An Optimized Silicon Retina Stereo Matching Algorithm Using Time-Space Correlation

Christoph Sulzbachner, Christian Zinner, Jürgen Kogler
AIT Austrian Institute of Technology GmbH
Safety & Security Department, Safe & Autonomous Systems
Donau-City-Strasse 1, 1220 Vienna, Austria
{christoph.sulzbachner|christian.zinner|juergen.kogler}@ait.ac.at

Abstract

This paper presents an optimized implementation of a Silicon Retina based stereo matching algorithm using time-space correlation. The algorithm combines an event-based time correlation approach with a census transform based matching method on grayscale images that are generated from the sensor output. The data processing part of the system is optimized for an Intel i7 mobile architecture and a C64x+ multi-core digital signal processor (DSP). Both platforms use an additional C64x+ single-core DSP system for acquisition and pre-processing of sensor data. We focus on the performance optimization techniques that had a major impact on the run-time performance of both processor architectures used.

1. Introduction

Stereo vision describes the analysis of a scene captured by two different points of view. The aim is finding corresponding points in the left and right image indicating the projection of a 3D point. For solving the correspondence problem within the images, various computational methods exist including area- and feature-based approaches [10, 12].

For biologically inspired stereo vision systems using Silicon Retinas, those approaches are restricted. These types of sensors are based on bio-inspired analogue circuits and have special characteristics. They do not deliver conventional image frames at a fixed frame-rate, but utilize an efficient, asynchronous, event-based data protocol that delivers information only on variations of intensity in a scene. Common image processing optimization techniques are improper due to the unconventional data interface. The aim of developing algorithms for these types of sensors is processing the asynchronous data directly, as it is delivered by the imagers. Reconstructing classical frame-based image representations from the asynchronous sensor data in memory combined with using conventional image processing algorithms throttles the performance significantly [6].

The introduced time-space correlation algorithm is both optimized on an Intel based architecture with intensive use of the Streaming SIMD Extensions (SSE), and an embedded distributed multi-core system consisting of both a C64x+ single-core for data acquisition and pre-processing, and a multi-core digital signal processor (DSP) for data processing. We show the performance analysis of the stereo matching algorithm and focus on the optimization techniques that had a major impact on the run-time performance for the processor architectures.

This paper is organized as follows: Section 2 gives an overview about existing work on Silicon Retina sensors, related stereo matching techniques and optimization techniques. Section 3 explains the concept of time-space correlation. Section 4 gives details about various performance optimizations on the Intel architecture using Microsoft and Intel C/C++ compilers, as well as on the Texas Instruments (TI) TMS320C64x+ architecture. Section 5 concludes the paper and gives an outlook to future work.

2. Related Work

The history of the Silicon Retina technology goes back to 1988, where Mead and Mahowald [7] developed a first silicon based implementation of a retina. Later in 2002, Kramer [5] presents an image sensor that encodes the contrast of illumination transients. Lichtsteiner *et al*. [8, 9] show recent developments in bio-inspired sensor systems that establish the basis for this type of sensor technology. Recent developments have higher sensor and time-stamp resolution [1].

2.1. Event Concept

The Silicon Retina technology is using an address-event-representation (AER) for exchanging data. AER was proposed in 1991 by Sivilotti [14] as a communication method

for exchanging neuronal information within biological systems. Later, AER has been modified to a protocol for exchanging asynchronous data streams. Each time a variation of intensity is recognized by the sensor, an event $\mathbf{E}$ is emitted by the sensor. Unaltered parts of a scene without changes in illumination need neither be transmitted nor processed. Thus, the amount of AER data depends on the actual scene.

An event $\mathbf{E}$ is a tuple of the pixel coordinates x and y, a timestamp t, and a sign s. The sign indicates the direction of the variation on intensity. The AER interface consists of two types of messages, a timestamp message TS and an event message AE, both using 32-bit bit-fields for data representation.

2.2. Real-Time Stereo Vision

Conventional stereo matching techniques require two imagers mounted in a certain distance observing a scene from two different points of view. For computing the disparity map of a scene, various algorithm approaches exist, covering both area- and feature-based approaches. Area-based approaches are based on computing image patches on both images to find a correlation maximum to determine the disparity information. In contrast, feature-based approaches extract features from both images trying to correlate those, finding correlation maxima [13]. In both cases, the correlation maximum for a pixel indicates its disparity.

