Scamp5d Vision System and Development Framework

Demonstration Paper

Jianing Chen
The University of Manchester
jianing.chen@manchester.ac.uk

Stephen J. Carey
The University of Manchester
stephen.carey@manchester.ac.uk

Piotr Dudek
The University of Manchester
p.dudek@manchester.ac.uk

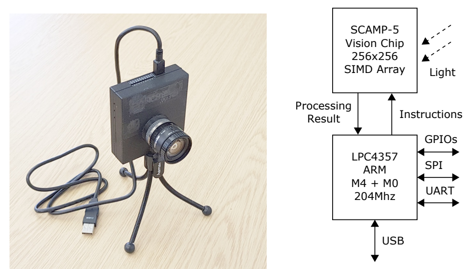
ABSTRACT

We demonstrate the Scamp5d integrated vision system and its development framework. The sensor of the camera is the SCAMP-5 vision chip, which provides sensor-level SIMD parallel processing capability, embedding a processor in each of its 256 × 256 pixels. A dual-core ARM micro-controller is used to operate the vision chip and provide additional computation capability and IO interfaces. The vision system is programmed using the C++ language. Common IO buses, as well as a USB2.0 port, allow the Scamp5d to be connected to micro-controllers, single board computers, or other hardware. The vision system can be remotely debugged, configured and re-programmed over the network. The open design of Scamp5d's software interface allows for easy integration with other software systems, such as ROS. As a self-contained vision system, Scamp5d can output highly processed data instead of video streams, which is suitable for applications such as miniature robots and distributed sensor networks. Simulation of the vision chip is provided through cross-compiling. Documentation, tutorials, and the simulation software are available for download.

ACM Reference Format:
Jianing Chen, Stephen J. Carey, and Piotr Dudek. 2018. Scamp5d Vision System and Development Framework: Demonstration Paper. In *International Conference on Distributed Smart Cameras (ICDSC '18), September 3–4, 2018, Eindhoven, Netherlands.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3243394.3243698

1 INTRODUCTION

Vision-based applications typically use a camera to capture video streams and then pass them on to digital processing hardware. Relatively high computational power is required to run vision algorithms, but traditional CPU/GPU systems may not be suitable for many applications due to their constraints in size and power consumption. Vision sensors[5] provide an alternative approach, shifting the computations to the sensor device itself. This provides performance and power advantages. High-bandwidth, high-performance parallel computing at the sensor level can be used to extract the relevant information from the raw pixel data, without transmitting it off-chip.



(a) (b)

Figure 1: (a) The 'Scamp5d' vision system. It weighs around 100 grams without lens and consumes around 1.5 to 2.5 Watts of power. (b) The architecture diagram of Scamp5d. The vision system is a stand-alone device that can handle many stages in a vision-based application, starting from capturing of the image to dispatching digested results.

2 HARDWARE

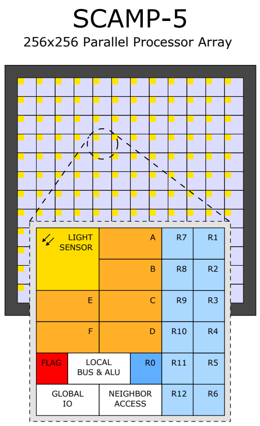
The Scamp5d vision system (Fig.1) is an portable integration of the SCAMP-5 vision chip [1] and an off-the-shelf micro-controller.

2.1 SCAMP-5 Vision Chip

The SCAMP-5 general-purpose programmable chip has been described in [1]. Briefly, it consists of 256x256 pixels, each with an associated processing element (PE), as illustrated in Fig.2(a). PEs are simple processors that have their own registers and ALU. In each PE, there are 7 read/write analog registers (AREG) which can store signed values. The advantage of using analog registers is that basic arithmetic operations, such as addition, subtraction, etc. can be performed with low power and ALU complexity, while retaining acceptable accuracy levels [3]. The light sensor of a PE is mapped to a read-only AREG. Each PE also has 13 1-bit digital registers (DREG). Boolean logic operations, such as OR, NOR and NOT, can be performed on DREGs. As a SIMD parallel processor array, all of the PEs execute the same instructions (provided by an external controller). AREG operations can be masked by a FLAG register so that data-dependent branch of a program can be realized across the processor array. A PE can also access any of its 4 neighboring PE's. Through iterative neighbor access, register values can be shifted to any nearby PEs in a local area. In addition to the PE array, the SCAMP-5 vision chip also provides hardware acceleration circuits for various image processing functions, such as flood-fill, low-pass spatial filters, global summations and address-event readout of sparse binary arrays. Overall, the chip provides a general purpose substrate for executing a variety of vision algorithms.

