'91 KACC 1991. 10. 22~24

Recognition of the 3-D Motion of a Human Arm with HIGIPS

Feng-hui Yao, Akikazu Tamaki, Kiyoshi Kato
Department of Electric, Electronic and
Computer Engineering, Faculty of Engineering
Kyushu Institute of Technology, Kitakyushu 804, Japan

Abstract

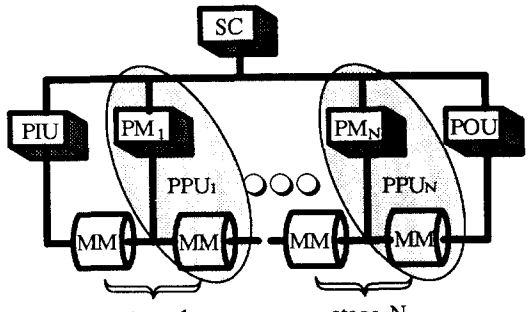
This paper gives an overview of HIGIPS design concepts and prototype HIGIPS configuration, and discusses its application to recognition of the 3-D motion of a human arm. HIGIPS which employs the combination of pipeline architecture and multiprocessor architecture, is a high-speed, high-performance and low cost **N**×**M** *multimicroprocessor parallel machine, where* **N** *is the number of pipeline stages and* **M** *is the number of processors in each stage. The algorithm to recognize the motion of a human arm with a single TV camera was developed on personal computer (NEC PC9801 series). As a constraint condition, some simple ring marks are used. Each joint of the arm is attached with a ring mark to obtain its centroid position when the arm moves. These centroid positions in the three-dimensional space are linked at each of the successive pictures of the moving arm to recover its overall motion. This algorithm takes about 2 seconds to process one image frame on the general-purpose personal computer. This paper mainly discuses how to partition this algorithm and execute on HIGIPS, and shows the speed up. From this application, it is clear that HIGIPS is an efficient machine for image processing and recognizing.*

1. Introduction

Image processing can be classified into low-level processing and high-level processing according to the degree of difficulty in processing. Low-level image processing can be performed at video rate in recent years. But high-level image processing has not yet so far. On the other hand, image processing machines can be classified into 1) fully parallel processor; 2) local parallel processor; 3) pipeline processor; and 4) multiprocessor. To perform the real-time high-level image processing and analysis, authors analyzed the advantages and disadvantages of these four kinds of machines, and on the basis of this analysis, proposed a new architecture for this purpose[1]. This new architecture is the combination of the pipeline architecture and multiprocessor architecture, i.e., the system in whole employs the pipeline architecture, and each stage of this pipeline employs multiprocessor architecture. This combination preserves the advantages of both and remedies each other's disadvantages. According to this new architecture, a prototype machine, named as HIGIPS, has been designed and developed. HIGIPS is a high-speed, high-performance and low cost $N \times M$ multimicroprocessor parallel machine. At present, a three-processing-stage HIGIPS ($N=3$) is fully operational and has been in use in studying and developing parallel image processing algorithms. This paper discusses its application to the recognition of the 3-D motion of a human arm.

2. Architecture Overview of HIGIPS

The following is a brief description of HIGIPS architecture. Much of it is summarized from [1] to [3].



PIU: Picture Input Unit, MM: Memory Module,
PM: Processor Module, SC: System Controller,
POU:Picture Output Unit,
PPU: Picture Processing Unit.

Fig.1. Block diagram of HIGIPS.

Fig.1 shows the block diagram of HIGIPS. As related before, HIGIPS is a $N \times M$ multimicroprocessor system which combined the pipeline architecture and multiprocessor architecture, i.e., the system in whole employs the pipeline architecture and each stage of the pipeline employs the multiprocessor architecture (where N is the number of pipeline stages and M is the number of microprocessors in each stage). The whole HIGIPS system consists of PIU (abbreviation of Picture Input Unit), N PPU's (Picture Processing Unit), POU (Picture Output Unit), and SC (System Controller). Therefore, the number of the entire pipeline stages will be $N+2$, if to consider PIU and POU as one stage of HIGIPS respectively. Each PPU acts as one stage of the entire pipeline. PPU$_i$ ($i=1, 2, \ldots, N$) has the identical construction, and is comprised of a PM (Processor Module) and a MM (Memory module). For convenience, PM and MM included in stage PPU$_i$ are noted as PM$_i$ and MM$_i$ separately.

