INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 04, April 2024, pp: 2401-2406

SMART CAMERAS IN EMBEDDED SYSTEMS

Mula Likhitha[1], Mohan Babu C[2]

[1]Dept of electronics and Communication Enginerring SJC Institute Of Technology Chickaballapura, karnataka, India.

[2]Assistant professor Dept of Electronics and Communication Engineering SJC Institute Of Technology Chickaballapura, karnataka, India

ABSTRACT

A "smart camera" comprises a compact integration of a video camera with a computer vision system. This discussion initiates by outlining the primary distinctions between smart cameras and conventional smart vision systems. It elaborates on the architecture of a smart camera, highlighting the utilization of an onboard microprocessor and PLDs to enable the incorporation of image processing algorithms within the camera itself. The text introduces a static thresholding algorithm as an example, showcasing its capability to detect non-uniformity in the inspection target. Furthermore, it presents an application scenario of a multi-camera inspection system, wherein up to twenty smart cameras can be interconnected to a single host computer. Concluding remarks include insights into the potential technological and application-based advancements in the realm of smart cameras.

1. INTRODUCTION

Recent advancements in technology are facilitating the emergence of a new breed of intelligent cameras, marking a significant advancement in sophistication. Unlike conventional digital cameras that merely capture images, smart cameras have the capability to capture detailed descriptions of scenes and conduct analysis based on the content they perceive. These devices hold potential for diverse applications, including but not limited to human and animal detection, surveillance, motion analysis, and facial recognition.

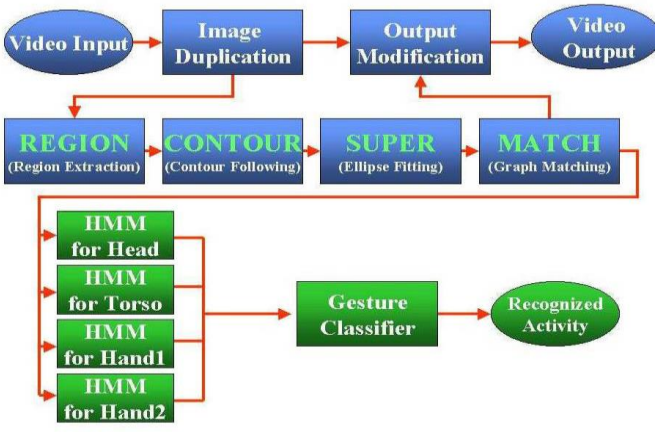
The demand for real-time performance in video processing is relentless. Smart cameras harness the power of very large-scale integration (VLSI) to deliver such analysis within a cost-effective, energy-efficient framework, equipped with ample memory capacity. Going beyond mere pixel processing and compression, these systems execute a broad spectrum of algorithms to derive meaningful insights from streaming video data.

2. ALGORITHMS FOR DETECTION AND RECOGNITION

Although various methods exist for real-time video analysis, our initial focus lies on human gesture recognition. This involves discerning actions such as walking, standing, or waving arms. Given the ongoing development in this area, our aim is to devise an embedded system capable of accommodating future algorithms while leveraging those tailored specifically for this purpose.

Our algorithms encompass both low-level and high-level processing. At the low level, they identify individual body parts and classify their movements in simplistic terms. Conversely, the high-level component, tailored to specific applications, utilizes this data to recognize actions of each body part and the overall activity of the person, considering contextual parameters.

The human detection and activity/gesture recognition algorithm comprises two primary components: Low level processing and high level processing.



INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 04, April 2024, pp: 2401-2406

A) Primary processing

The system acquires images from the video input, which may be uncompressed or compressed (such as MPEG and motion JPEG), and employs four distinct algorithms to detect and identify human body parts.

Region extraction: In the initial algorithm, the pixels of an image, as depicted , are converted into an M × N bitmap and background elements are removed. Subsequently, it identifies the skin area of the body part using a YUV color model with down sampled chrominance values

Contour tracing: Following region extraction, as depicted in the subsequent stage entails connecting the individual clusters of pixels to form contours that outline the regions geometrically. This algorithm employs a 3 × 3 filter to track the edge of the component in any of eight different directions

Elliptical fitting: In order to rectify distortions in image processing arising from various factors such as clothing, obstructing objects, or overlapping body parts, an algorithm is employed to fit ellipses to the pixel regions, as illustrated. This approach aims to simplify the attributes of body parts. By utilizing parametric surface approximations, the algorithm calculates geometric descriptors for segments including area, compactness (circularity), weak perspective invariants, and spatial relationships.

