of the efficiencies of the boost and the buck converters on the EMU limits the maximally achievable efficiency, measured to reach up to 78.7%. For input power levels above 600 μW, the efficiency is higher than 70%. The lowest input power level in the experiment is 198 μW and achieves an efficiency of 43% when suppling the vision sensor with an average power consumption of 6.85 mW.

C. Motion Estimation Accuracy

To find an optimal value for the block size, a sample image data set was first acquired from a development prototype with an SD card to continuously record images of a controlled displacement. These images were then post-processed to compare the estimated and the measured displacements. A Matlab model of the block-matching algorithm tested different block sizes between 96 pixels (using the whole image as one single block) and 16 pixels (dividing the whole image into 36 blocks). The size of the search area was set to $\pm$8px in x- and y-direction. A block size of 48px showed the best combination of sensitivity against rotations and high x-direction accuracy. After a traveled distance of 42 m, its estimated position deviates only 57 cm from the reference value, which corresponds to only 1.4% of the traveled distance. Though this accuracy does depend on the type of floor surface, we assume it also holds for our real-world experiment, which has an equivalent experimental set-up.

D. Real-World Velocity Estimation

In order to demonstrate the velocity estimation in a typical wearable scenario, the wearable vision sensor prototype was attached to the glasses (see Fig. 3), such that the Stonyman vision chip faces downwards and captures the floor in front of the person. For this experiment, the user tested a stand-walk pattern for a few seconds each and with different walking speeds. The prototype was continuously powered to acquire as much data as possible.

The vision sensor executes a loop of acquiring two images, estimating the displacement between two images and writing the images and the results onto the NVM. Because changing the clock frequencies during runtime causes some overheads and for optimizing the application for speed, a CPU frequency of 48 MHz is used for the whole application. The search area of the block-matching algorithm is configured to $\pm$3 pixels in x-direction and $\pm$8 pixels in y-direction. The block size of the block-matching algorithm is set to 48px, which was found to be optimal according to the pre-characterization.

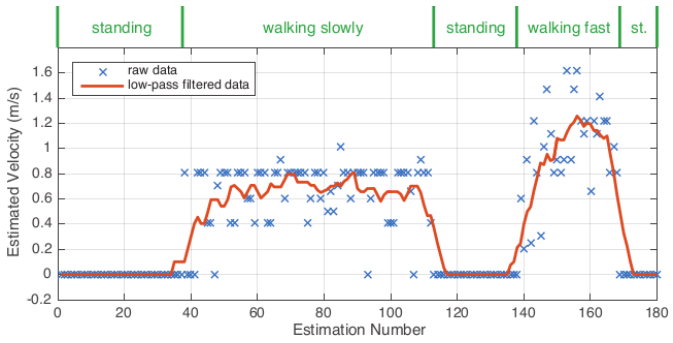

Fig. 5: Velocity of the walking person estimated by the vision sensor prototype (blue crosses) and post-processed in Matlab using a low-pass filter (red line).

Fig. 5 presents the walking speed estimations depicted as blue crosses, as well as the low-pass filtered data using a moving-average FIR filter in Matlab (solid line). The different phases of the experiment can clearly be recognized. Furthermore, the velocity estimation sensor is able to distinguish between different walking speeds.

VI. Conclusions

In this work, we have proposed a wearable prototype of a visual velocity estimation sensor that is able to operate without any battery. The vision sensor can be attached to someone's glasses to estimate the walking speed. It harvests energy from a solar panel to produce opportunistic velocity estimations, making the sensor reading rate a function of the input power. The hardware and firmware of the vision sensor has been highly optimized for ultra-low power consumption, high performance and satisfies the constraints of transiently powered systems. Different motion estimation algorithms were investigated for calculating a velocity estimation using visual information. Experimental evaluations show that the final vision sensor prototype can produce reliable velocity estimations in an energy-proportional manner. The concept of transiently powered systems is thus a promising approach for designing purely harvesting-based devices with significant form factor and cost restrictions. With the vision sensing prototype we have demonstrated that even computationally expensive applications can be efficiently and reliably executed in this context.

Acknowledgments This research was funded by the Swiss National Science Foundation under grant 157048: Transient Computing Systems. The authors would also like to thank D. Honegger and H. Oleynikova for their assistance.

References

[1] A. Barjatya, "Block matching algorithms for motion estimation," *Trans. Evolution Computation*, 2004.

[2] S. S. Beauchemin *et al.*, "The computation of optical flow," *ACM computing surveys (CSUR)*, 1995.

[3] N. A. Bhatti *et al.*, "Energy Harvesting and Wireless Transfer in Sensor Network Applications: Concepts and Experiences," *Trans. on Sensor Networks*, 2016.

[4] I. Centeye, "Stonyman and Hawksbill Vision Chips Silicon Documentation," 2013, datasheet.

[5] C. S. Corporation, "FM25V10: 1-Mbit Serial (SPI) F-RAM," 2015, datasheet.

[6] G. Farnebäck, "Very high accuracy velocity estimation using orientation tensors, parametric motion, and simultaneous segmentation of the motion field," in *Proc. ICCV Conf.* IEEE, 2001.

[7] J. J. Gibson, "The perception of the visual world." 1950.

[8] A. Gomez *et al.*, "Dynamic energy burst scaling for transiently powered systems," in *Proc. DATE Conf.* IEEE, 2016, pp. 349–354.

[9] D. Honegger *et al.*, "An open source and open hardware embedded metric optical flow cmos camera for indoor and outdoor applications," in *Proc. ICRA Conf.* IEEE, 2013.

[10] B. K. Horn *et al.*, "Determining optical flow," in *Artificial Intelligence 17*, 1981.

[11] T. Instruments, "MSP432P401R: Low-Power Mixed-Signal Microcontroller," 2015, datasheet.

[12] ——, "MSP Low-Power Microcontrollers," http://www.ti.com/lit/sg/slab034ad/slab034ad.pdf, 2016, accessed: 2016-07-22.

[13] M. Magno *et al.*, "Ultra-low power context recognition fusing sensor data from an energy-neutral smart watch," in *IoT Infrastructures: Second International Summit*. Springer International Publishing, 2016.

[14] A. Mayberry *et al.*, "ishadow: Design of a wearable, real-time mobile gaze tracker," in *Proc. MobiSys Conf.* ACM, 2014.

[15] V. More *et al.*, "Visual odometry using optic flow for unmanned aerial vehicles," in *Proc. CCIP Conf.* IEEE, 2015.

[16] H. Oleynikova *et al.*, "Reactive avoidance using embedded stereo vision for mav flight," in *Proc. ICRA Conf.* IEEE, 2015.

[17] K. Schneider *et al.*, "Computing optic flow with ardueye vision sensor," DTIC Document, Tech. Rep., 2013.

[18] L. Spadaro *et al.*, "Poster abstract: Kinetisee - a perpetual wearable camera acquisition system with a kinetic harvester," in *Proc. IPSN Conf.*, 2016.

[19] J. Zhao *et al.*, "Non-vector space landing control for a miniature tailed robot," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, Sept 2015, pp. 2154–2159.