2008 International Symposium on Nonlinear Theory and its Applications
NOLTA'08, Budapest, Hungary, September 7-10, 2008

Application of Cellular Wave Computers in High Speed Real-Time Processing: Measurements of a Finger Tracking Algorithm for Human-Machine Interface

Norbert Bérci and Péter Szolgay

Faculty of Information Technology
Péter Pázmány Catholic University
Práter u. 50/A, 1083 Budapest, Hungary
Email: berci.norbert@itk.ppke.hu, szolgay.peter@itk.ppke.hu

Abstract—Several application areas (virtual and augmented reality, medical and industrial applications, machine control for disabled peple to name a few) need novel human-machine interfaces to be able to express the required complex command set or to adapt to special needs. One of the ambitious solutions is control via hand gestures. We have recently developed a set of algorithms which solve the task of tracking a human finger in ordinary visual conditions for the replacement of the conventional mouse. In order to utilize the algorithm successfully we had the challenge to design it such a way that it is able to run smoothly in a real-time environment. The only viable alternative was to do it via a spatio-temporal algorithm on a cellular wave computer. In this paper we present measurements of this algorithm implemented on some current cellular wave computer platforms and investigate the properties, representative parameters, limits of it.

1. Motivation

Nowadays graphical user interfaces become more and more complex to keep pace with the need to express a vast number of commands for detailed control. There are also application areas where conventional human-machine interfaces like the keyboard and the mouse are hard or even impossible to use: let us think about an operating room or laboratory where reducing the number of physical contacts between devices and (possibly several different) users means to be able to reduce the risk of infections. In the field of computer aided design, engineering and especially in virtual reality systems control via hand gestures is a natural choice. Unlike voice control – the other natural command mechanism – selectivity is as simple as setting the viewport of the camera, or zooming in.

In this paper we examine two different version of a finger tracking algorithm and their run-time properties implemented on two different non-linear analog processing systems.

2. The CNN Architecture and its Hardware Implementations

We have choosen the CNN architecture for design and implementation platform, since it is extremely well suited to image processing tasks. It has processing elements arranged in a grid which allows one-to-one mapping of each to a pixel. Due to its analog and locally connected nature, processing is very fast and it allows non-linear dynamics to appear. A more detailed description can be found in [1].

2.1. The Bi-i System

The first development platform is the Bi-i system [2][3] which is a standalone smart camera. The main parts are a CNN type ACE16k [4] image sensor-processor analogic chip for the spatio-temporal image processing tasks, a Texas Instruments TMS320C6415 DSP for fixed point computations and an ETRAX 100 LX from Axis as a communicating processor.

2.2. Finger Tracking on the Bi-i System

The algorithm implementing finger tracking on the Bi-i system can be seen on figure 1. The two main parts of the algorithm are the pixel processing part which runs on the ACE16k processor and contains the preprocessing, skeltonization, masking and centroidization parts and the hand model part which runs on the DSP and its role is to support and aid the recognition task. It is utilized for error recovery also. A more detailed explanation can be found in [5].

2.3. Measurements

As can be seen on figure 2 the average processing time is about 12 ms, but the actual time is a bit unsteady, mainly due to the hand model and the input dependent processing algorithm parts (error recovery, skeletonization).

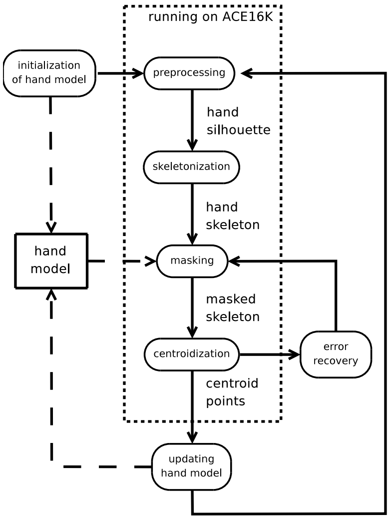

Figure 1: The flowchart of the implemented algorithm. Blocks inside the dashed line run on the ACE16K visual processor, all the others have been implemented on the DSP.

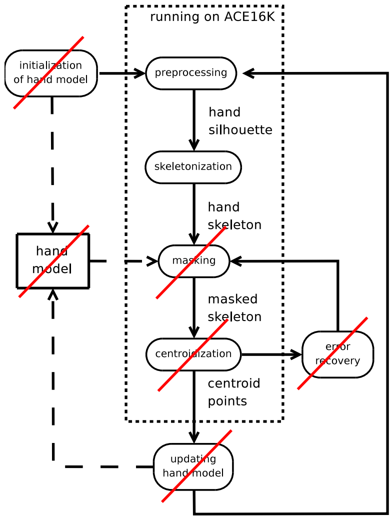

Figure 3: The flowchart of the implemented algorithm. Some algorithm parts were dropped out, so it is completely executed on the Q-Eye visual processor

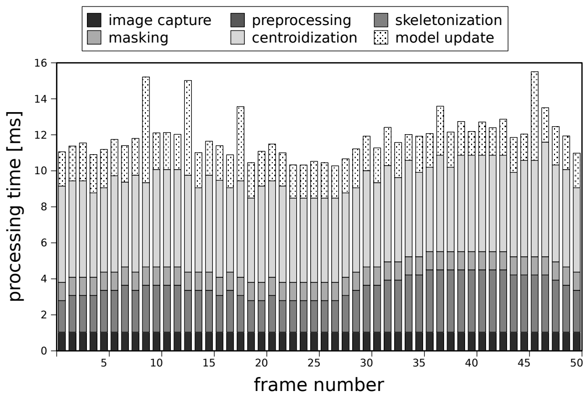

Figure 2: Measurements of frame processing times broken up into smaller algorithmic parts of the flowchart.

