STUDY AND ANALYSIS OF REMOTE SENSING DATA PARALLEL PROCESSING

Chenyang Li [1,2,3], Guoqing Zhou[1,2,3,4]*, Xiang Zhou[2,3,4], Dequan Liu [2,3,4]

[1] School of Marine Science and Technology Tianjin University, Tianjin, 300072 China. gzhou@glut.edu.cn (G.Z.);
[1]lichenyang_1008@tju.edu.cn (C. L.)
[2] The Center for Remote Sensing, Tianjin University, Tianjin, 300072 China.(2996868199@qq.copm)
[3] GuangXi Key Laboratory for Spatial Information and Geomatics, Guilin University of Technology, Guilin,Guangxi, 541004, China
[4] School of Microelectronics Tianjin University, Tianjin, 300072 China. gzhou@glut.edu.cn (G.Z.);
[4]zqx0711@tju.edu.cn (X.Z.); 017232001@tju.edu.cn (D.L.)

Commission VI, WG VI/4

KEY WORDS: Data processing, Parallel, FPGA, Quaternion, LU matrix factorization, Fast matrix multiplication

ABSTRACT:

This paper analyzes a varieties of procedure of remote sensing data processing, and explores the common mathematical models, common algorithm models, and public function processing units of data processing shared by different tasks or even different parts within an individual task. Public modules are established to improve the parallelism of remote sensing data processing based on FPGA, which has excellent parallel processing performance. In addition, in order to reduce the resource consumption and increase the calculation efficiency of the designed FPGA program, the method of avoiding floating-point arithmetic and division operation in FPGA programming are discussed in this paper. There are a large number of common calculation modules between different tasks, such as the rotation matrix calculation module in attitude solution, geometric correction, and orthorectification task. Image preprocessing, feature information extraction, image threshold separation, and connected region markers are all common processing modules for a target detection task. In the same task, there is also a common calculation module. When using the FPGA design program, the power series of 2 can be used to convert the floating-point operation to fixed-point operation withan acceptable precision. A similar approach can transform the division operation into multiplication and shift operations, thereby improve the computational performance of FPGA programming.

* Corresponding author

1. INTRODUCTION

With the development of new Earth observation satellites, the spatial, spectral and temporal resolution of satellite sensors are continuously improved. The earth observation data obtained by remote sensing, has grown dramatically. It is an imperative demand to increase data processing accuracy and speed for space application. When processing massive remote sensing data, the traditional uni-processor serial program is affected by the parallelism of data processing, mainly because the serious drawbacks in real-time data processing. Therefore, the application of remote sensing images suffers from limitation, especially in real-time disaster monitoring and military applications. In view of the above problems, it makes great sense to research the parallel data processing of high-performance remote sensing data clusters. With the development of MEMS (Micro-electromechanical Systems) and sensing technology in the past several decades, the application of field programmable gate array (FPGA) technology has greatly facilitated the data processing efficiency. Besides, FPGA technology has also been widely used in parallel data processing of remote sensing. For application of FPGA in parallel processing of remote sensing data, in order to optimize the parallelism of the designed algorithm, adequate attention should be given to analysis of feasibility and design strategy. For feasibility analysis, it is necessary to clarify the general task flow of remote sensing data processing, so that the associated attributes and inheritance relationships between different data processing tasks could be identified. In addition, the detailed algorithm implementation process for each task provides a method to establish the parallelism between the algorithms of each task, and thereby reduce the required calculation time. With appropriate design strategy, not only calculation but also resource could be optimized by means of column methods as long as solution accuracy is guaranteed.

Considering these two concerns, the current paper is mainly aimed at an improved performance in the parallel data processing of remote sensing based on FPGA.

2. ANALYSIS OF PARALLEL DATA PROCESSING OF REMOTE SENSING

2.1 General procedure for remote sensing data processing

The flow chart of data processing in common remote sensing technology is shown in Fig.1:

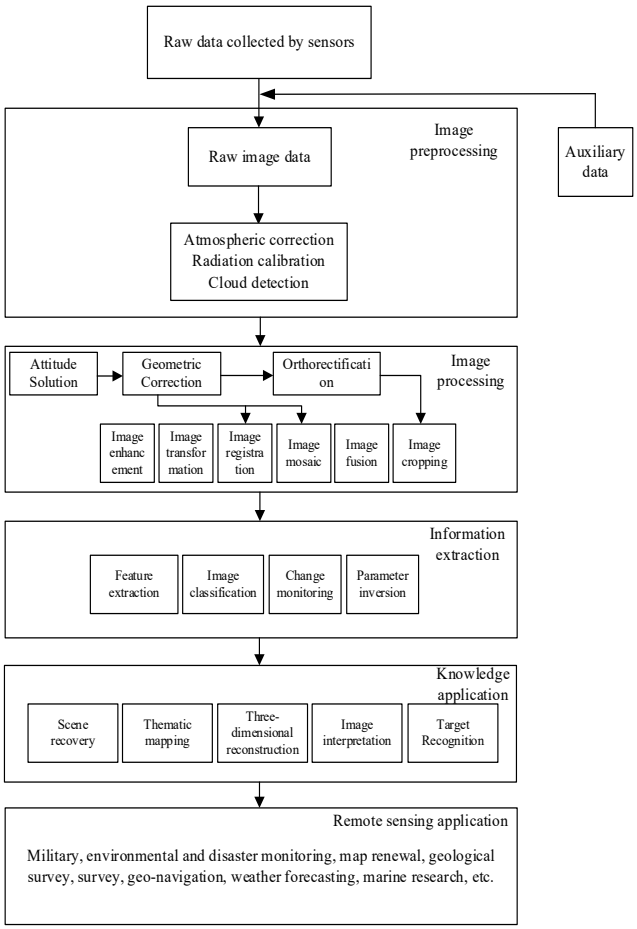

Figure 1. General procedure for remote sensing data processing(Gu et al., 2016)