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 04, April 2024, pp: 2401-2406

Matching graphs: Each region extracted and modeled with ellipses corresponds to a node within a graphical representation of the human body. A piecewise quadratic Bayesian classifier utilizes the parameters of the ellipses to compute feature vectors, comprising binary and unary attributes. These feature vectors are then matched with those of body parts or meaningful combinations of parts, which are computed offline. To streamline the process, the algorithm commences with the face detection, which is typically the most straightforward.

Advanced processing:

The high-level processing module, adaptable to various applications, juxtaposes the motion sequence of each body part defined as a spatiotemporal series of feature vectors across multiple frames with known posture and gesture patterns. It employs multiple hidden Markov models concurrently to assess the overall activity of the body. Discrete HMMs are utilized, generating eight directional code words to assess upward, downward, leftward, rightward, and circular movements of each body part.

Human actions often entail intricate sequences of movements. Therefore, we amalgamate the motion pattern of each body part with the subsequent one to generate a new sequence. Through dynamic programming, we compute probabilities for both the original and amalgamated sequences to discern the person's actions. Intervals between gestures serve as cues for the commencement and culmination of discrete actions.

A quadratic Mahalanobis distance classifier amalgamates HMM outputs with varying weights to create reference models for diverse gestures. For instance, a pointing gesture might signify a command to "advance to the next slide" in a smart meeting environment or "open the window" in a smart vehicle.

Conversely, a smart security camera may interpret the gesture as suspicious or menacing.parts. High-level information obtained from one perspective activates recognition algorithms using the second cameras.

Soft-tissue reconstruction:

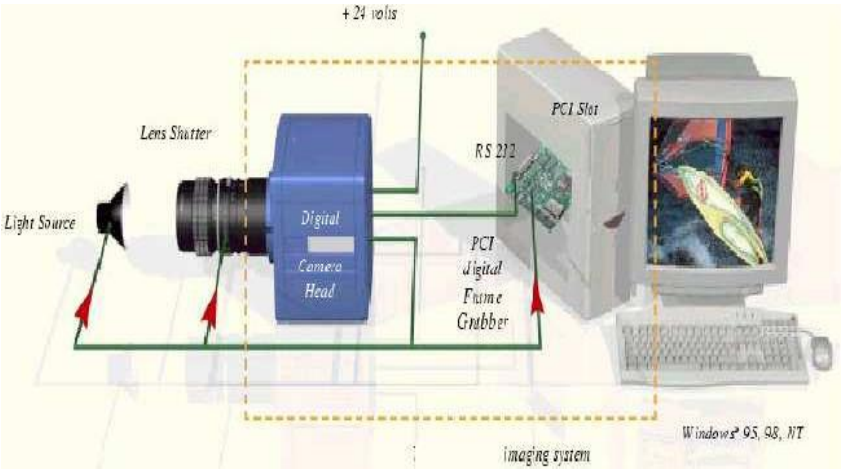
MatLab serves as a platform for developing our algorithms. While this technical computation and visualization environment operates significantly slower than embedded platform implementations, particularly crucial for real-time video processing, it enables us to transfer our MatLab implementation to C code running on a very long instruction word (VLIW) video processor. This facilitates architectural measurements on the application and necessary optimizations for designing a bespoke VLSI smart camera.

Specifications

During the development phase, we assess the algorithms based on accuracy and other conventional benchmarks. However, an embedded system must meet additional real-time requirements:

Frame rate: The system must process a specified number of frames per second to effectively analyze motion and yield meaningful results. The attainable frame rate depends on the algorithms employed and the computational capability of the platform, with some systems achieving exceptionally high rates.

Latency: The time taken to produce results for each frame is crucial, especially as smart cameras are likely to be integrated into closed-loop control systems. High latency can impede the timely initiation of events based on actions captured in the video feed.