The pipeline performance of HIGIPS is obtained by arranging PIU, PPU$_1$, ..., PPU$_N$, POU in a line and making them work synchronously.

MM$_i$ is the global memory (GM for short) of PPU$_i$ for storing image data. MM$_i$ consists of memory banks M$_{i1}$, M$_{i2}$, bus switches BS$_i$, BS$_i$', and it functions as a time-sharing dual port memory. For PPU$_i$ ($i=1, 2, \ldots, N$), when BS$_i$ is set *ON* and BS$_i$' *OFF*, M$_{i1}$ is connected to PM$_{i-1}$ of PPU$_{i-1}$ and M$_{i2}$ to PM$_i$ of PPU$_i$. This state of MM$_i$ is called *phase A*. Likewise, when BS$_i$ is set *OFF* and BS$_i$' *ON*, M$_{i1}$ is connected to PM$_i$ of PPU$_i$ and M$_{i2}$ to PM$_{i-1}$ of PPU$_{i-1}$. This state of MM$_i$ is called *phase B*. The pipeline performance of the entire HIGIPS is obtained by this two phase action (i.e., to set BS$_i$ and BS$_i$' *ON*/*OFF* alternately) of the time-sharing dual port memory.

PM$_i$ is a multiprocessor which includes M general-purpose microprocessors (NEC μPD2016). One of them

works as a sub-controller (SUBC), and others are slave processors (SMPU). SUBC manages all SMPU's in a stage, and SUBC's of all the stages are controlled by SC via the parallel interface bus (GP-IB). SUBC and SMPU's are composed of the local memory (LM), bus controller, bus arbiter, programmable parallel interface (8255), and serial interface (RS-232C). The different between them is only that SMPU does not include GP-IB interface. LM is for storing image data, user program, and monitor program. All SUBC's and SMPU's are autonomous processors.

Moreover, each stage has a common ROM (CROM) in which the minimum monitor program for starting up is burned. Whenever the whole system gets into the steady state, CROM is detached automatically. SUBC and SMPU are employing the same monitor program. SUBC and SMPU's can access GM randomly via the common bus. The bus requirement is arbitrated by the arbiter on each board.

PIU consists of the video-camera (TVC), A/D converter module (ADM), picture input unit controller (PIC), and image memory module (MM_{in}). Under the control of PIC, the analog signal from TVC is digitized and stored into MM_{in}. On the other hand, PIC is managed by SC via GP-IB interface bus.

POU is composed of the picture output controller (POC), image memory module (MM_{out}), frame buffer memory (FBM), D/A converter module (DAM), and RGB analog display. The processed image is output from MM_{out} to display via FBM under the control of POC. POC is also controlled by SC via GP-IB interface bus.

SC manages all autonomous processors by controlling PIC, POC, and SUBC's in each stage. And also, it provides the programming environment of the whole system. The monitor program and image processing program are all developed on it.

Table 1: Some hardware characteristics of HIGIPS

Characteristics	Individual Processor	HIGIPS (N=3, M=2)
Instruction/s (maximum)	4.0×10^6	3.6×10^7
Multiplies or divides/s	2.0×10^5 - 2.5×10^5	1.8×10^6 - 2.25×10^6
Pipeline bus rate (bytes/s)	8.42×10^5	8.42×10^5
Program load rate (bytes/s)	1.0×10^6	1.0×10^6

Some hardware characteristics of each individual HIGIPS processor and those of prototype HIGIPS ($N=3$, $M=3$) are summarized in table 1. At present, the prototype HIGIPS employs nine processors, so its processing speed is around nine times of that of an individual processor. A higher processing speed can be certainly obtained by increasing M (the number of processors in each stage). The maximum value of M for the prototype HIGIPS is eight.