As can be seen from the flow chart, the data processing can be roughly divided into several stages according to task types, including image preprocessing stage, image processing stage, information extraction stage, knowledge application stage, and remote sensing application stage. After remote sensing images are preprocessed by atmospheric and radiation correction, attitude solution geometric correction and orthorectification are the key to image information extraction and remote sensing applications. and finally, the final remote sensing products required by the user can be obtained through information extraction, image classification, and inversion of the parameters for the region of interest.

2.2 Data processing flow and algorithm analysis of different task

2.2.1 Radiation calibration data processing and algorithm analysis: The data processing of the radiometric calibration mainly includes, acquisition of original images under different brightness levels by the light homogenizing device, determination of the uniformity and oversaturation of the image. The non-uniformity correction coefficient is calculated by two-point method, which provides an approach to obtain a uniform image. With the calculated radiation calibration coefficient by two-point method, the radiation corrected image is finally achieved.

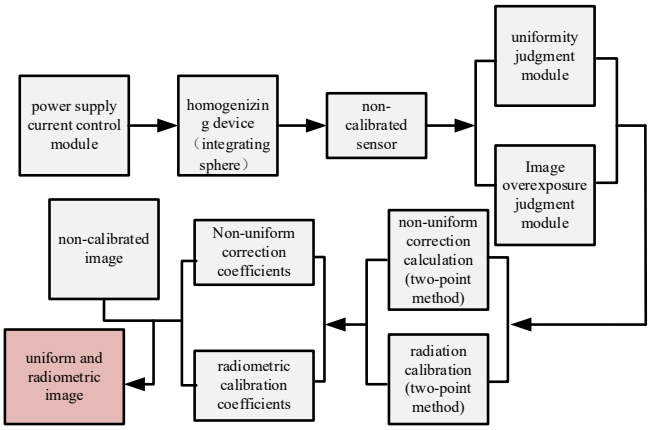

Figure 2. Radiation calibration data processing flow chart

2.2.2 Cloud detection data processing flow and algorithm analysis: the next step After the radiation pre-processing is cloud detection on the image. In this task, the images with higher cloud coverage are directly removed, in order to reduce the number of images for processing and improve the efficiency of other subsequent remote sensing tasks. The main data processing flow of the cloud detection includes image feature information extraction, image threshold separation, and labelling of connected components. The data processing flow chart is as follows:

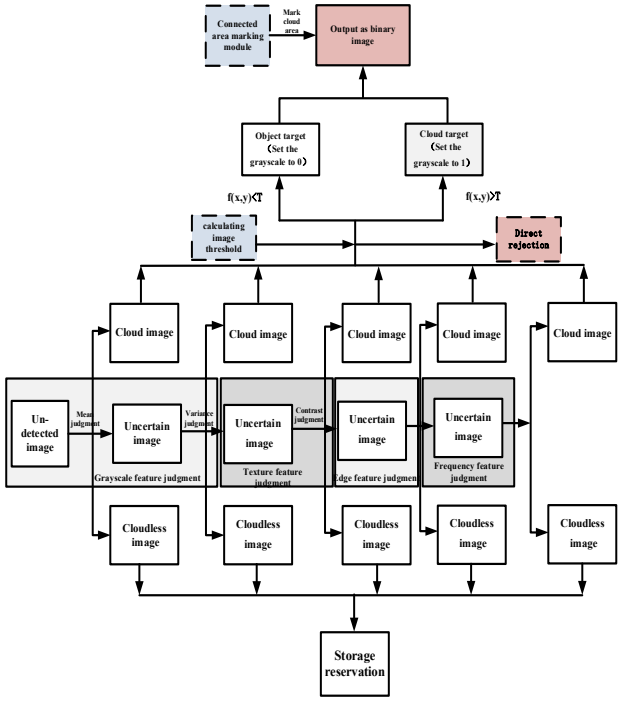

Figure 3. Cloud detection data processing flow chart

2.2.3 Analysis of data procee and algorithm for Satellite attitude calculation: There are two tasks for satellite attitude calculation: (1) relative attitude calculation; (2) absolute attitude calculation.

The relative calculation solution includes detection and matching of local features, mismatch elimination and relative attitude parameter calculation (Huang et al. 2018).

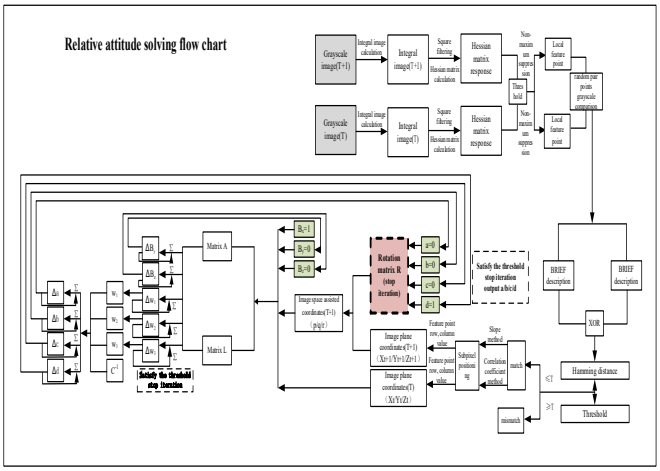

Figure 4. Relative attitude calculation flow chart

In the absolute attitude calculation, the algorithms includes detection and matching of local features; mismatch elimination; absolute attitude parameter calculation (Huang et al. 2018).