Zinner *et al*. [2] shows an optimized software based implementation of a stereo matching algorithm based on a census transform with a large mask size using an Intel Core 2 Duo running at 2GHz. In their work, they cover performance optimization techniques for fast Hamming distance calculation, fast census transform and fast aggregation. Using those techniques a speedup factor of 112 of the functional behavior implementation and a performance of 42fps for QVGA could be achieved. In our work, we are using some of these techniques for the area-based correlation part of our time-space correlation algorithm.

Schraml *et al*. [11] shows a stereo matching approach with a performance of 200fps for a resolution of 128×128 using standardized area-based correlation algorithms on reconstructed images using sorted dynamic lists.

An evaluation of state-of-the-art stereo matching algorithms including both area- and feature-based approaches revealed that such methods cannot exploit the main characteristics of the Silicon Retina sensor technology properly [6].

3. Time-Space Correlation

The stereo vision algorithm is based on the correlation of the events in time and space. Both algorithm paths are calculated separately and combined at the end to a final resulting disparity map. Figure 1 shows the overview of the algorithm with all steps. The stereo matching consists of a time correlation that is processed for each event and a census-based correlation that is processed on generated grayscale images. The space correlation part of this time-space correlation stereo matching algorithm is covered in the work of Humenberger *et al*. [3]. Only the time correlation methods are further described.

Rectification, Undistortion: In a previous offline calibration process the required parameters are determined for the left and right channel and Look-Up-Tables (LUT) are created to enable fast online rectification. During the mapping from source coordinates to the destination coordinates the event can be placed in sub-coordinates between the main coordinates. In case of Silicon Retina data, events are represented binary and are not interpolated for the calculation of sub-coordinate information. Therefore, a nearest neighbor method is used where the rectified events are mapped to the grid point closest to the destination coordinates.

Scale: The scale step is an optional reduction of the calculation effort for data processing by reducing the data resolution in the x- and y-direction by the same amount. Scaling by a factor of n implies that the disparity range also needs to be modified at the same amount. Scaling of the coordinates for the scale factor 2^n is computed by

$$AE'_{rect}(x', y', t) = \begin{cases} n > 0 & AE_{rect}(x \; shl \; n, y \; shl \; n, t) \\ n \leq 0 & AE_{rect}(x \; shr \; n, y \; shr \; n, t) \end{cases} \tag{1}$$

For reconstruction of the final disparity map DM', the reduced coordinate space has to be scaled back with a scaling factor of $-n$. Scaling speeds up the performance of the algorithm significantly, but reduces the quality of the disparity calculation.

Time Correlation: All rectified events are directly processed by the time correlation procedure by matching them against the event history of the other imager according to their concurrency. Based on the concurrency of a match, a concurrency gauge can be directly derived. All calculated gauges are written into the *Weighting Matching Image* (WMI). The correlation is calculated according to

$$\mathrm{WMI}_t(x, y, d) = \begin{cases} wf(\tau(\mathrm{AE}_{rect,l}(x, y, t)) - \tau(\mathrm{AE}_{rect,r}(x - d, y, t - n\Delta t))), \\ \qquad \mathrm{AE}_{rect,l}(x, y, t) = \mathrm{AE}_{rect,r}(x - d, y, t - n\Delta t) \\ 0, \text{no matching candidate} \end{cases} \tag{2}$$

where

$$\tau(\mathrm{AE}_{rect}(x, y, t)) = t \tag{3}$$

and the weighting function is defined as

$$wf(\Delta t_{lr}) = \begin{cases} n_{max}\Delta t - \Delta t_{lr} & \text{inverse linear} \\ n_{max}\Delta t \cdot e^{-\frac{\Delta t_{lr}}{a}} & \text{gauss} \end{cases} \tag{4}$$