Transitioning to an embedded platform also necessitates efficient memory utilization. In anticipation of highly integrated smart cameras, minimizing memory usage in the system is imperative to conserve chip area and reduce power consumption. Excessive memory usage often indicates inefficient implementation.

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

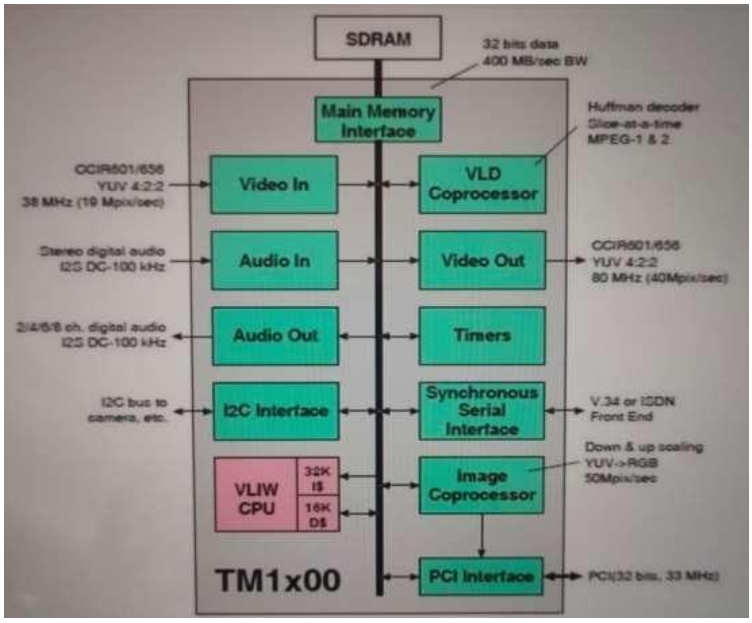
Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 04, April 2024, pp: 2401-2406

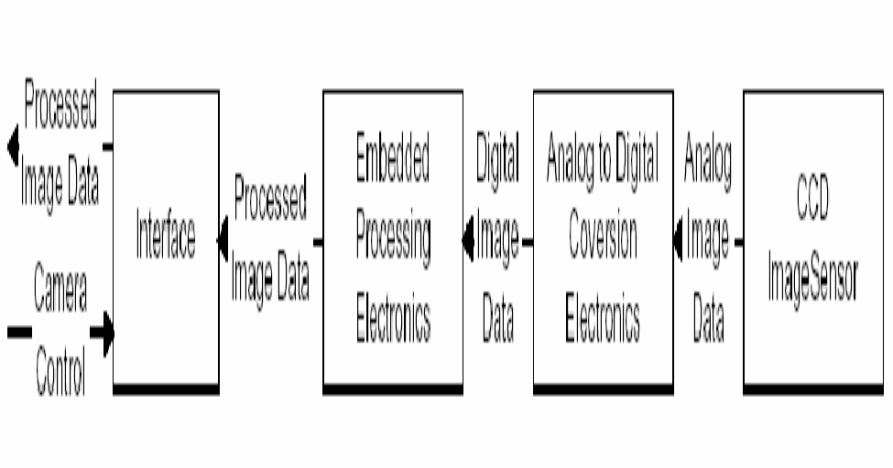
3. COMPONENTS

Our development strategy involves utilizing readily available components to process video in real-time from a standard source, debug algorithms and programs, and establish connections between multiple smart cameras in a networked system. The chosen video processor is the 100-MHz Philips TriMedia TM-1300, which boasts a 32-bit fixed- and floating-point processor with dedicated image coprocessor, a variable length decoder, an optimizing C/C++ compiler, integrated peripherals for VLIW concurrent real-time input/output, and a comprehensive set of application library functions, including support for MPEG, motion JPEG, and 2D text and graphics.



Testbed Architecture

Our testbed architecture, illustrated in Figure 3, incorporates two TriMedia boards linked to a host PC to facilitate programming support. Each PCI bus board is connected to a Hi8 camera providing NTSC composite video. Multiple boards can be integrated into a single computer for concurrent video operations. The utilization of a shared memory interface offers enhanced performance compared to networks typically utilized in VLSI cameras, enabling us to functionally implement and debug multiple-camera systems using actual video data.