ICDSC '18, September 3–4, 2018, Eindhoven, Netherlands

ACM ISBN 978-1-4503-6511-6/18/09.
https://doi.org/10.1145/3243394.3243698



(a)

```
// Frame Loop
while(1){
   // sync with frame rate timing
    vs_wait_frame_trigger();

   // poll events, update threshold and readout_mode
    vs_process_message();

    // load global value to vision chip
    scamp5_in(E,threshold);// E = threshold

    // SCAMP-5 Kernel
    scamp5_kernel_begin();
        get_image(C,D); // C = this frame
        sub(D,C,B);     // D = this frame - prev. frame
        abs(A,D);       // A = |D|
        sub(A,A,E);     // A = A - threshold
        where(A);       // FLAG = (A > 0)
        MOV(R5,FLAG);   // R5 = FLAG
        all();          // FLAG = 1
        mov(B,C);       // previous frame = this frame
   scamp5_kernel_end();

   // output images when required
   if(readout_mode>=1){
       scamp5_output_image(R5,display_2);
   }

    // blink an LED to show working status
    if(vs_loop_counter_get()%30==0){
        vs_led_toggle(VS_LED_1);
    }
    vs_loop_counter_inc();
}
```

(b)

Figure 2: (a) The brief architecture of the SCAMP-5 vision chip. The pixel-processors in the SIMD array contain local memory and have the capability to perform arithmetic/logic operations. (b) A C++ code snippet for the vision system. Instructions of the vision chip are issued by the micro-controller, which is programmed using C++ language. Code sections called 'SCAMP-5 Kernel' contain atomic functions to be executed in parallel on the vision chip.

2.2 Micro-Controller

The micro-controller (MCU) used is an NXP LPC4357, integrating two ARM Cortex cores, M4 and M0, running at 204 MHz. The two cores share an internal SRAM of 132KB. The programs are stored in internal Flash memory of 512 KB. Instructions for the vision chip are dispatched by the M0 core, with the PE array executing analog operations at 5 MHz and digital operations at 10 MHz. The M0 core also performs other vision chip control tasks. The M4 core is used to perform IO services and has the capability to run user programs. Programming of the vision system is essentially programming the MCU. Serial IO buses, such as USB2.0, SPI, and UART, allow the output from the vision system to be sent to a variety of other devices in a system.

3 DEVELOPMENT FRAMEWORK

3.1 Compiler and Library

The vision system is programmed in GNU C/C++ language. An extensive development library provides easy-to-use APIs for the microcontroller's peripherals and the vision chip's functionality. Computation executed in parallel on SCAMP-5 PEs are grouped into 'SCAMP-5 Kernels', which are written in special sections among the M0 source code (Fig.2b). Kernels compiled into SCAMP-5 instructions on the first encounter when the program is running.

Vision algorithm source code running on the M0 core can be also cross-compiled into a simulation client on any PC. The simulation client connects and sends instructions to a simulation server which emulates and provides visualization of the state of registers within the SCAMP-5 vision chip. The simulation client can be launched independently under a debugger and stepped-through line-by-line, which eases the development of a vision algorithm.

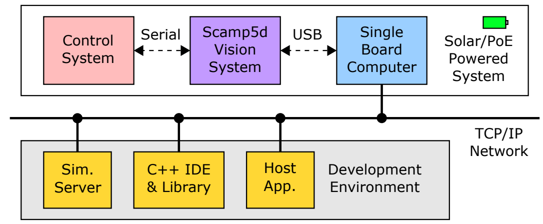


Figure 3: A networked development framework. While the vision system can work standalone, it can also be operated remotely via TCP networks through single board computers (e.g. a Raspberry Pi). Interactive monitoring, tuning, and re-flashing are also possible through such a network.

3.2 Host Environment

While the Scamp5d system can operate fully autonomously, it can also be networked through single board computers or regular PCs via the USB2.0 port. A driver library for both Windows and Linux OS is provided. All communications with the vision system can be bridged to TCP interface using a utility. This means the vision system can be interacted with remotely over Ethernet or WiFi networks. A pre-made GUI Host Application is also developed to program, debug, and interact with the vision system. This application is capable of rendering and recording images, event streams, and other output data received from the vision system or the simulation server using various types of plots. Users can customize their own GUI items (e.g. buttons and sliders) in the M0 program to interactively tune parameters during runtime. A networked development framework example is illustrated in Fig.3. Re-flashing of the M0 program can also be done through the network.

4 DEMONSTRATIONS

The software packages are available for download in[2]. A Scamp5d vision system has been recently applied in the control and navigation of MAVs[4]. A live demonstration will showcase the entire vision system and its development environment including compiler, simulator, and host GUI. A series of vision algorithms will also be demonstrated.

ACKNOWLEDGMENTS

This work has been supported by EPSRC Grant EP/M019284/1.

REFERENCES

[1] S.J. Carey, D.R.W. Barr, B. Wang, A. Lopich, and P. Dudek. 2013. A 100,000 FPS vision sensor with embedded 535 GOPS/W 256×256 SIMD processor array. In *Proc. 2013 IEEE Symp. VLSI Circuits*. 182–183.

[2] J. Chen. 2018. Scamp5d Vision System Development Framework Documentation. http://personalpages.manchester.ac.uk/staff/jianing.chen/scamp5d_lib_doc_html/index.html.

[3] P. Dudek and P. J. Hicks. 2005. A general-purpose processor-per-pixel analog SIMD vision chip. *IEEE Trans. Circuits and Systems I: Regular Papers* 52, 1 (Jan 2005), 13–20.

[4] C. Greatwood, L. Bose, T. Richardson, W. Mayol-Cuevas, J. Chen, S. J. Carey, and P. Dudek. 2017. Tracking control of a UAV with a parallel visual processor. In *Proc. 2017 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*. 4248–4254. https://doi.org/10.1109/IROS.2017.8206286

[5] A. Zarandy, R. Dominguez-Castro, and S. Espejo. 2002. Ultra-high frame rate focal plane image sensor and processor. *IEEE Sensors Journal* 2, 6 (Dec 2002), 559–565.