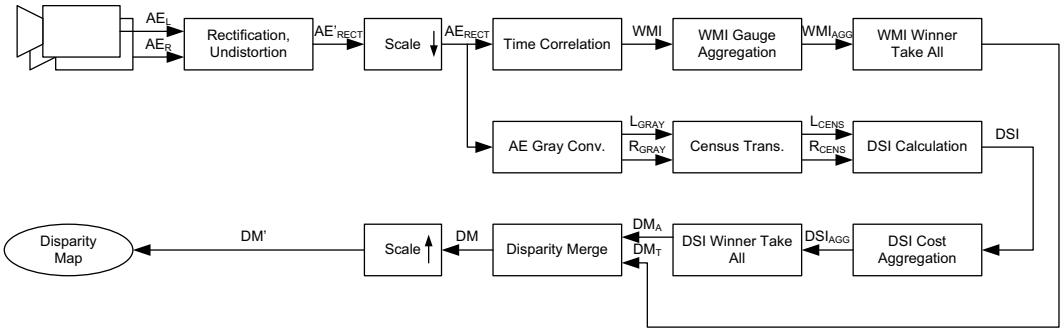

Figure 1. Time-Space Correlation Stereo Matching Approach

This means that pairs of events which are closer in time are emphasized. At the same time, all gauges inside of the WMI data structure are regularly decreased by a specific amount. This approach of continuously decaying the gauges, making the correlation method noise resistant, because small spikes that might originate from stochastically matched noise events will vanish very fast. Stronger correlation spikes caused by real scene content will still be distinctive for some time, but once there are no more changes in the scene, these spikes will also disappear. Therefore, the behavior of the time correlation part is similar to the Silicon Retina sensor itself, since disparity information vanishes when scenes become static.

WMI Gauge Aggregation: In an optional aggregation step all gauges are spatially aggregated, which smooths the results and helps in eliminating outliers. For this operation, a filter kernel with a variable mask size can be selected.

WMI Winner Takes All: For computing the best match, each disparity plane is analyzed finding the highest concurrency gauge whenever a match is valid. This value directly indicates the disparity information of a coordinate corresponding to the left image. A minimum threshold is used for avoiding the contribution of strong noise to the disparity map and previous calculated gauges which are still in the WMI. All results from the maximum search are written into the disparity map which is further combined with the results from the spatial disparity search.

AE Gray Conversion: The spatial calculation of the stereo matching is processed on generated grayscale images. Therefore events are integrated over time using the time correlation history as integration interval. As a default value for background a value of 128 is used. A positive signed event contributes with a positive value to the grayscale image, respectively a negative signed event contributes with a negative value. This conversion procedure is computed by

$$\mathrm{p}_{t,l/r}(x,y) = 128 + \sum_{i=t-n\Delta t}^{t} \mathrm{AE}_{rect,l/r}(x,y,i) \qquad (5)$$

After a time cycle, the grayscale image converges to the default value. Depending on the actual motion in the scene, this factor can be modified for gathering high-contrast images.

Census Transform: A census transform is a robust method for representing the intensity differences within pixel patches used for the calculation of matching costs. The resulting bit strings describe the differences for each pixel. A 8×8 census mask is used and applied on the generated grayscale images.

DSI Calculation: For cost calculation, the Hamming distances of these bit strings are computed and written into the Disparity Space Image (DSI). In the DSI the number of unequal bits within the bit strings are stored.

DSI Cost Aggregation: An optional step is cost aggregation for smoothing the costs and eliminating outliers. Aggregation is realized by an ordinary boxfilter with a configurable filter size is applied to the complete DSI.

DSI Winner Takes All: The best match is searched in the DSI for each coordinate pair within the disparity range. Each x-y pair is analyzed and the lowest cost value represents the disparity of the most plausible match. All disparity found values are then written into the disparity map.

Disparity Merge: After the calculation of both disparity maps DM_A and DM_T, the valid matches are further merged into a final disparity map DM. The data merging is done according to

$$DM(x,y) = DM_T(x,y) \times (1-\alpha) + DM_A(x,y) \times \alpha \quad (6)$$

where α represents the merge factor.

4. Performance Optimization

The algorithm is implemented as a module allowing multiple instances at the same time. To achieve higher performance, multiple instances with different scale factors can be processed in parallel. The following performance optimization techniques can be applied independent of the scaling factor. The performance metric cycle-per-pixel (cpp) is

used as a common metric, which describes the normalized processing performance of an architecture.

4.1. Event-Based Optimization

Compiler perform various optimizations to improve the performance of programs. An essential technique for optimizing loops is software pipelining, what is an optimization technique supported by the compiler to reschedule loops so that the cycles for executing a loop can be minimized and the data output is maximized. To exploit those compiler features, the user has to adapt the code for the compiler manually in a vast of cases.