3. Recognition of the 3-D Motion of a Human Arm

This section discusses the recognition of the 3-D motion of a human arm. Before going further, it is necessary to simply summarize the prototype HIGIPS again. The prototype HIGIPS consists of a PIU, three PPU's, and a POU. Each PPU is comprised of a SUBC and two SMPU's, and works as one processing stage of HIGIPS. PPU's are numbered as PPU_1, PPU_2 and PPU_3, they are corresponding to stage *1*, *2*, and *3*, accordingly. Likewise, SUBC included in PPU_i ($i=1, 2, 3$) is numbered as $SUBC_i$, and SMPU's included in PPU_i are numbered as $SMPU_{i1}$ and $SMPU_{i2}$ separately. All processors of HIGIPS (SUBC's and SMPU's) are running at 8-MHz, and the maximum input image size of PIU is 640×400 bytes. For this algorithm, the input image frame with the size of 320×200 bytes is used. On this basis, let us relate how HIGIPS performs the recognition of the 3-D motion of a human arm.

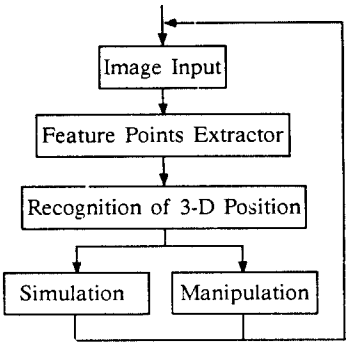


Fig.2. Block diagram of the algorithm for recognizing 3-D motion of a human arm.

3.1 Algorithm Description

The algorithm for recognizing the 3-D motion of a human arm was developed on personal computer (NEC PC-9801 series) as shown in Fig.2[5].

Before explaining this algorithm, let's relates the constraints for this algorithm. They are shown as follows.

(1) Four kinds of marks are attached to shoulder, elbow, west, and fingertip, respectively as shown in Fig.3. Mark 0 is comprised of a line and two discrete points. Mark 1 is a ring with a gap. Mark 2 and mark 3 are the same shape with mark 1 but with different diameters.

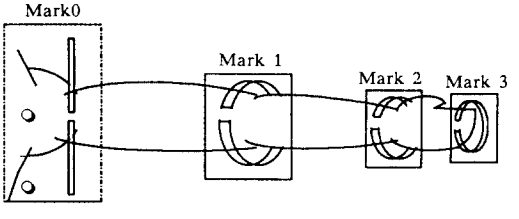


Fig.3. Shapes and positions of marks attached to the arm.

(2) The background in the image data is constant in all the intervals in which image data has been recorded.

(3) The background and the wears of the experiment person must be black and marks are white in order to extract feature points of marks easily.

(4) Experimental person can not move his body when he moves his arm in 3-D space.

The algorithm will not work properly if the constraints given above are not satisfied. Each block in Fig.2 is explained in following.

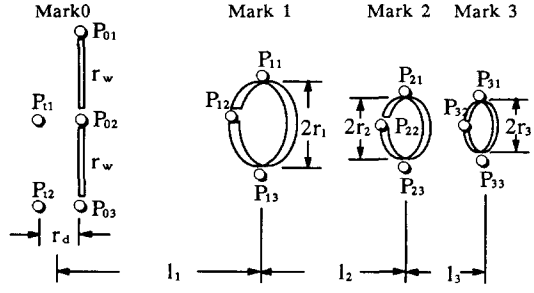


Fig.4. Feature points of each mark.

(1) Image Input

Video signal from the video camera is digitized and saved in the memory.

(2) Feature Points Extractor

This block extracts the feature points from raw gray-level image data. Feature points of each marks are shown in Fig.4. P_{tj}, P_{in} (j=1, 2, i=0, 1, 2, 3, and n=1, 2, 3) are the feature points of each marks separately. r_d, r_w, l_i (when i=0, l_0 does not exist), and r_n are known parameters.

(3) Recognition of 3-D Motion

The task to recognize the motion of a human arm is to find the positions of shoulder joint, elbow joint, west joint, and fingertip in 3-D space.

The coordinates of shoulder in 3-D space can be found according mark 0. As shown in Fig.5, the video camera is set at Q in 3-D space. P_{tj} and P_{0n} are projected onto XY-plane. Their corresponding points in XY-plane are shown by P_{tj}' and P_{0n}' (j=1, 2, and n=1, 2, 3) respectively. These points can be extracted from the input image. According to the coordinates of Q, P_{tj}' and P_{0n}', the coordinates of P_{tj} and P_{0n} in 3-D space can be calculated as follows.