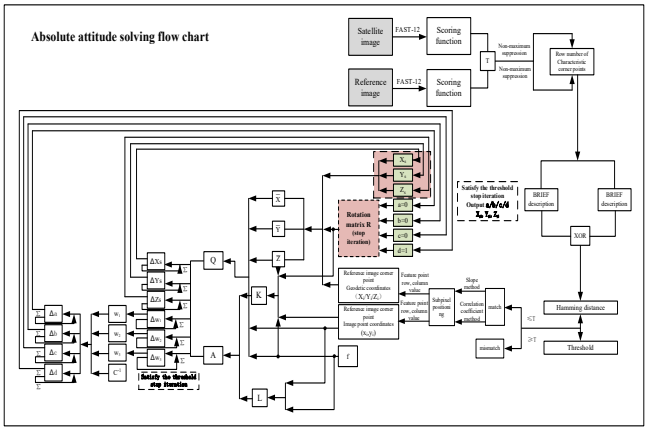

Figure 5. Absolute attitude calculation flow chart

By analyzing the task processing flow of the relative attitude calculation and the absolute attitude calculation, it can be found that the two tasks share common in the method of elimination of mismatched feature points and use quaternion to calculate the rotation matrix. Therefore, parallel optimization could be performed with common algorithms of relative attitude calculation and absolute attitude calculation.

2.2.4 Satellite orthorectification data processing flow and algorithm analysis:

The calculation process of this task mainly includes: establishment of rigorous geometric processing model, and obtainment of the pixel coordinates of "virtual control point" and geodetic coordinates by the rigorous geometric processing model. After Establishment of the Rational Function Model(RFM), relevant model parameter are determined with according to least square method. then RFM is simplified with optimization in the irrelevance of RFM parameters, leading to reduced amount of model parameters; the orthoimage position and the number of rows and columns are obtained through strict geometric processing model; and subsequently, the orthophoto pixel coordinates and geodetic coordinates could be calculated. With substitution of orthophoto pixel coordinates and geodetic coordinates into the optimized RFM model, the pixel

coordinates of the original image could be derived, and its grey value could be determined according to the bilinear interpolation method. The determined grey level is further employed tothe orthophoto pixel, and orthorectification is completed (Zhang et al. 2018). The main flow chart for data processing is shown in Fig. 6:

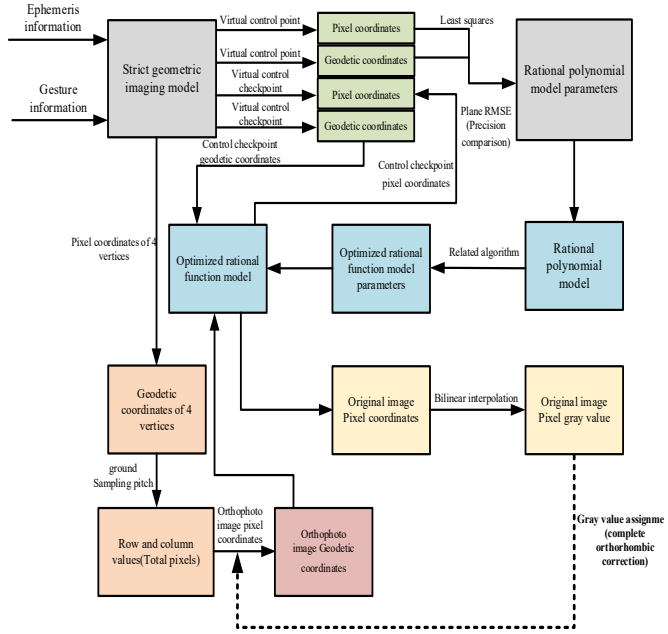

Figure 6. Data processing flow chart for orthophoto correction without ground control point

2.2.5 Target detection data processing flow and algorithm analysis: with the example of ship target detection, the main flow chart of data processing includes, image data preprocessing (filtering, denoising, enhancement), target feature extraction and target discrimination, connected area labelling. The main calculation process is shown in Fig. 7:

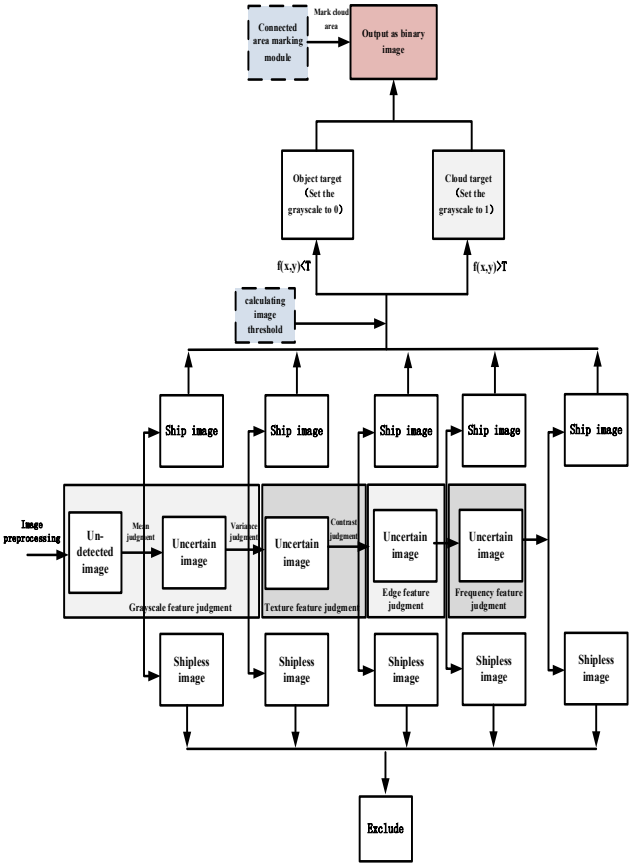

Figure 7. Ship detection data processing flow chart

Aaccording to the above analysis, the common processing modules between different tasks are:

The image gray value calculation module in radiometric calibration part is also applicable to other target detection task processes such as cloud detection, ship monitoring, etc., and can be mainly used as image enhancement.

(2) The common processing modules among attitude calculation, geometric correction and orthorectification mainly include diagonal element calculation module with the outer quaternion method instead of Euler angle, and the least squares solution module.

(3) The cloud detection and ship target detection tasks share more common processing modules, including filtering and denoising module in image preprocessing, image threshold segmentation, binarization module, connected area labelling module, and redundant data elimination module.

There are common processing modules within an individual tasks, including:

In radiation calibration part: The two-point method module is used to calculate not only the non-uniformity correction coefficients but also the radiation calibration coefficients.

In attitude calculation part, the absolute attitude and relative attitude calculation involve common module in the eigenvalue description sub-calculation module and the mismatch elimination.

3. REMOTE SENSING DATA PARALLEL PROCESSING DESIGN METHOD BASED ON FPGA

The remote sensing data processing based on FPGA is programmed by HDL, and the tasks of remote sensing data processing are implemented in FPGA. Programming with FPGA needs to improve calculation accuracy as well as efficiency, and reduce resource consumption, so an optimization in processing algorithm for original data is achieved, to provide more convenience for FPGA programming. Remote sensing data processing consists of multiple tasks. Each task consists of specific algorithm modules. Different algorithm modules have specific implementation steps. Each implementation step includes multiple computational units. The establishment of a common calculation unit can facilitate the application of parallel algorithm for different calculation modules, and thereby the parallel processing efficiency is improved for each task.

3.1 Remote Sensing Data Processing Design Flow Based on FPGA

The design and implementation of the FPGA-based system for remote sensing data processing task usually adopts a top-down hierarchical design strategy in the development process. The task is the top-level hierarchy, and each calculation module of the task is a second-level hierarchical module, and then the solution steps in each calculation module are further refined into elements in third level of hierarchy until the top-level design task is fully decomposed into the basic FPGA computing unit or IP core.

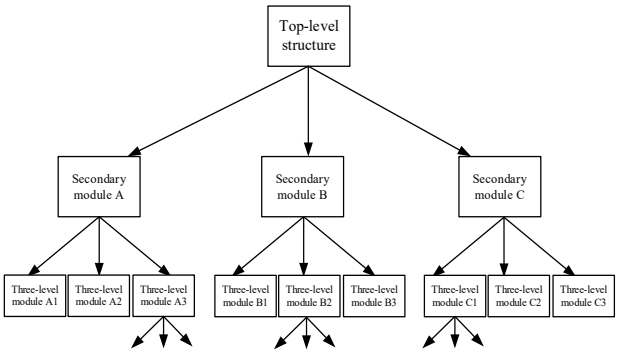

Figure 8.Remote sensing data processing design strategy based on FPGA (Geng Tian, Wenbo Xu. 2008.)

Consequently, the remote sensing data processing can be roughly divided into the first, second and third layer of operation according to the hierarchy.

The top layer is a task layer, which mainly includes radiation calibration, cloud detection, satellite attitude calculation, geometric correction, orthorectification, target recognition (flood, ship, etc.).

The second-level layer is the algorithm layer, mainly including gray mean, gray variance, gray histogram, image filtering (mean filtering, median filtering, Gaussian filtering, square filtering, wavelet transform, etc.), feature detection and matching (SURF detection, FAST corner detection, etc.), feature extraction (gray features, shape features, texture features, edge features, frequency features), threshold separation (global threshold segmentation, local pose adaptation threshold segmentation), image markers (connected domain markers) , interpolation calculation (Lagrangian interpolation, SLERP interpolation, bilinear interpolation), least squares calculation (recursive least squares), etc.

The third layer is a computing unit layer, and mainly includes: arithmetic operations, logical operations, bitwise operations, relational operations, equality operations, shift operations, conditional operations, and the like. Among them, Among them, there are more applications unit includes: addition, subtraction, multiplication, division, logarithm, and matrix operations. There are a large number of matrix operations in remote sensing data processing algorithms, including computational content: matrix addition, matrix multiplication (matrix fast multiplication), matrix inversion, matrix hybrid product, matrix determinant, etc. The parallel data processing of remote sensing based on FPGA can obtain the algorithm relationship between different tasks through hierarchical analysis of the data processing algorithm flow of each task, and then plans the parallelism of remote sensing task.