The bottlenecks of efficiently AER data processing are that the events are unsorted in terms of the coordinates depending on the actual scene, and TS and AE messages are compressed, meaning that multiple AE with the same timestamp are transmitted after one TS.

AE bursts only have a few kiB and are organized in internal memory for fast data access. Generally, buffers with larger data amounts are organized in the external memory. For efficiently data processing from external memory, data is either load to the internal memory in smaller amounts, processed and copied back to their original address using DMA transfers in the background, or level 2 caching is applied. In our approach, we are using both approaches.

Additionally, algorithms have to differ between AE and TS datatypes. The simplest way is to implement a condition statement in a loop statement. Depending on the type of compiler, this may disqualify the loop from software pipelining because it contains control code. For avoiding branches and to afford software pipelining, a method to avoid the condition statement is required. Therefore, all data is processed in common disregarding its datatype. If the datatype was correct, the results will be used, otherwise the processed data is obsolete.

This approach can be applied to a vast of time-independent functions *e.g.* rectification or scaling. Time-dependent functions *e.g.* time correlation can further be optimized using efficient data structures.

4.2. Efficient Data Structure

Generally, the AER data stream is unstructured in terms of the coordinates of the events. Optimizing algorithms that need to process a non-coherent amount of data in memory is very challenging, due to cache problems or the inefficient access of the memory. To optimize those functions is generally memory bound, meaning that the program performance is limited by memory usage.

For defining a data structure in memory, the data characteristics is important. Silicon Retina data has a very sparse data characteristic, meaning that only detected motion is delivered and unaltered parts of a scene without motion will not cause events.

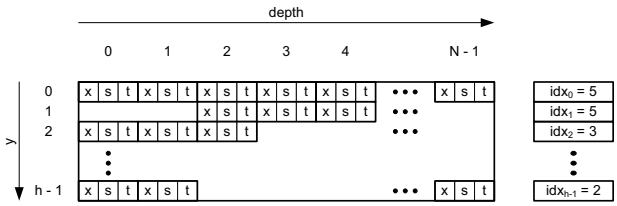

Figure 2. Principle of ring-based data structure showing a spare set of AE.

The introduced stereo matching algorithm requires a parametric history of time for event correlation. Handling image data over time, results in a memory structure that requires $width \times height \times delay$ bytes using 8 bits per event. A $5ms$ data history with a timestamp resolution of $10ns$ would result in a required memory amount of $34GB$.

Thus, for data compression and to avoid problems with spatial matrices, we are using a ring-buffer data structure. The buffers are organized as a coherent memory segment using an index idx_y for addressing the write position per line. For fast data access and index manipulation modulo operations and circular ring overflowing is used.

Figure 2 shows the principle of the data structure for storing a sparse set of events. The data structure consists of one ring-buffer per line of the image, that is able to handle N events. All acquired events are inserted into the corresponding line of the ring-buffer.

This concept allows representing the sparse cube data structure as coherent memory. Thus, block transfers using DMA transfers or caching can be applied. The introduced data structure is not able to handle all events in the specific time slice like the cube data structure, but has the advantage of an efficient memory access and guarantees handling a defined amount of events. Additionally, handling multiple events fired at the same coordinate with different signs is supported.

4.3. Fast Event Scaling

Scaling is always a trade-off between quality and efficiency. To avoid additional computational effort, the scaling factors are restricted to powers of two and the sign of the events is not weighted. Thus, for event scaling only the coordinates of an event need to be modified using shift operators. Depending on the available register set of the target platform, multiple events can be loaded, processed and stored in parallel.

Intel: The memory access is optimized using the `_mm_loadu_si128` and `_mm_storeu_si128` instrinsics handling four events in parallel. The processing is optimized by exploiting several logical intrinsics.

TMS320C64x+: Due to the event data size of 32-bit, using advanced memory intrinsics do not improve the performance significantly. Performance improvements are pos-

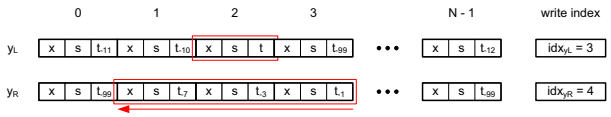

Figure 3. Principle of time correlation showing the reverse iteration for correlating index 2 from the left imager to the event history of the right imager starting with index 3.

sible exploiting the data packing and logical intrinsiscs, allowing processing coordinates in parallel.