Suppose that the coordinates of shoulder joint and video camera are $P_s(X_s, Y_s, Z_s)$, $Q(X_q, Y_q, Z_q)$, respectively, those of feature points are $P_{ij}(X_{ij}, Y_{ij}, Z_{ij})$ and $P_{0n}(X_{0n}, Y_{0n}, Z_{0n})$ accordingly, and coordinates of feature points projected on XY-plane are $P_{tj}'(X'_{tj}, Y'_{tj}, Z'_{tj})$ and $P_{0n}'(X'_{0n}, Y'_{0n}, Z'_{0n})$ separately, the following equations exist.

$$\overrightarrow{QP}_{0n} = k_n \overrightarrow{QP}'_{0n} \quad (n=1, 2, 3) \tag{1}$$

where $\overrightarrow{AB}$ represents a vector starting from point $A(X_a, Y_a, Z_a)$ to point $B(X_b, Y_b, Z_b)$ in 3-D space. Because P_2 is in the middle of P_1 and P_3, then,

$$\overrightarrow{QP}_{02} = 1/2(\overrightarrow{QP}_{01} + \overrightarrow{QP}_{03}) \tag{2}$$

And because the length of mark 0 is r_w, then,

$$|\overrightarrow{QP}_{01} - \overrightarrow{QP}_{02}| = r_w \tag{3}$$

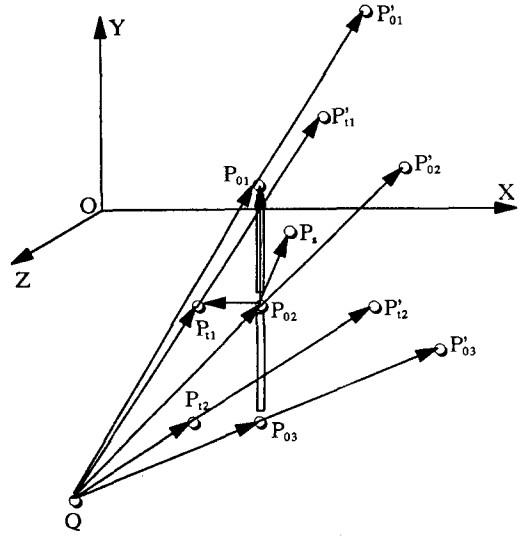


Fig.5. Relationship between the position of shoulder joint and feature points.

From equations (1), (2), and (3), coordinates of P_{0n} can be obtained.

Likewise, because $\overrightarrow{P_{02}P_{01}}$ is at right angles to $\overrightarrow{P_{02}P_{t1}}$, then,

$$\overrightarrow{P_{02}P_{01}} \bullet \overrightarrow{P_{02}P_{t1}} = 0 \tag{4}$$

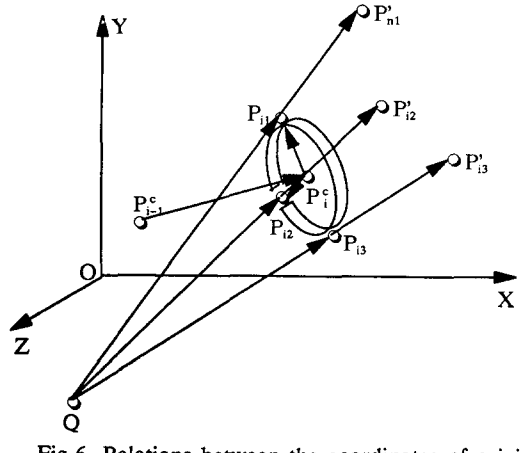


Fig.6. Relations between the coordinates of a joint and those of feature points of its ring mark.

And because $\overrightarrow{QP}_{tj}$ and $\overrightarrow{QP}'_{tj}$ lie on the same line, then,

$$\overrightarrow{QP}_{tj} = k_{tj} \overrightarrow{QP}'_{tj} \quad (j=1, 2) \tag{5}$$

From equation (4) and (5), coordinates of P_{ij} can be found.