3.2 Public Model of Remote Sensing Data Processing

To study the parallel data processing of remote sensing based on FPGA, it is necessary to dwell on the relationship between each task, each algorithm module and each computing unit, and design common modules according to common mathematical models, common typical algorithms and public functions among different tasks, in order to improve the efficiency of remote sensing data processing. When designing common mathematical models, common typical algorithms and public functions, due to the characteristics of FPGA programming, it is necessary to make corresponding improvements in remote sensing data processing, in terms of precision, efficiency and resource consumption.

3.2.1 Public mathematical model of remote sensing data processing based on FPGA: When calculating the angle of satellite attitude based on FPGA, the traditional Euler angle model consists of a complex rotation matrix structure, which requires a large number of trigonometric functions. The calculation consumes a lot of hardware resources and the calculation of the iteration times is unfavorable to the real-time processing of remote sensing data. In addition, for reverse derivation of Euler angle with rotation matrix, the non-uniqueness in calculation result leads to convergence problem of the calculation. as a result, in order to facilitate the application FPGA to attitude calculation, a quaternion model was introduced to replace the Euler angle model.

Quaternion model:

The quaternion is a hypercomplex number in the form of $q = d + ia + jb + kc$ .Where a, b, c represented an arbitrary real number. i, j, k represented imaginary unit and satisfied $i^2 = j^2 = k^2 = -1$ and $jk = -kj = i$ and $ki = -ik = j$ and $ij = -ji = k$. If d, a, b, c are satisfied $d^2 + a^2 + b^2 + c^2 = 1$, its belongs to a unit quaternion(Diebel J, 2006).

According to the quaternion definition, the rotation matrix of the coordinate system C_a could be transformed to the coordinate system C_b indicated by R.

$$R = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} = \begin{bmatrix} d^2 + a^2 - b^2 - c^2 & 2(ab - cd) & 2(ac + bd) \\ 2(ab + cd) & d^2 - a^2 + b^2 - c^2 & 2(bc - ad) \\ 2(ac - bd) & 2(bc + ad) & d^2 - a^2 - b^2 + c^2 \end{bmatrix} \quad (1)$$

Where R is the quaternion expression of the rotation matrix, which can describe the rotation with an arbitrary angle of the rigid body. This matrix is identical to the orthogonal traditional rotation matrix used in photogrammetry.

When describing the relationship between the coordinate system C_a and C_b, the expression with the Euler angle is:

$$C_a \xrightarrow{Lz(\kappa)} \circ \xrightarrow{Ly(\omega)} \circ \xrightarrow{Lx(\varphi)} C_b \quad (2)$$

In the equation, each rotation of the coordinate axis can be represented by a quaternion:

$$q_z = \cos(\kappa/2) + 0i + 0j + \sin(\kappa/2)k \quad (3)$$

$$q_y = \cos(\omega/2) + 0i + \sin(\omega/2)j + 0k \quad (4)$$

$$q_x = \cos(\varphi/2) + \sin(\varphi/2)i + 0j + 0k \quad (5)$$

The virtual quaternion transformed from C_a to C_b coordinates is:

$$qb_a = q_z q_y q_x \quad (6)$$

Expansion of the above equation gives the quaternion for the Eu ler angle:

$$d = \cos(\varphi/2)\cos(\omega/2)\cos(\kappa/2) + \sin(\varphi/2)\sin(\omega/2)\sin(\kappa/2) \quad (7)$$

$$a = \sin(\varphi/2)\cos(\omega/2)\cos(\kappa/2) - \cos(\varphi/2)\sin(\omega/2)\sin(\kappa/2) \quad (8)$$

$$b = \cos(\varphi/2)\sin(\omega/2)\cos(\kappa/2) + \sin(\varphi/2)\cos(\omega/2)\sin(\kappa/2) \quad (9)$$

$$c = \cos(\varphi/2)\cos(\omega/2)\sin(\kappa/2) - \sin(\varphi/2)\sin(\omega/2)\cos(\kappa/2) \quad (10)$$

The Euler anglecould be calculated by quaternion:

$$\sin\omega = 2(db - ac) \quad (11)$$

$$\tan\varphi = \frac{2(da + bc)}{1 - 2(a^2 + b^2)} \quad (12)$$

$$\tan\kappa = \frac{2(dc - ab)}{1 - 2(b^2 + c^2)} \quad (13)$$

$$\varphi = \arctan(-R_{13}/R_{33}) \quad (14)$$

$$\omega = \arcsin(-R_{23}) \quad (15)$$

$$\kappa = \arctan(R_{21}/R_{22}) \quad (16)$$

The use of quaternions instead of Euler angles greatly simplifies the calculation of angles in attitude calculation, geometric correction and orthorectification.

When designing the rotation matrix model using FPGA, the calculation the diagonal elements in R involves the same variables, for example R12 and R21 have the same variables ab and cd; R13 and R31 have the same variables ac and bd; R23 and R32 have the same variables bc and ad. As a result, the FPGA only needs a single calculation for these variables during calculating of R, which reduces the use of floating-point multipliers and improves data processing parallelism.

3.2.2 Common computing unit for data parallel processing based on FPGA:

(1) Matrix inversion module - LU matrix decomposition:

Matrix inversion operations is needed in attitude calculation, geometric correction, or orthorectification tasks. In the FPGA-based remote sensing processing, if the inversion is directly performed by the adjoining matrix method, more hardware resources are consumed, and the real-time performance is inferior. therefore, it could not provide desired real-time performance. and, it is necessary to find an alternative for implementing matrix inversion in the FPGA.