4.4. Fast Time Correlation

Event time correlation is obtained by computing the concurrency of an event that appeared at one imager to the event history of the other imager in the specific disparity range. Following the lens undistortion and rectification step of the stereo matching algorithm, the search space is reduced to one dimension of the actual line. As a simplification, the correlation does not handle multiple events with the same coordinates and different signs. The correlation operation is required to operate in a time-valid data range, iterating the buffer and comparing all items. Due to the particular structure of the data using only write indices and overflows, the iteration direction can be reversed.

Thus, the matching process enables a speedup depending on the number of entries in the ring-buffers. In the worst case, matching the whole array with length N is required. Experimental results showed that depending on the motion of the scene the matching performance could be significantly improved.

Figure 3 shows an example for correlating an event from the left imager to the event history of the right imager in reverse direction. As long as the time delay $t_L - t_R$ is within a valid range or the iteration wraps over, the system continues matching events. In the example, the timestamp t_{-99} is invalid and thus it stops the matching process.

Intel, TMS320C64x+ A major performance boost has been achieved using the reverse iteration direction. Minor improvements were possible by splitting larger loops and using intrinsics.

4.5. Fast WMI Gauge Aggregation and WTA

The WMI data structure has the dimensions of $x \times (y \cdot (d_{max} - d_{min}))$ using 8 bpp. This memory layout is adequate for writing data by the correlation step, but for filtering all disparity planes the layout is not usable. For optimizing the memory access, the memory of the WMI is reorganized without modifying the data, shown in figure 4. A similar method was introduced by Zinner *et al.* [2] for restructuring data for aggregation using a census-based matching.

The data in the WMI is processed in parallel finding one local minimum in each line.

Intel: Aggregation is optimized by employing `ippiFilter_8u_C1R`. For the WTA, the memory access is optimized using `_mm_loadu_si128` and `_mm_storeu_si128`. Data is processed in chunks of 16 pixels exploiting arithmetical and logical intrinsics.

TMS320C64x+: Aggregation is optimized by processing four pixels in parallel using the `_mem4` intrinsic for data exchange and by exhaustive use of the data packing and arithmetic intrinsics. The WTA is further optimized with the `_maxu4` intrinsic.

4.6. Fast Disparity Merge

The disparity merge is based on two multiplications and one addition. The function is modified in terms of reducing redundant data processing by pre-processing.

Intel: The memory access is optimized using `_mm_loadu_si128` and `_mm_storeu_si128` and the data processing was optimized using pack and arithmetical intrinsics.

TMS320C64x+: Aggregation is optimized by processing four pixels in parallel using the `_mem4` intrinsic for data exchange and by an exhaustive use of the `_mpyu4` and `_add2` intrinsics.

5. Platforms

The target platforms for the stereo matching algorithm are an Intel i7-620M architecture for offline data analysis and a distributed embedded system using both a single-core and a multi-core fixed-point DSP.

The performance of the time correlation step is strongly depending on the valid data history that is used for correlation. For the evaluation of the performance speedup of the algorithm, practical settings and a recorded scene with real data were used. The intended scenario is used for time critical decision applications.

5.1. Intel i7

The Intel Core i7-620M is a two core mobile processor commonly used in notebook PCs clocked at 2.66GHz. The platform supports the Streaming SIMD extensions SSE4.2, which were vital for the achieved performance. As a development environment, we evaluated both Microsoft Visual

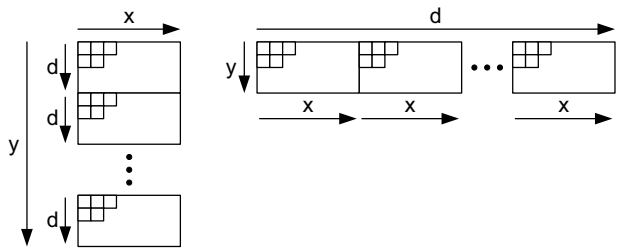

Figure 4. Principle of modifying the memory view for aggregation WTA.

Studio 2005 and Intel C++ Studio XE 2011. Both compilers can deal with the Intel Performance Primitives for Image Processing (ippIP) [4].