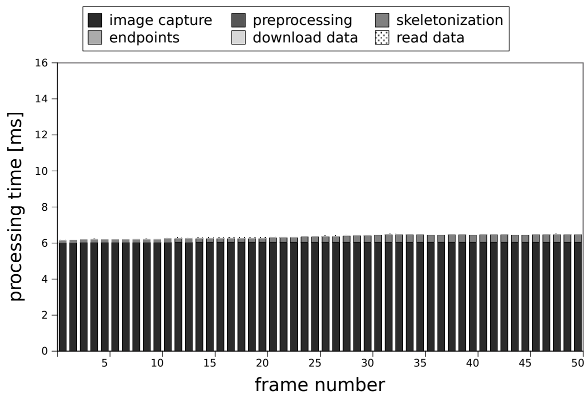

Figure 4: The flowchart of the implemented algorithm. Blocks inside the dashed line run on the ACE16K visual processor, all the others have been implemented on the DSP.

3. The eye-RIS System

The eye-RIS system is a programmable vision system using Anafocus' Smart Image Sensor technology (Q-eye chip), and an Altera NIOS-II 32 bit RISC processor implemented in FPGA for the control and ordinary computing tasks.

3.1. Object Tracking on the eye-RIS System

Compared to the previous version, several algorithm components have been removed. We gained faster processing, but the resulting system is not so robust (input should be higher quality than the first version). All parts of the algorithm run on the Q-eye chip, so the conventional RISC processor in the system has not been used for computation. Since we have removed the hand model, it is now considered a general object tracking system.

3.2. Measurements

The measurements of the modified algorithm running on the Eye-RIS system is presented in figure 4. The processing time has been remarkably decreased mostly due to the removed algorithm components. What it the main advantage of the new algorithm is that despite its simplicity, it solves the same problem. The frame processing time of approximately 6.4 ms enables very high frame rate computation. It is also worth to note that the major part of the time

subtask	avg. ratio	avg. time
subtask	of frame time	spent
image capture	94.42%	6,0399
preprocessing	0.05%	0,0034
skeletonization	4.72%	0,3038
endpoint detection	0.55%	0,0353
points download	0.08%	0,0054
read data	0.17%	0,0111

Table 1: Ratio between the different computation subtasks when computing one frame.

spent on image grabbing. Table 1 summarizes the average ratio between each computation task. The computation itself takes only a fraction of a milisecond, and its major part is the skeletonization.

4. Preprocessing

The algorithm presented only deals with object (finger) tracking. Its input is a segmented image: background removed, only the object should be present. Despite segmentation in general is a difficult task, in our case it can be solved by several different means: in the case of finger tracking it can be based on skin color discrimination, otherwise object motion based or high contrast based segmentation can be utilized. We have used the latter, since high illumination is also needed for high frame rate image acquisition.

5. Skeletonization in Central Role

We must emphasize that the vital part of the algorithm is the skeletonization. We have used the native implementation of this algorithm on both systems, but it turned out that the robustness and precision could be greatly improved by another skeletonization process. The main problem is the relative unstability of the skeleton which means that under special circumstances, little (in extreme cases only one pixel) difference in the input image makes a huge difference in the computed skeleton. Different skeletonization algorithms are under investigation.

6. Conclusions

- execution up to 100-150 FPS under ordinary (office) lightning conditions
- the second version of the algorithm meets the even stricter requirements of real-time systems, since deviances in computation time are minimal, the computation time varies only in the skeletonization part, which itself depends on the tracked object size and shape
- processing time only verly slightly depends on the number of object tracked
- the algorithm is able to track multiple objects and under ordinary (office) conditions, tracking time very slightly depends on the number of the tracked objects

7. Acknowledgments

The authors wish to express their gratitude for their institute, the Péter Pázmány Catholic University, Faculty of Information Technology and especially to professor Tamás Roska for his thoughts and insights which guided us in the research. We also wish to thank to all of the collegaues who we have discussed the topics with and their comments greatly improved this paper.

References

[1] L. O. Chua and T. Roska, *Cellular Neural Networks and Visual Computing*. Cambridge, UK: Cambridge University Pess, 2002.

[2] Á. Zarándy and C. Rekeczky, “Bi-i: a standalone cellular vision system, Part I. Architecture and ultra high frame rate processing examples,” in *Proc. 8th Int. Workshop on CNNs and their Appl. (CNNA)*, Budapest, Hungary, Jul. 22–24, 2004, pp. 4–9.

[3] ——, “Bi-i: a standalone cellular vision system, Part II. Topographic and non-topographic algorithms and related applications,” in *Proc. Int. Workshop on CNNs and their Appl. (CNNA)*, Budapest, Hungary, Jul. 22–24, 2004, pp. 10–15.

[4] G. Liñán, R. Domínguez-Castro, S. Espejo, and Á. Rodríguez-Vázquez, “ACE16k: A programmable focal plane vision processor with 128x128 resolution,” in *Proc. Eur. Conf. on Circ. Theory and Design (ECCTD)*, Espoo, Finland, Aug. 28–31, 2001, pp. 345–348.

[5] N. Bérci and P. Szolgay, “Vision based human-machine interface via hand gestures,” in *Proc. Eur. Conf. on Circ. Theory and Design (ECCTD)*, Seville, Spain, Aug. 26–30, 2007, pp. 496–499.