The essence of the LU decomposition algorithm refers to decomposition of the large matrix into several small matrices, so thatstandard LU could be applied to the decomposed small matrices. It has the same inversion process for matrices with different size. Here, a 5×5 matrix is taken as an example. First, B is the inverse matrix to be solved, and it is partitioned in the following way:

$$B = \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} & b_{15} \\ b_{21} & b_{22} & b_{23} & b_{24} & b_{25} \\ b_{31} & b_{32} & b_{33} & b_{34} & b_{35} \\ b_{41} & b_{42} & b_{43} & b_{44} & b_{45} \\ b_{51} & b_{52} & b_{53} & b_{54} & b_{55} \end{bmatrix} = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \quad (17)$$

where, $B_{11} = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix}$

$$B_{12} = \begin{bmatrix} b_{14} & b_{15} \\ b_{24} & b_{25} \\ b_{34} & b_{35} \end{bmatrix} \quad B_{21} = \begin{bmatrix} b_{41} & b_{51} \\ b_{42} & b_{52} \\ b_{43} & b_{53} \end{bmatrix}^T \quad B_{22} = \begin{bmatrix} b_{44} & b_{45} \\ b_{54} & b_{55} \end{bmatrix}$$

According to the definition of the LU decomposition algorithm, one can obtain:

$$\begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ 0 & U_{22} \end{bmatrix} \quad (18)$$

where, $B_{11} = L_{11}U_{11}$, $B_{12} = L_{11}U_{12}$, $B_{21} = L_{21}U_{11}$,

$B_{22} = L_{21}U_{12} + L_{22}U_{22}$

$$\begin{cases} L_{11} = \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix} \\ U_{11} = \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix} \end{cases} \quad (19)$$

where, $u_{11} = b_{11}$; $u_{12} = b_{12}$; $u_{13} = b_{13}$; $l_{21} = b_{21}/b_{11}$

$l_{31} = b_{31}/b_{11}$; $u_{22} = b_{22} - l_{21}u_{12}$; $u_{23} = b_{23} - l_{21}u_{13}$;

$l_{32} = (b_{32} - l_{31}u_{12})/u_{22}$; $u_{33} = b_{33} - l_{31}u_{13} - l_{32}u_{23}$

After calculating L_{11} and U_{11}:

$$\begin{cases} U_{12} = L_{11}^{-1}B_{12} \\ L_{21} = B_{21}U_{11}^{-1} \end{cases} \quad (20)$$

Make

$$B'_{22} = B_{22} - L_{21}U_{12} = \begin{bmatrix} b'_{11} & b'_{12} \\ b'_{21} & b'_{22} \end{bmatrix} = L_{22}U_{22} \quad (21)$$

L_{22} and U_{22} could be written as:

$$L_{22} = \begin{bmatrix} 1 & 0 \\ l'_{21} & 1 \end{bmatrix}; U_{22} = \begin{bmatrix} u'_{11} & u'_{12} \\ 0 & u'_{22} \end{bmatrix} \quad (22)$$

where, $u'_{11} = b'_{11}$; $u'_{12} = b'_{12}$; $l'_{12} = b'_{21}/u'_{11}$;

$u'_{22} = b'_{22} - l'_{21}u'_{12}$

Therefore,

$$B^{-1} = \begin{bmatrix} U_{11} & U_{12} \\ 0 & U_{22} \end{bmatrix}^{-1} \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix}^{-1} \quad (23)$$

where,

$$\begin{bmatrix} U_{11} & U_{12} \\ 0 & U_{22} \end{bmatrix}^{-1} = \begin{bmatrix} U_{11}^{-1} & M_{12}^{-1} \\ 0 & U_{22}^{-1} \end{bmatrix} \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix}^{-1} = \begin{bmatrix} L_{11}^{-1} & 0 \\ N_{21}^{-1} & L_{22}^{-1} \end{bmatrix}$$

$M_{12}^{-1} = -U_{11}^{-1}U_{12}U_{22}^{-1}$ $\quad$ $N_{21}^{-1} = -L_{22}^{-1}L_{21}L_{11}^{-1}$

According to the equations (17)-(23), the flow of the LU decomposition algorithm is shown in Figure 9. The calculation process is as follows:

1) B_{11} is decomposed using the LU module to obtain L_{11} and U_{11};
2) $L_{11}$and U_{11} are inverted and substituted into the equation (20) to get U_{12} and U_{21};
3) U_{12} and L_{21} are substituted into equation (21) to get B'_{22} ;
4) B'_{22} is decomposed using the LU module to obtain L_{22} and U_{22};
5) L_{11}, L_{21}, L_{22} and U_{11}, U_{12}, U_{22} are substituted into equation (23) to obtain B^{-1}.

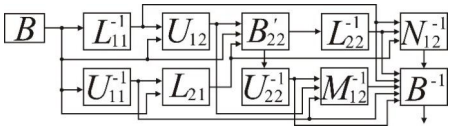

Figure 9. Flow chart of LU decomposition algorithm

(2) Module for matrix multiplication

The time complexity of sequentially performing matrix multiplication is $O(n^3)$.Therefore, as the matrix dimensions increase, the time required to perform matrix sequential multiplication increases dramatically.

In order to efficiently obtain the result of matrix multiplication, hardware acceleration of matrix multiplication can be performed by FPGA through the parallel structure of matrix multiplication. The parallel structure of matrix multiplication is mainly composed of mutually independent multiply accumulate unit PE. Since each PE does not need data exchange and communication, the parallel structure of the matrix multiplication has good scalability. With sufficient hardware resources, matrix multiplication of arbitrary dimensions can theoretically be realized.