5.2. Texas Instruments TMS320C64x+

The distributed embedded system is a prototype for a final system using a C6455 DSP starter kit (DSK) and a C6474 evaluation module (EVM). The C6455 DSK is used for data acquisition and pre-processing and the C6474 EVM consisting of two DSPs each with three cores is primary used for stereo matching. For data processing only one DSP is used. The C6455 has $2MB$ of on-chip internal memory storing all performance relevant data in the internal memory. The C6474 is configured to use $1MB$ internal memory per core storing only performance relevant data structures in internal memory. Both DSPs are running at $1GHz$. For accessing data from the external memory, both DMA block transfers for exchanging data between external to internal memory, and level 2 caching is used.

Data Acquisition: Both Silicon Retina sensors are memory mapped to the external memory interface (EMIF) using first-in-first-out (FIFO) buffers. A direct memory access (DMA) transfer is processed automatically, when the half full interrupt is triggered. The next DMA context is automatically setup using the linking feature of the DMA controller. As soon as the data is acquired completely a transfer completion interrupt is triggered, initiating further pre-processing.

Data Pre-processing: The acquired data is labeled to identify the imager and pre-processed with noise filters, rectification and lens undistortion, and a static load balancer. After rectification and lens undistortion, it is guaranteed that events in a scene are in the same line in the left and right coordinate system. Thus, a static load balancer partitions the events to core n depending on the height segment following

$$n = \frac{E(y)N}{H} \tag{7}$$

where H is the height of the optical resolution, N is the number of cores and $E(y)$ is the y-coordinate of the rectified event.

Inter-core and Inter-chip communication: The C6455 DSK and the C6474 EVM are connected over Serial Rapid IO. For data acquisition, the C6474 has a reserved memory segment and each core has a separate core context. These parameters and the location of the algorithm instance are known by the C6455 by loading the memory map of the C6474 after startup. The C6455 transmits the pre-processed data to the reserved segment that is able to handle N bursts organized in a ring data structure. Before transmitting the data, the core context is read to determine the next position. Afterwards, the raw data and the modified core context are transmitted to the C6474 followed by a SRIO doorbell triggering an interrupt. This interrupt is directly routed to a specific core, where the location and size information of the received raw data is determined from the core-context and further processed with the stereo matching algorithm.

Table 2 shows the performance comparison of the stereo matching algorithm for the distributed embedded system.

6. Conclusion and Future Work

In our work, we presented a software optimized implementation of a stereo matching algorithm for Silicon Retina sensors. Due to the special characteristics of these types of sensors, we introduced a time-space correlation method for stereo matching. The algorithm is optimized for an Intel i7-620M mobile architecture and a TMS320C64x+ based distributed embedded system. For the Intel architecture, we evaluated two different compilers resulting in an overall average performance boost of 3 for the compiler optimized version of the functional behavior implementation, and 20.86 and 20.93 for the hand-optimized implementation. On the embedded system, we achieved a performance boost of 45.92.

Next, we will improve the stereo matching algorithm in terms of noise resistance and enhanced disparity merging. We expect that the performance of these modifications can also be improved with the introduced techniques.

Acknowledgment

The research leading to these results has received funding from the European Community's Seventh Framework Program (FP7/2007-2013) under grant agreement n° ICT-216049 (ADOSE).

Function	Calls	Unopt. [cpp]	Opt. [cpp]
Rect., Undist.	1	179.86	2.67
Scale	1	1398.49	28.60
Time Corr.	1	12002.35	209.53
WMI C. Aggr.	1	12210.18	27.00
WMI WTA	h	143.03	7.41
AE Gray Conv.	1	297.54	20.47
Census Trans.	2	7195.42	337.32
DSI C. Calc.	1	1305.88	10.51
DSI C. Aggr.	h	12762.03	405.47
DSI WTA	h	185.54	8.97
Disp. Merge	1	1132.88	5.04
Total		488813.20	1062.99

Table 2. Performance of the unoptimized and optimized implementation on the distributed embedded system. Resolution is QVGA, disparity range is 30, time correlation history is 5 and scaling factor is 1 with data burst of 7kAE. Optimization is further using level 2 caching.

