Team: 03

An Intelligent Architecture Based on Field Programmable Gate Arrays Designed to Detect Moving Objects by Using Principal Component Analysis

Introduction

Field-Programmable Gate Arrays (FPGAs) have increasingly become a cornerstone in high-performance computing due to their reconfigurability and ability to perform parallel processing. These characteristics make FPGAs particularly suitable for tasks that require significant computational power and real-time processing capabilities. One such application is in the realm of image processing, where rapid and accurate analysis of image data is crucial.

In the context of image processing, Principal Component Analysis (PCA) is a widely used statistical technique that simplifies the complexity of data by transforming it into a set of orthogonal components, thereby reducing the dimensionality while preserving the variance. PCA is particularly useful for background subtraction and object detection in video streams, which are computationally intensive tasks.

The article explores an intelligent architecture based on FPGAs designed to detect moving objects using PCA. The focus is on leveraging the inherent parallel processing power of FPGAs to implement the PCA algorithm efficiently. The implementation aims to address the challenges associated with high-speed image data processing, such as managing large volumes of data and performing complex mathematical operations in real-time.

Principal Component Analysis (PCA) on FPGA

Overview of PCA

Principal Component Analysis (PCA) is a statistical technique used for dimensionality reduction while preserving variance. It transforms data into a new coordinate system where the greatest variances lie on the first few principal components. This process is critical in image processing for tasks such as background subtraction and object detection.

Implementation Steps

The FPGA-based implementation of PCA involves several computational stages optimized for parallel processing, leveraging the inherent capabilities of FPGAs.

1. **Computation of the Covariance Matrix:** The covariance matrix, representing variances and covariances between data dimensions, is computed using parallel operations on the FPGA. This parallelization accelerates the calculation process significantly.
2. **Matrix Diagonalization Using the Jacobi Method:** Matrix diagonalization to find eigenvalues and eigenvectors is performed using the Jacobi method. The FPGA executes multiple Jacobi rotations simultaneously, leveraging its parallel processing capabilities to reduce the computation time drastically.
3. **Projection of Images onto the PCA Subspace:** Projecting images onto the PCA subspace involves transforming the original image data using the computed eigenvectors. This step requires matrix multiplications that are efficiently handled by the FPGA's parallel processing units, enabling real-time processing.

FPGA Hardware Architecture

The architecture consists of several interconnected modules as shown in Fig. 1, each performing specific functions essential for the overall system operation.

1. **CMOS Sensor Controller**: The CMOS sensor controller manages image acquisition from the CMOS sensor, ensuring correct timing and formatting for further processing.
2. **Image Capture Controller**: This module allows selection of specific areas of interest within captured images, providing flexibility in focusing computational resources on relevant portions.
3. **External Memory Controller**: The external memory controller interfaces with SDRAM to store captured images and intermediate results, managing memory efficiently to ensure quick data access.
4. **Communications Controller with PC**: This controller facilitates communication between the FPGA and a connected PC, handling command and data transmission for remote control and monitoring.
5. **Head Controller**: The head controller synchronizes operations of all other modules, ensuring tasks are executed in the correct sequence and timing.

6. **PCA Algorithm Block**: The PCA algorithm block is the system's core, implementing all necessary computations for PCA and motion detection. It includes a covariance calculation unit, eigenvalue/eigenvector computation unit, and projection unit.

Motion Detection Algorithm

The motion detection algorithm utilizes the PCA-based background model to identify moving objects. The following steps outline the process:

1. **Calculating the Mean of Captured Images:** Multiple images are captured, and their pixel values are averaged to compute the mean image, serving as the background model.
2. **Computing Eigenvectors and Eigenvalues:** The covariance matrix of the mean-subtracted images is computed, and the Jacobi method is applied to find eigenvalues and eigenvectors.
3. **Transforming Input Images:** New input images are projected onto the PCA subspace using the computed eigenvectors, allowing effective comparison with the background model.
4. **Dynamic Thresholding:** The difference between the transformed input image and the background model is dynamically thresholded to detect moving objects. The threshold adapts based on observed differences, ensuring robust motion detection under varying conditions.

Performance and Computational Benefits

The FPGA-based implementation offers numerous advantages over traditional CPU-based systems:

1. **High Processing Speed:** FPGAs are inherently designed for parallel processing, allowing the system to achieve frame processing rates of up to 120 frames per second. This speed is crucial for real-time applications where timely processing of each frame is essential.
2. **Real-Time Performance:** The efficient handling of computationally intensive tasks, such as matrix multiplications and eigenvector calculations, ensures real-time performance. This capability is vital for applications like motion detection, where delays can render the system ineffective.
3. **Customization and Flexibility:** FPGAs provide the ability to customize hardware specifically for PCA and motion detection algorithms. This customization optimizes

performance and resource utilization, ensuring that the system is both powerful and efficient.

4. **Low Latency:** The embedded nature of the FPGA-based system reduces latency, providing immediate processing and response times. This low latency is critical for applications requiring rapid feedback and action.

Results

The system achieves high processing speeds, with up to 120 frames per second, and demonstrates remarkable accuracy in detecting moving objects in various lighting conditions. The FPGA implementation of the PCA algorithm and the associated hardware optimizations enable efficient and real-time image processing and object detection. The system's performance is validated through experiments using an intelligent camera setup, showcasing its potential for real-time applications in various fields requiring object detection and tracking.

Conclusion

The article effectively demonstrates the significant advantages of FPGA technology for implementing complex algorithms like PCA. The parallel processing capabilities and customizability of FPGAs offer substantial computational benefits, making them ideal for high-speed, real-time applications in image processing and motion detection. This research highlights the potential of FPGAs to advance the field of computer vision by providing efficient, reliable, and scalable solutions.