From the coordinates of P_{ij} and P_{on} obtained above, the coordinates P_s of shoulder joint in 3-D space can be found according the following three equations.

$$\overrightarrow{P_{02}P_{01}} \bullet \overrightarrow{P_{02}P_s} = 0 \tag{6}$$

$$\overrightarrow{P_{02}P_{11}} \bullet \overrightarrow{P_{02}P_s} = 0 \tag{7}$$

$$|\overrightarrow{P_{02}P_s}| = r_s \tag{8}$$

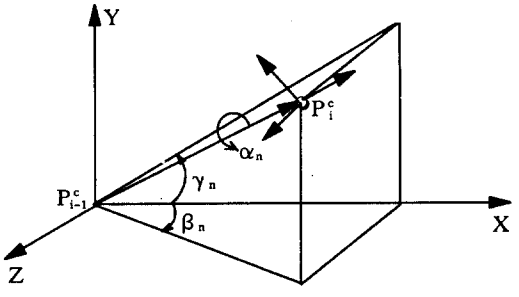


Fig.7. Relations between controid coordinates and angles to X-, Y-, and Z-axis of a joint.

The coordinates of P_s is employed to find other coordinates of elbow joint, west joint, and fingertip. Fig.6 shows the relations between the feature points of mark *i* (i=1, 2, 3) and other joints. Coordinates of joint (centroids of ring mark) is represented as $P_i^c(X_i^c, Y_i^c, Z_i^c)$ (when i=0, $P_i^c(X_i^c, Y_i^c, Z_i^c)=P_s(X_s, Y_s, Z_s)$). Then there exists the following equations.

$$\overrightarrow{QP}_{in} = k_n \overrightarrow{QP'}_{in} \tag{9}$$

$$\overrightarrow{P_{i-1}^c P_i^c} \bullet \overrightarrow{P_i^c P}_{in} = 0 \qquad (n=1, 2, 3) \tag{10}$$

$$|\overrightarrow{P_{i-1}^c P_i^c}| = l_i \qquad (i=1, 2, 3) \tag{11}$$

$$|\overrightarrow{P_i^c P}_{in}| = r_i \qquad (i=1, 2, 3) \tag{12}$$

From equations (9), (10), (11), and (12), coordinates of P_i^c can be found.

Angles of each joint rotated around X-, Y-, Z-axis can be calculated as follows. As shown in Fig.7, P_i^c can be thought as that it is translated l_i from P_{i-1}^c and is rotated α_i, β_i, γ_i around X-, Y-, and Z-axis, accordingly. Therefore, there exists,

$$\begin{bmatrix} X_i^c - X_{i-1}^c \\ Y_i^c - Y_{i-1}^c \\ Z_i^c - Z_{i-1}^c \\ 1 \end{bmatrix} = C(\gamma_i, \beta_i, \alpha_i, l_i) \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{13}$$

From equation (13), β_i and γ_i can be obtained. And α_n is calculated from the following equation.

$$\begin{bmatrix} X_b - X_{i-1}^c \\ Y_b - Y_{i-1}^c \\ Z_b - Z_{i-1}^c \\ 1 \end{bmatrix} = C(\gamma_i, \beta_i, \alpha_i, l_i) \begin{bmatrix} 0 \\ 0 \\ r_i \\ 1 \end{bmatrix} \tag{14}$$

(4) Simulation/Manipulation

Coordinates of each joint and their angles to X-, Y-, and Z-axis are displayed. They are used for recovering the 3-D motion of the human arm.

3.2 Algorithm Partitioning

For any image processing, its corresponding processing flow graph can be obtained. The processing flow graph for this algorithm is shown in Fig.8. A node indicated by P_k (k=1, 2, ..., 12) in the flow graph represents a task which is a basic processing unit, and a branch indicated by (X,Y) which means that it starts from X and ends at Y represents the relationship between the processing units, e.g., (P_4,P_5) means that the task P_5 can not be executed until task P_3 is finished.

The following gives the meaning of each P_k in flow graph L1.

(1) P_s and P_e

They mark the "begin" and "end" of the whole processing.

(2) P_1, P_2 and P_3

They are the processing to extract the feature points of mark 1, mark 2, and mark 3, respectively.