Experiments and Optimizations

As data representation transitions to a more abstract form, the volume of input/output data decreases. However, predicting the change in required memory size becomes less straightforward due to complex relationships that can develop between abstract data. For instance, describing 100 ellipses using six single-precision, floating-point parameters necessitates only 2.4 Kbytes of memory, whereas storing information about two adjoining ellipses requires 10 Kbytes. Building on these initial experiments, we optimize our smart camera implementation by incorporating techniques to expedite video operations, such as substituting new algorithms better suited to real-time processing and employing TriMedia library routines in place of C-level code.

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

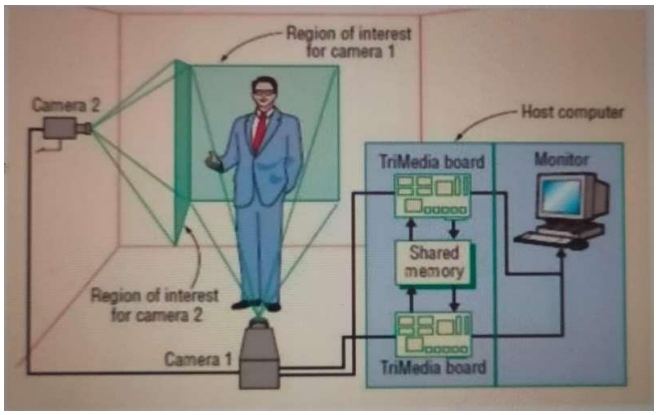
Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 04, April 2024, pp: 2401-2406

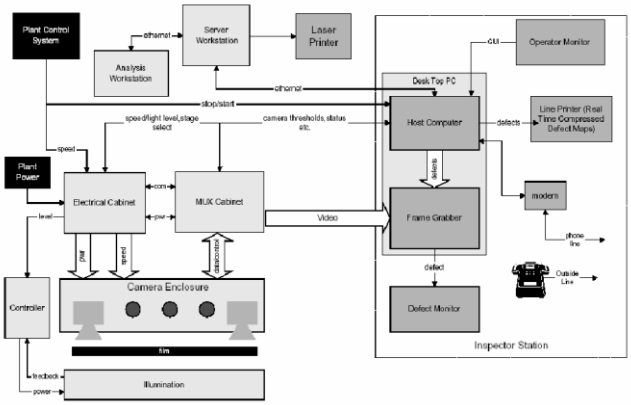
Algorithmic Changes

Initially, we employed a method involving fitting super ellipses (generalized ellipses) to contour points, which proved to be the most time-consuming step. Rather than focusing on optimizing the existing code, we opted to switch to a different algorithm. By replacing the original approach developed from principal component analysis with moment-based initialization, we streamlined the Levenberg-Marquardt fitting procedure, resulting in reduced execution time. Following the conversion of the original Matlab implementation to C, we conducted experiments to evaluate the effectiveness of the smart camera system and identify bottlenecks. The unoptimized code averaged 20.4 million cycles to process a single input frame, equivalent to a rate of 5 frames per second. Initially, we assessed the CPU times of each low-level processing step to pinpoint where the cycles were being consumed. Microsoft Visual C++ proved more suitable for this purpose than the TriMedia compiler, as it could track the running time of each function as well as its subfunctions' times. Figure 4a displays the distribution of processing times for the four body-part-detection algorithms, while Figure 4b depicts the memory characteristics of each low-level processing stage.



Control-to-Data Transformation

Expanding the processor's issue width can leverage the high degree of parallelism inherent in region extraction. Employing a processor with more functional units could consequently reduce processing time during this phase. However, contour following, which converts pixels into abstract forms like lines and ellipses, incurs even greater time consumption. Moreover, the algorithm operates serially, examining a small pixel window to sequentially trace the contour's boundary. At each step, it must determine the location of the contour's next pixel in a clockwise direction. While this method is accurate and intuitive, it offers limited instruction-level parallelism (ILP). To address this limitation, we evaluated all possible directions in parallel and consolidated the true/false results into a byte, serving as an index to access the boundary pixel in a lookup table. Additionally, we manipulated the control-flow structure of the algorithm to further enhance ILP. These optimizations doubled the speed of the contour-following stage.