Function	Calls	Unoptimized		Optimized		Hand-optimized	
		MSVC [cpp]	Intel [cpp]	MSVC [cpp]	Intel [cpp]	MSVC [cpp]	Intel [cpp]
Rect., Undist.	1	2.55	2.55	0.19	0.19	0.15	0.13
Scale	1	85.48	85.19	12.45	12.79	2.99	3.27
Time Corr.	1	368.29	368.00	289.40	294.20	96.40	96.04
WMI C. Aggr.	1	683.74	681.69	129.52	130.38	9.15	8.76
WMI WTA	h	21.68	21.71	4.12	3.43	1.19	1.12
AE Gray Conv.	1	11.64	11.67	3.92	3.99	3.82	3.57
Census Trans.	2	546.37	545.46	114.44	114.34	8.55	8.79
DSI C. Calc.	1	484.72	482.48	298.31	297.74	11.69	11.79
DSI C. Aggr.	h	720.60	721.85	128.67	128.42	5.92	5.90
DSI WTA	h	25.56	25.84	4.54	4.54	2.57	2.66
Disp. Merge	1	52.88	52.97	10.01	10.01	1.56	1.48
Total		3003.51	2999.41	995.57	1000.03	143.99	143.51

Table 1. Performance of the unoptimized and optimized implementation on an Intel i7 architecture using two compilers. Resolution is QVGA, disparity range is 30, time correlation history is 5 and scaling factor is 1 with data burst of 7kAE.

References

[1] C. Posch and D. Matolin and R. Wohlgenannt. A QVGA 143 dB Dynamic Range Frame-Free PWM Image Sensor With Lossless Pixel-Level Video Compression and Time-Domain CDS. *IEEE Journal of Solid-State Circuits*, 46(1):259–275, 2011.

[2] C. Zinner and M. Humenberger and K. Ambrosch and W. Kubinger. An Optimized Software-Based Implementation of a Census-Based Stereo Matching Algorithm. *Proceedings of the 4th International Symposium on Visual Computing (ISVC)*, 2008.

[3] M. Humenberger, C. Zinner, M. Weber, W. Kubinger, and M. Vincze. A fast stereo matching algorithm suitable for embedded real-time systems. *Computer Vision and Image Understanding Journal (Special Issue on Embedded Vision)*, 114(11):1080–1202, 2010.

[4] Intel Corporation. Intel Integrated Performance Primitives for Intel Architecture. 2007. Document Number: A70805-021US.

[5] J. Kramer. An on/off transient imager with event-driven, asynchronous read-out. *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2002.

[6] J. Kogler, C. Sulzbachner, and W. Kubinger. Bio-inspired stereo vision system with silicon retina imagers. In *Computer Vision Systems*, volume 5815 of *Lecture Notes in Computer Science*, pages 174–183. Springer Berlin / Heidelberg, 2009.

[7] C. Mead and M. Mahowald. A silicon model of early visual processing. *Neural Networks Journal*, 1, 1988.

[8] P. Lichtsteiner and C. Posch and T. Delbruck. A 128×128 120dB 30mW asynchronous vision sensor that responds to relative intensity change. *Proceedings of the IEEE International Solid-State Circuits Conference*, 2006.

[9] P. Lichtsteiner and C. Posch and T. Delbruck. A 128×128 120dB 15μs Latency Asynchronous Temporal Contrast Vision Sensor. *Proceedings of the IEEE International Solid-State Circuits Conference*, 2008.

[10] D. Scharstein, R. Szeliski, and R. Zabih. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Stereo and Multi-Baseline Vision, 2001. (SMBV 2001). Proceedings. IEEE Workshop on*, pages 131 –140, 2001.

[11] S. Schraml, P. Schön, and N. Milosevic. Smartcam for real-time stereo vision - address-event based embedded system. In *In: Ranchordas, Alpesh, Arajo, Helder and Vitri, Jordi (eds.) VISAPP 2007 - Proceedings of the Second International Conference on Computer Vision Theory and Applications*, volume 2, pages 466–471, Barcelona/Spain, March 2007.

[12] O. Schreer. *Stereoanalyse und Bildsynthese*. Springer Berlin / Heidelberg, 2005.

[13] J. Shi and C. Tomasi. Good features to track. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[14] M. Sivilotti. *Wiring consideration in analog VLSI systems with application to field programmable networks*. PhD thesis, California Institute of Technology, 1991.