(3) P_4 and P_5

P_4 is the calculation of coordinates of feature points of mark 1. Details are shown as follows. From equation (9), the following equations are obtained.

$$X_n=(X_n'-X_q)k_n+X_q \tag{15}$$
$$Y_n=(Y_n'-Y_q)k_n+Y_q \tag{16}$$
$$Z_n=-Z_q k_i+Z_q \tag{17}$$

And from equation (10), (11), and (12), the following equation is obtained.

$$(X_n-X_0^c)^2+(Y_n-Y_0^c)^2+(Z_n-Z_0^c)^2=l_1^2+r_1^2 \tag{18}$$

From equation (15), (16), (17), and (18), X_n, Y_n, and Z_n can be obtained (n=1, 2, 3), which are the coordinates of the feature points of mark 1 in 3-D space.

P_5 is to find centroid coordinates of mark 1 in 3-D space. Equation (9), (10), (11), and (12) are rewritten as follows.

$$(X_n-X_1^c)^2+(Y_n-Y_1^c)^2+(Z_n-Z_1^c)^2=r_1^2 \tag{19}$$

$$\begin{aligned}&(X_n-X_0^c)X_1^c+(Y_n-Y_0^c)Y_1^c+(Z_n-Z_0^c)Z_1^c\\&=1/2\{(X_n^2+Y_n^2+Z_n^2)-[(X_0^c)^2+(Y_0^c)^2+(Z_0^c)^2]\\&+l_1^2-r_1^2\end{aligned} \tag{20}$$

From equation (19) and (20), the centroid coordinates (X_1^c, Y_1^c, Z_1^c) of mark 1 in 3-D space can be get.

(4) P_6 and P_7

P_6 represents the calculation of coordinates of feature points of mark 2. They can be found from the following equations.

$$X_n=(X_n'-X_q)k_n+X_q \quad (21)$$
$$Y_n=(Y_n'-Y_q)k_n+Y_q \quad (22)$$
$$Z_n=-Z_qk_i+Z_q \quad (23)$$
$$(X_n-X_1^c)^2+(Y_n-Y_1^c)^2+(Z_n-Z_1^c)^2=l_2^2+r_2^2 \quad (24)$$

where n=1, 2, 3.

P_7 indicates the calculation of centroid coordinates (X_2^c, Y_2^c, Z_2^c) of mark 2 in 3-D space. They can be get from the following equations.

$$(X_n-X_2^c)^2+(Y_n-Y_2^c)^2+(Z_n-Z_2^c)^2=r_1^2 \quad (25)$$

$$(X_n-X_0^c)X_2^c+(Y_n-Y_0^c)Y_2^c+(Z_n-Z_0^c)Z_2^c$$
$$=1/2\{(X_n^2+Y_n^2+Z_n^2)-[(X_0^c)^2+(Y_0^c)^2+(Z_0^c)^2]$$
$$+l_2^2-r_2^2 \quad (26)$$

(5) P_8 and P_9

P_8 is to calculate coordinates of feature points of mark 3. They can be found from the following equations.

$$X_n=(X_n'-X_q)k_n+X_q \quad (27)$$
$$Y_n=(Y_n'-Y_q)k_n+Y_q \quad (28)$$
$$Z_n=-Z_qk_i+Z_q \quad (29)$$
$$(X_n-X_2^c)^2+(Y_n-Y_2^c)^2+(Z_n-Z_2^c)^2=l_3^2+r_3^2 \quad (30)$$

where n=1, 2, 3.

P_9 shows the calculation of centroid coordinates (X_3^c, Y_3^c, Z_3^c) of mark 3 in 3-D space. They can be get from the following equations.

$$(X_n-X_3^c)^2+(Y_n-Y_3^c)^2+(Z_n-Z_3^c)^2=r_1^2 \quad (31)$$

$$(X_n-X_0^c)X_3^c+(Y_n-Y_0^c)Y_3^c+(Z_n-Z_0^c)Z_3^c$$
$$=1/2\{(X_n^2+Y_n^2+Z_n^2)-[(X_0^c)^2+(Y_0^c)^2+(Z_0^c)^2]$$
$$+l_2^2-r_2^2 \quad (32)$$

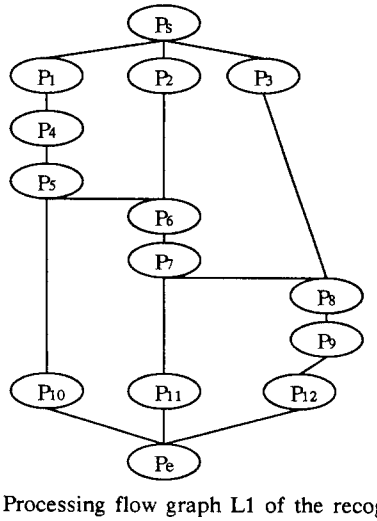


Fig.8. Processing flow graph L1 of the recognition of 3-D motion of a human arm.