4. OPTIMIZATION RESULTS AND CONCLUSION

The combination of these methods significantly enhances CPU performance for the application. Optimization elevates the program's frame rate from 5 to 31 frames per second, while latency decreases from about 340 to 40-60 milliseconds per frame. With the incorporation of HMMs and other high-level processing components, the program now operates at approximately 25 frames per second. Our board-level system represents a pivotal initial step in designing a highly integrated smart camera. Although the current system is directly applicable to certain applications, including security and medicine, a VLSI system will facilitate the development of high-volume embedded computing products. Given that

digital processors and memory rely on advanced small-feature fabrication, while sensors necessitate relatively large pixels for efficient light collection, designing the system as two chips housed within a multichip module is pragmatic.

Separating the sensor from the processor also aligns with architectural principles, considering the well-understood and straightforward interface between the sensor and computation engine. Leveraging existing sensor technology offers significant advantages over pixel-plane processors until they become more prevalent. Nevertheless, integrating special-purpose SIMD processors into the multiprocessor can prove beneficial for boundary analysis and other operations, while also conserving power, a critical consideration given the expense and effort involved in deploying multiple cameras, particularly in outdoor settings.

5. REFERENCES

[1] Advanced Imaging Europe Magazine: "The intelligent camera", October (2020) 12– 16.

[2] Smart Cameras vs. PC-based Machine Vision Systems, http://www.coreco.com/,(2022).

[3] Longbottom, D.: "Latest Developments in Sensor and Camera Technology", Alrad Instruments Ltd., White paper, http://www.ukiva.org/IPOTMV02Latest.pdf, (2023).

[4] Smart Cameras –" A complete vision system in a camera body", Vision Components, http://www.is.irl.cri.nz/products/smartcam. html.

[5] Lehotsky, D. A.: "Intelligent High Sensitivity CCD Line Scan Camera with embedded Image Processing algorithms", DALSA INC, (2023).

[6] Cavallaro, "Privacy in video surveillance," IEEE SignalProcessing Magazine, vol. 24, no. 2, pp. 168–169, 2007.

[7] M. Bramberger, A. Doblander, A. Maier, B. Rinner, andH. Schwabach, "Distributed embedded smart cameras forsurveillance applications," Computer, vol. 39, no. 2, pp. 68–75,2006.

[8] S. Fleck, Privacy sensitive video surveillance, Ph.D. thesis,Eberhard-Karls-Universitat T ¨ubingen, 2008. ¨

[9] S. Soro and W. Heinzelman, "A survey of visual sensornetworks," Advances in Multimedia, vol. 2009, Article ID640386, 21 pages, 2009.

[10] B. Rinner, T. Winkler, W. Schriebl, M. Quaritsch, and W. Wolf,"The evolution from single to pervasive smart cameras," inProceedings of the 2nd ACM/IEEE International Conference onDistributed Smart Cameras, p. 10, 2008.

[11] I. Mart´ınez-Ponte, X. Desurmont, J. Meessen, and J.-F.Delaigle, "Robust human face hiding ensuring privacy," inProceedings of the International Workshop on Image Analysis forMultimedia Interactive Services, p. 4, 2005.

[12] J. Schiff, M. Meingast, D. K. Mulligan, S. Sastry, and K.Goldberg, "Respectful cameras: detecting visual markers inreal-time to address privacy concerns," in Proceedings ofthe IEEE International Conference on Intelligent Robots andSystems, pp. 971–978, 2007.

[13] A. Chattopadhyay and T. E. Boult, "PrivacyCam: a privacypreserving camera using uCLinux on the blackfin DSP,"in Proceedings of the IEEE Computer Society Conference onComputer Vision and Pattern Recognition, pp. 1–8, 2007.

[14] T. Winkler and B. Rinner, "TrustCAM: security and privacy-protection for an embedded smart camera based on trustedcomputing," in Proceedings of the International Conference onAdvanced Video and Signal-Based Surveillance, p. 8, 2010.