Taking A×B=C as an example, the parallel computation of matrix multiplication is described. The size of the matrix A, B, and C is 39×39. As shown in Figure 10, A0101~A3939 and B0101~B3939 are elements in matrix A and matrix B, respectively; C0101~C3939 are elements in matrix C. In order to ensure the correct calculation result, the elements in the matrix A and the matrix B need to be sent to the corresponding PE processing unit in parallel simultaneously, and the matrix A and the matrix B sends data to the corresponding PE processing unit, according to the column priority and the row priority respectively.

A0101 is transported to the PE processing unit of the first row (ie PE0101~PE0139), A0201 is transported in the PE processing unit of the second row (ie PE0201~PE0239), and so on, A3901 is transported in the PE processing unit of the 39th row (ie PE3901~PE3939); B0101 is transported to the PE processing unit of the first column (ie PE0101~PE3901), and so on, B0139 is transported to the PE processing unit of the 39th column (ie PE0139~PE3939) at time t. After the above data completes the multiplication and accumulation operation, the second column data of the matrix A and the second row of the matrix B are sent to the corresponding PE processing unit for multiplication and accumulation operations in a similar manner, and the elements C0101 to C3939 in the matrix C are updated. When the elements of the last column of matrix A and the elements of the last row of matrix B complete the multiplication and accumulation operation according to the above procedure, the final matrix C is obtained. From the above analysis. the time complexity of matrix multiplication is reduced from the sequential execution of to the parallel execution of (Wu, 2011).

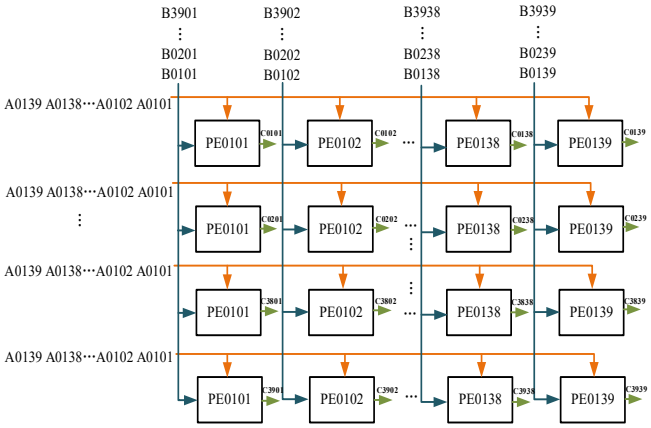

Figure 10. Parallel structure of matrix multiplication

3.3 Optimization Method of Parallel Remote Sensing Data Processing Based on FPGA

3.3.1 Method of avoiding floating point arithmetic

FPGA hardware programs has its distinctive programming and implementation methods. The algorithm is mainly divided into fixed-point operation and floating-point operation. Since floating-point operations consume more computer resources, in actual applications, fixed-point operations are preferred to reduce resource consumption. The commonly used method to transform floating point into fixed point is directly shifting the floating point data, and the floating point data becomes fixed point data after the shift. After the calculation is completed, the result is reversely shifted and converted into floating point data. E.g:

$$M = 0.41a + 0.62b + 0.26c \tag{24}$$

Expand the floating point number by 2^7 power, taking the integer part to calculate the fixed point number, and finally reducing the result by 2^7 power, the above equation is optimized as:

$$M = (52a + 79b + 33c) >> 7 \tag{25}$$

Alternatively, the floating point number is approximated by a power series of 2, and the floating point operation is converted into a fixed point shift calculation after the approximation, as:

$$\mathrm{X} = -0.862a^2 \approx -0.875a^2 = -a^2 + \frac{a^2}{8} \tag{26}$$

Dividing by 8 can be implemented by shifting 3 bits towards the right in the FPGA, ie:

$$\mathrm{X} = -a^2 + a^2 >> 3 \tag{27}$$

The optimized algorithm reduces a multiplier and avoids floating-point operations. Therefore, reasonable algorithm optimization can reduce the consumption of hardware resources.

3.3.2 Method of avoiding division

The division operation consumes a large amount of hardware resources. When the division operation is performed, the denominator can be approximated to a power series of 2 on the premise of allowed accuracy. After the conversion is completed, the division operation is converted into multiplication, shift, and addition.To calculate the mean value for a 9×9 image, the following method can be used to avoid division calculation:

$$\overline{f_{i,j}}(a) = \frac{\sum_{i=1}^{9}\sum_{j=1}^{9} f(a)}{81} = \frac{\sum_{i=1}^{9}\sum_{j=1}^{9} f(a)}{1024} \times \frac{1024}{81} \approx \sum_{i=1}^{9}\sum_{j=1}^{9} f(a) \times \frac{1}{1024} \times 12.6419753$$

$$\approx \frac{\sum_{i=1}^{9}\sum_{j=1}^{9} f(a)}{2^{10}} \times (8 + 4 + 0.5 + 0.125 + 0.015625 + 0.00097656)$$

$$\approx \frac{\sum_{i=1}^{9}\sum_{j=1}^{9} f(a)}{2^{10}} \times (2^3 + 2^2 + 2^{-1} + 2^{-2} + 2^{-6} + 2^{-10}) \quad (28)$$