(6) P_{10}

This is the operation to find the angles of the centroid of mark 1. From equation (13) and (14), β_1, γ_1, and α_1 are shown as follows.

$$\beta_1=\sin^{-1}\left(-\frac{Z_1^c-Z_0^c}{l_1}\right) \quad (33)$$

$$\gamma_1=\sin^{-1}\left(-\frac{Y_1^c-Y_0^c}{l_1\cos\beta_1}\right) \quad (34)$$

$$\alpha_1=\sin^{-1}\left(\frac{(Z_b-Z_0^c)\sin\beta_1\sin\gamma_1-(Y_b-Y_0^c)\cos\beta_1+l_1\sin\gamma_1}{Z_b\cos\beta_1\cos\gamma_1}\right) \quad (35)$$

(7) P_{11}

This gives the angles of the centroid of mark 2. According to equation (13) and (14), β_2, γ_2, and α_2 are shown in the below.

$$\beta_2=\sin^{-1}\left(-\frac{Z_2^c-Z_1^c}{l_2}\right) \quad (36)$$

$$\gamma_2=\sin^{-1}\left(-\frac{Y_2^c-Y_1^c}{l_2\cos\beta_2}\right) \quad (37)$$

$$\alpha_2=\sin^{-1}\left(\frac{(Z_b-Z_1^c)\sin\beta_2\sin\gamma_2-(Y_b-Y_1^c)\cos\beta_2+l_2\sin\gamma_2}{Z_b\cos\beta_2\cos\gamma_2}\right) \quad (38)$$

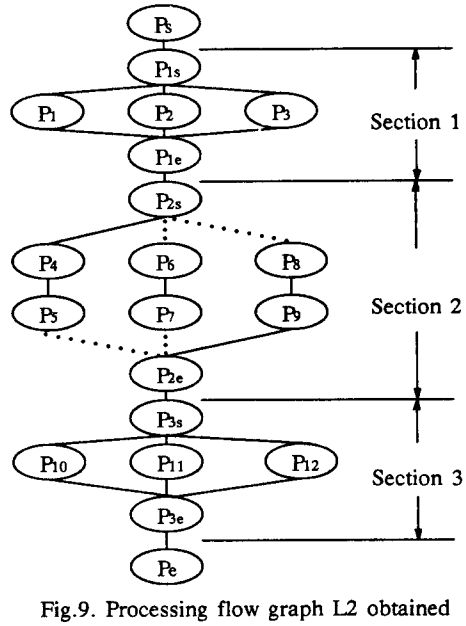


Fig.9. Processing flow graph L2 obtained from L1 by partitioning.

(8) P_{12}

This is to figure out the angles of the centroid of mark 3. β_3, γ_3, and α_3 are given by equation (13) and (14) as follows.

$$\beta_3 = \sin^{-1}\left(-\frac{Z_3^c - Z_2^c}{l_3}\right) \tag{39}$$

$$\gamma_3 = \sin^{-1}\left(-\frac{Y_3^c - Y_2^c}{l_3 \cos\beta_3}\right) \tag{40}$$

$$\alpha_3 = \sin^{-1}\left(\frac{(Z_b - Z_2^c)\sin\beta_3 \sin\gamma_3 - (Y_b - Y_2^c)\cos\beta_3 + l_3 \sin\gamma_3}{Z_b \cos\beta_3 \cos\gamma_3}\right) \tag{41}$$

To run this algorithm on prototype HIGIPS machine, it is necessary to partition it into N sections (here, N=3). The partitioned flow graph L2 is shown in Fig.9. Criteria for algorithm partitioning can be refered at [4].