Make $A = \frac{\sum_{i=1}^{9}\sum_{j=1}^{9} f(a)}{2^{10}}$, the error of the above equation is:

$$\Delta\% = \frac{\left| \frac{A}{2^{10}} \times (2^3 + 2^2 + 2^{-1} + 2^{-2} + 2^{-6} + 2^{-10}) - A/81 \right|}{A/81} \times 100\%$$

$$\approx 0.0030\% \quad (29)$$

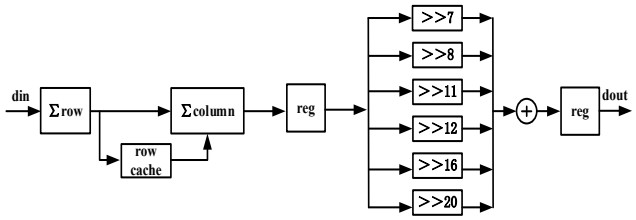

Figure 11. Mean value calculation module of 9×9 image based FPGA

In the judgment of the threshold, in order to avoid the division o peration, it can be converted into a multiplication form, such as:

$$\frac{\Delta a}{\Delta b} \le N \quad (30)$$

Equation (30) can be reconstructed as:

$$\Delta a \le N \cdot \Delta b \quad (31)$$

In this case, it only needs to judge whether Δa is less than or equal to $N\Delta b$.

According to the above method, when designing the algorithm, it can avoid floating-point operations and division operations as much as possible, thereby reducing hardware resource consumption and improving parallel computing speed.

4. CONCLUSION

The algorithm of remote sensing data processing is analyzed through hierarchy design strategy. There are a large number of public data processing modules for data processing between different remote sensing tasks, including public mathematical models, common algorithm modules, and public computing units. The public data processing module obtained by the analysis can be optimized by the algorithm for implementation in the FPGA. Among them, the main focus of analysis is laid on geometric correction, orthorectification part of the rotation matrix common mathematical model quaternion model, LU decomposition matrix inversion module and matrix parallel multiplication calculation module. The establishment of the above public modules based on FPGA can increase the parallelism of the processing of each task algorithm. According to the characteristics of the FPGA technology, in order to decrease hardware resource consumption rate and increase efficiency the calculation with acceptable accuracy, the method of avoiding floating point operation and the the division operation are analyzed. By using the power of 2 to approximate to the floating-point number and denominator, the floating-point operations and division operations can be equivalently transformed to multiply and shift operations that are advantageous for FPGA implementation. According to the above data processing method, it can ensure that the parallel processing of remote sensing data based on FPGA exhibits a more efficient computing performance.

ACKNOWLEDGEMENTS

This work was supported by grants from the Natural Science Foundation of China (grant No. 41431179, 41601365); GuangXi Key Laboratory for Spatial Information and Geomatics Program (Contract No.17-259-16-09); GuangXi innovative Development Grand under Grant 2018AA13005; GuangXi Key Research and development Program of China under Grant number 2016YFB502501; GuangXi innovative Development Grand Grant, (the grant number: GuiKe AA18118038); The State Oceanic Administration under Grant number [2014]#58; GuangXi Natural Science Foundation under grant number 2015GXNSFDA139032; and Guangxi Key Laboratory of Spatial Information and Geomatics Program under grant numbers 15-140-07-01 and 16-380-25-12.

REFERENCES

Haiyang Gu, Haitao Li, Yi Yang., 2016: *Research and Application of Parallel Processing Technology for Massive Remote Sensing Data.*Science Press.

Guoqing Zhou, Jingjin Huang, Lei Shu., 2018. An FPGA-Based P-H method on-board solution for satellite relative attitude.*Geomatics and information science of Wuhan University*, 42(12), 1838-1846.

Jingjin Huang, Gouqing Zhou, Xiang Zhou, Rongting Zhang., 2018. A New FPGA Architecture of FAST and BRIEF Algorithm for On-Board Corner Detection and Matching. *Sensors*, 1-17,doi:10.3390/s18041014.

Jingjin Huang, Guoqing Zhou, Dianjun Zhang, Guangyuan Zhang. 2018., On-Board Ortho-Rectification for Images Based on an FPGA. *International Journal of Remote Sensing.* doi.org/10.1080/01431161.2018.1500728.

Guoqing Zhou, Rongting Zhang, Dianjun Zhang, Jingjin Huang. 2018., Real-time ortho-rectification for remote-sensing Images. *International Journal of Remote Sensing.* doi.org/10.1080/01431161.2018.1488296.
.

Guoqing Zhou, Rongting Zhang., 2017. On-Board Ortho-Rectification for Images Based on an FPGA. *Remote sensing.* doi:10.3390/rs9090874.

Rongting Zhang, Guoqing Zhou, Guangyun Zhang., 2018. RPC-Based Orthorectification for Satellite Images Using FPGA. *Sensors*, doi:10.3390/s18082511.

Guoqing Zhou, Linjun Jiang, Jingjin Huang., 2018.FPGA-Based On-Board Geometric Calibration for Linear CCD Array Sensors, *Sensors*,doi.org/10.3390/s18061794.

Geng Tian, Wenbo Xu., 2008: Xilinx FPGA development practical tutorial, *Tsinghua University Press.*

Diebel J., 2006. Representing attitude: Euler angles, *unit quaternions, and rotation vectors,* 1-35.

Yu Anzhu, Jiang Ting., 2016. Dual quaternion method to solve exterior orientation parameters for satellite linear array image. *Acta Geodaetica et Cartographica Sinica.* 45(2), 186-198.

Guiming Wu., 2011. Parallel Algorithms and Architectures for Matrix Computations on FPGA. *Graduate School of National University of Defense Technology*.