Dot lines in L2 show the local pseudo pipeline performance in section 2. From Fig.8, it is clear that tasks from P_4 to P_9 can not be performed in parallel. This is because that mark 1, mark 2, and mark 3 are linked in an input image scene. But thinking about the real-time recognizing, centroid coordinates calculation of mark 1 in image frame *i* can be performed parallel with those of mark 2 in image frame *i–1*, and those of mark 3 in image *i–2*. Thereby, if to treat three successive image scenes at once (when frame *i+1* is provided, frame *i–2* will not be used anymore), P_4, P_6, and P_8 can performed in parallel. This is why we draw the partitioned flow graph as shown in section 2 of Fig.9.

3.3 Processing on Each Stage

According to the partitioned flow graph L2, system controller (SC) of HIGIPS assigns the processing from P_{1s} to P_{1e} to stage 1, P_{2s} to P_{2e} to stage 2, and P_{3s} to P_{3e} to stage 3. P_s and P_e are executed by SC itself.

P_{is} and P_{ie} (i=1, 2 and 3) mean the "begin" and "end" of stage *i*. When SUBC*i* gets "begin" message from SC, it will execute P_{is} which is to order SMPU's to start processing in its stage. When all processing assigned to its stage are finished, SUBC*i* will execute P_{ie} and sends "end" message to SC.

In stage 1, P_1, P2, and P3 are assigned. They extract feature points for mark 1, mark 2, and mark 3, respectively. But as a preparation, stage 1 must firstly extract the feature points for mark 0, and calculate coordinates of them in 3-D space, and gives the position of shoulder joint. This processing is only performed once because the shoulder does not move. The feature points of mark 1, mark 2, and mark 3, and the position of shoulder joint (P_s) are transferred to stage 2.

In stage 2, tasks from P_4 to P_9 are assigned. It calculates the coordinates of mark 1, mark 2, and mark 3.

In stage 3, tasks from P_{10} to P_{12} are assigned. The angles to X-, Y-, Z-axis are figured out.

Coordinates and angles of every joint are moved to POU which recovers the 3-D motion of a human arm and shows on the display.

4. Summary and Future Work

The maximum *K* and *L* for PIU of prototype HIGIPS is 640 and 400, respectively. But for this specific problem, *K* and *L* is set to 320 and 200, accordingly. Around N×M times speed up is obtained for this problem.

The number of slave processors in each stage of the prototype HIGIPS is 2, but it can be easily extended up to 7. Then the number of processors in each stage will be eight. And because the prototype HIGIPS has three processing stages, the processing speed of HIGIPS, at most, will be about 24 times higher than that of a individual processor. With the extended prototype HIGIPS, a higher processing rate can be obtained. This paper concentrated on the recognition of 3-D motion of a human arm. HIGIPS can be applied to other kinds of processing. These are left to do in the future.

Acknowledgments

Authors would like to appreciate Dr. & Prof. S. Ishikawa for his permission to use the algorithm for recognition of the 3-D motion of a human arm.

Reference

[1] Feng-hui Yao, Akikazu Tamaki and Kiyoshi Kato, The development of HIGIPS — a High-speed General-purpose Image Processing System. In: *Proceedings of International Computer Symposium*, Taiwan, China. December 17–19, 1990, pp.278–283.

[2] Feng-hui Yao, Akikazu Tamaki and Kiyoshi Kato, A High-speed General-purpose Image Processing System — HIGIPS. To appear in: *The Transactions of the Institute of Electronics, Information and Communication Engineers.*

[3] Feng-hui Yao, Akikazu Tamaki and Kiyoshi Kato, Image Processing on HIGIPS. To appear in: *Proceedings of International Conference on Information and Systems*, **AMSE** October 9–11, 1991, Hangzhou, China.

[4] Feng-hui Yao, Akikazu Tamaki and Kiyoshi Kato, An Application of HIGIPS to Recognition of Pendulum like Motion. To appear in: *Proceedings of* **IECON**'91, October 28 to November 1, 1991.

[5] Seiji Ishikawa, Noriaki Harada and Kiyoshi Kato, Recognizing a 3-D Motion of a Human Arm by Monocular Vision. In *Human Interface*, 1988, vol.3, pp.107-110 (in Japanese).